

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

TRACE MODE 6

Интегрированная SCADA/HMI-SOFTLOGIC-MES-
EAM-HRM-система для разработки АСУ ТП,
АСКУЭ и систем управления производством

Том 2

18-е издание (к релизу **6.10**)

ТУ 5043-001-18957709-2015

Москва, 2015

AdAstra Research Group, Ltd.

TRACE MODE, T-FACTORY.exe, АдАстра, AdAstrA, autobuilding и автопостроение являются зарегистрированными торговыми марками AdAstra Research Group, Ltd.

Торговые марки, торговые имена, сервисные марки и сервисные имена, зарегистрированные другими компаниями и использованные в данном руководстве, принадлежат соответствующим компаниям.

© 1998-2015 AdAstra Research Group, Ltd.

Все права защищены.

Содержание

| | |
|--|-----------|
| Глава 7 | 13 |
| Программирование алгоритмов | 13 |
| Программирование алгоритмов в TRACE MODE 6..... | 14 |
| Операции с программами..... | 14 |
| Подключение программы к проекту..... | 14 |
| Выполнение программы в реальном времени | 15 |
| Окно структуры программы..... | 16 |
| Выбор языка программирования..... | 16 |
| Типовые инструменты редактирования программ..... | 17 |
| Масштабирование диаграмм | 17 |
| Инструменты для работы с закладками..... | 17 |
| Создание элементов программ с помощью табличных редакторов | 18 |
| Настройка редакторов и отладчика | 21 |
| Описание языка Техно ST..... | 25 |
| Лексическая структура языка Техно ST..... | 25 |
| Идентификаторы Техно ST..... | 25 |
| Ключевые слова Техно ST..... | 25 |
| Разделители Техно ST..... | 26 |
| Комментарии Техно ST | 26 |
| Синтаксис Техно ST | 26 |
| Основная точка входа в программу..... | 27 |
| Переменные и константы Техно ST | 27 |
| Определение переменных и констант..... | 28 |
| Числовые константы Техно ST..... | 30 |
| Строковые константы Техно ST..... | 30 |
| Особенности вычислений..... | 31 |
| Операторы языка Техно ST | 31 |
| Символьные операторы | 31 |
| Операторы Техно ST..... | 34 |
| Функции Техно ST..... | 42 |
| Стандартные функции C в ST-программе | 42 |
| Специальные функции в ST-программе | 42 |
| Пользовательские функции Техно ST | 43 |
| Внешние библиотеки функций..... | 47 |
| Массивы Техно ST..... | 48 |
| Одномерные массивы | 49 |
| Многомерные массивы | 49 |
| Структуры Техно ST..... | 51 |
| Описание языка Техно IL | 53 |
| Синтаксис Техно IL | 53 |
| Операнды Техно IL..... | 54 |
| Операторы и модификаторы Техно IL..... | 54 |
| Редактирование SFC-программ..... | 59 |
| Выделение элементов алгоритма SFC..... | 59 |
| Задание SFC-шагов и SFC-условий | 60 |

| | |
|--|------------|
| Редактирование алгоритма SFC | 60 |
| Редактирование FBD-программ..... | 64 |
| Размещение FBD-блоков в рабочем поле редактора..... | 65 |
| Редактирование диаграммы FBD-блоков | 66 |
| Привязка входов и выходов FBD-диаграммы | 67 |
| Описание FBD-блоков..... | 69 |
| Раздел 'Логические' | 69 |
| Раздел 'Побитовые' | 71 |
| Раздел 'Арифметические'..... | 78 |
| Раздел 'Тригонометрические' | 82 |
| Раздел 'Алгебраические' | 85 |
| Раздел 'Функции сравнения' | 92 |
| Раздел 'Функции выбора' | 96 |
| Раздел 'Триггеры и счетчики' | 106 |
| Раздел 'Генераторы' | 112 |
| Раздел 'Управление' | 115 |
| Раздел 'Ввод/вывод. Переходы' | 140 |
| Раздел 'Регулирование' | 142 |
| Редактирование LD-программ | 165 |
| Размещение блоков в рабочем поле LD-редактора | 167 |
| Редактирование LD-диаграммы | 168 |
| Привязка входов и выходов LD-диаграммы и задание связанных переменных | 168 |
| Описание LD-блоков | 169 |
| Раздел 'Контакты' | 169 |
| Раздел 'Катушки'..... | 170 |
| Создание пользовательских функциональных блоков..... | 172 |
| Отладка программ | 173 |
| Автоматическое выделение конструкций языка | 174 |
| Меню 'Программа' и панель инструментов отладчика..... | 174 |
| Удаленная отладка..... | 175 |
| Окно просмотра переменных | 176 |
| Диалог 'Быстрый просмотр'..... | 177 |
| Окно стека вызовов функций | 178 |
| Окно 'Сообщения'..... | 178 |
| Глава 8..... | 181 |
| Разработка графического интерфейса | 181 |
| Редактор представления данных | 182 |
| Режимы работы РПД..... | 182 |
| Главное меню и панели инструментов РПД..... | 183 |
| Задание параметров РПД..... | 185 |
| Контекстные меню РПД..... | 186 |
| Таблица 'Графические элементы' | 187 |
| Операции с графическими экранами..... | 190 |
| Задание параметров графического экрана..... | 190 |
| Особенности вызова графического экрана..... | 192 |
| Специальные операции с графическими экранами | 194 |
| Сохранение экрана в файл..... | 195 |
| Перезагрузка шаблона экрана в реальном времени | 195 |
| Операции с графическими слоями | 196 |

| | |
|---|-----|
| Создание и удаление графических слоев..... | 196 |
| Управление видимостью графических слоев | 196 |
| Блокировка редактирования графического слоя | 197 |
| Z-позиционирование графических слоев..... | 197 |
| Выделение элементов слоя | 197 |
| Операции с графическими элементами..... | 199 |
| Размещение ГЭ | 199 |
| Выделение ГЭ | 201 |
| Перемещение и масштабирование ГЭ | 201 |
| Удаление ГЭ | 202 |
| Копирование и вставка ГЭ..... | 202 |
| Поворот ГЭ | 203 |
| Позиционирование ГЭ..... | 203 |
| Позиционирование узловых точек ГЭ | 205 |
| Тиражирование ГЭ | 206 |
| Операции с аргументами в РПД..... | 207 |
| Задание типовых свойств ГЭ | 208 |
| Статические атрибуты ГЭ..... | 209 |
| Динамизация атрибута ГЭ | 214 |
| Буфер обмена окна 'Свойства объекта' | 220 |
| Сочетания клавиш в окне 'Свойства объекта' | 220 |
| Динамические свойства ГЭ | 221 |
| Динамическая заливка ГЭ..... | 221 |
| Динамическое перемещение ГЭ | 223 |
| Динамическое масштабирование ГЭ | 225 |
| Динамическое вращение ГЭ | 226 |
| Динамический контур ГЭ | 227 |
| Функции управления ГЭ..... | 229 |
| Функция передачи значения..... | 231 |
| Функция управления видимостью ГЭ | 233 |
| Функция перехода на экран..... | 233 |
| Функция ввода комментария..... | 234 |
| Функция отправки всплывающей подсказки | 234 |
| Функция отправки строки | 234 |
| Функция 'Выполнить'..... | 234 |
| Описание встроенных графических элементов | 235 |
| Группа ГЭ 'Линии' | 235 |
| ГЭ 'Линия' | 235 |
| Группа ГЭ 'Текст'..... | 236 |
| ГЭ 'Текст' | 236 |
| Группа ГЭ 'Каналы' | 237 |
| ГЭ 'Канал'..... | 237 |
| Группа ГЭ 'Меню' | 238 |
| ГЭ 'Меню управления' | 238 |
| Группа ГЭ 'Ломаные и кривые' | 239 |
| ГЭ 'Ломаная линия'..... | 239 |
| ГЭ 'Многоугольник' | 239 |
| ГЭ 'Ломаная с заливкой' | 239 |
| ГЭ 'Разомкнутая кривая' | 239 |
| ГЭ 'Замкнутая кривая'..... | 239 |
| Группа ГЭ 'Прямоугольники' | 240 |
| ГЭ 'Контур'..... | 240 |

| | |
|--|-----|
| ГЭ 'Прямоугольник' | 240 |
| ГЭ 'Панель' | 240 |
| ГЭ 'Рамка' | 240 |
| Специфические атрибуты ГЭ 'Панель' и 'Рамка' | 240 |
| Группа ГЭ 'Плоские фигуры' | 241 |
| ГЭ 'Плоский клапан' | 241 |
| ГЭ 'Треугольник' | 241 |
| ГЭ 'Овал' | 241 |
| ГЭ 'Стрелка' | 241 |
| ГЭ 'Эллипс, сектор' | 241 |
| ГЭ 'Плоский клапан 2' | 242 |
| ГЭ 'Плоский насос' | 242 |
| Группа ГЭ 'Ресурсы' | 243 |
| ГЭ 'Текстовый ресурс' | 243 |
| ГЭ 'Растровое изображение' | 243 |
| ГЭ 'Векторное изображение' | 244 |
| ГЭ 'Видеоклип' | 244 |
| ГЭ 'Рисунок из файла' | 245 |
| ГЭ 'Текст из файла' | 245 |
| ГЭ 'Стандартный видеоклип' | 246 |
| Группа ГЭ 'Объемные фигуры' | 248 |
| Общие специфические атрибуты объемных ГЭ | 248 |
| ГЭ 'Цилиндр' | 249 |
| ГЭ 'Сфера' | 249 |
| ГЭ 'Конус' | 250 |
| ГЭ 'Тор' | 250 |
| ГЭ 'Пирамида' | 251 |
| ГЭ 'Емкость' | 251 |
| ГЭ 'Клапан' | 251 |
| ГЭ 'Насос' | 252 |
| ГЭ 'Труба' | 252 |
| ГЭ 'Рельефный конус' | 252 |
| ГЭ 'Криволинейный конус' | 253 |
| ГЭ 'Градиент' | 253 |
| Группа ГЭ 'Кнопки' | 254 |
| ГЭ 'Кнопка' | 254 |
| ГЭ 'Группа кнопок' | 254 |
| ГЭ 'Картинка-кнопка' | 255 |
| Группа ГЭ 'Выключатели' | 257 |
| ГЭ 'Выключатель' | 257 |
| Группа ГЭ 'Тренды' | 258 |
| ГЭ 'Тренд' | 258 |
| ГЭ 'Архивный тренд' | 263 |
| ГЭ 'Тренд XY' | 264 |
| ГЭ 'Архивная гистограмма' | 265 |
| Группа ГЭ 'Объекты' | 268 |
| ГЭ 'Объект' | 268 |
| ГЭ 'Объект в окне' | 270 |
| ГЭ 'Ссылка на экран' | 270 |
| Группа ГЭ 'Таблицы' | 272 |
| ГЭ 'Переключатель каналов' | 272 |
| ГЭ 'События' | 274 |

| | |
|--|-----|
| ГЭ 'Архивная таблица' | 277 |
| ГЭ 'Архивная таблица 2' | 278 |
| ГЭ 'База данных' | 278 |
| Группа ГЭ 'ActiveX' | 280 |
| Группа ГЭ 'Свободные формы' | 281 |
| Группа ГЭ 'Приборы' | 284 |
| ГЭ 'Ползунок' | 284 |
| ГЭ 'Стрелочный прибор' | 286 |
| Группа ГЭ 'Диаграммы' | 288 |
| ГЭ 'Круговая диаграмма' | 288 |
| ГЭ 'Гистограмма' | 289 |
| Группа ГЭ 'Дата и время' | 291 |
| ГЭ 'Дата и время' | 291 |
| Группа ГЭ 'Отчет тревог' | 293 |
| ГЭ 'Строка ОТ' | 293 |
| ГЭ 'ОТ узла' | 293 |
| Группа ГЭ 'T-FACTORY' | 296 |
| ГЭ 'Диаграмма Ганта' | 296 |
| Группа ГЭ 'Зоны' | 299 |
| ГЭ 'Зона' | 299 |
| Группа ГЭ 'Элементы зданий' | 300 |
| ГЭ 'Стена' | 300 |
| ГЭ 'Ломаная стена' | 300 |
| ГЭ 'Окно' | 300 |
| ГЭ 'Дверь' | 300 |
| ГЭ 'Соединитель стен' | 300 |
| ГЭ 'Лестница' | 300 |
| ГЭ 'Прямоугольное помещение' | 301 |
| ГЭ 'Забор' | 301 |
| Группа ГЭ 'Электрические элементы зданий' | 302 |
| ГЭ 'Трансформатор' | 302 |
| ГЭ 'Рубильник' | 302 |
| ГЭ 'Заземление' | 303 |
| Группа ГЭ 'Кондиционирование' | 304 |
| ГЭ 'Поток' | 304 |
| ГЭ 'Заслонка' | 304 |
| ActiveX в TRACE MODE 6 | 305 |
| Свойства РПД как ActiveX-контейнера | 305 |
| Взаимодействие с компонентами ActiveX | 305 |
| Интерфейс IDispatch | 305 |
| Custom-интерфейсы для ActiveX | 307 |
| Дополнительные интерфейсы для ActiveX | 308 |
| Операции с графическими объектами | 310 |
| Операции с ресурсными библиотеками | 311 |
| Библиотека избранных ГЭ и библиотека избранных eГЭ | 314 |
| Графические панели | 315 |
| Операции с аргументами в eРПД | 316 |
| Описание элементов графических панелей | 316 |
| Группа eГЭ 'Линии' | 317 |
| eГЭ 'Линия' | 317 |
| Группа eГЭ 'Текст' | 317 |

| | |
|---|-----|
| еГЭ 'Текст' | 317 |
| Группа еГЭ 'Меню' | 318 |
| еГЭ 'Меню управления' | 318 |
| Группа еГЭ 'Ломаные и кривые' | 318 |
| еГЭ 'Ломаная линия' | 318 |
| еГЭ 'Многоугольник' | 318 |
| еГЭ 'Ломаная с заливкой' | 318 |
| еГЭ 'Разомкнутая кривая' | 318 |
| еГЭ 'Замкнутая кривая' | 318 |
| Группа еГЭ 'Прямоугольники' | 319 |
| еГЭ 'Контур' | 319 |
| еГЭ 'Панель' | 319 |
| еГЭ 'Рамка' | 319 |
| Специфические атрибуты еГЭ 'Панель' и 'Рамка' | 319 |
| Группа еГЭ 'Плоские фигуры' | 319 |
| Общие специфические атрибуты плоских еГЭ | 319 |
| еГЭ 'Прямоугольник' | 320 |
| еГЭ 'Плоский клапан' | 320 |
| еГЭ 'Треугольник' | 320 |
| еГЭ 'Овал' | 320 |
| еГЭ 'Стрелка' | 320 |
| еГЭ 'Эллипс, сектор' | 320 |
| Группа еГЭ 'Ресурсы' | 321 |
| еГЭ 'Текстовый ресурс' | 321 |
| еГЭ 'Растровое изображение' | 321 |
| еГЭ 'Видеоклип' | 321 |
| еГЭ 'Стандартный видеоклип' | 321 |
| Группа еГЭ 'Объемные фигуры' | 321 |
| Общие специфические атрибуты объемных еГЭ | 321 |
| еГЭ 'Цилиндр' | 321 |
| еГЭ 'Сфера' | 322 |
| еГЭ 'Конус' | 322 |
| еГЭ 'Тор' | 322 |
| еГЭ 'Пирамида' | 322 |
| еГЭ 'Емкость' | 322 |
| еГЭ 'Клапан' | 322 |
| еГЭ 'Насос' | 323 |
| еГЭ 'Труба' | 323 |
| еГЭ 'Рельефный конус' | 323 |
| еГЭ 'Криволинейный конус' | 323 |
| еГЭ 'Градиент' | 323 |
| Группа еГЭ 'Кнопки' | 323 |
| еГЭ 'Кнопка' | 324 |
| еГЭ 'Кнопка XOR' | 324 |
| еГЭ 'Переход' | 325 |
| Группа еГЭ 'Выключатели' | 325 |
| еГЭ 'Выключатель' | 325 |
| Группа еГЭ 'Тренды' | 325 |
| еГЭ 'Тренд' | 325 |
| Группа еГЭ 'Дата и время' | 326 |
| еГЭ 'Дата и время' | 326 |
| Группа еГЭ 'Значение аргумента' | 327 |

| | |
|---|------------|
| еГЭ 'Ввод значения' | 327 |
| Группа еГЭ 'Гистограммы' | 327 |
| еГЭ 'Гистограмма' | 327 |
| Группа еГЭ 'Отчет тревог' | 327 |
| еГЭ 'Строка ОТ' | 327 |
| еГЭ 'ОТ узла' | 328 |
| Глава 9 | 329 |
| Архивирование | 329 |
| Отчет тревог узла | 330 |
| Формат строки ОТ | 331 |
| Системные сообщения | 333 |
| Сообщения по каналам | 335 |
| Сообщения по каналам FLOAT и DOUBLE FLOAT | 335 |
| Сообщения по каналам HEX16 и HEX32 | 336 |
| Сообщения по каналу TIME | 337 |
| Сообщения по каналу СОБЫТИЕ | 337 |
| Сообщения по каналам ПЕРСОНАЛ и ПОЛЬЗОВАТЕЛЬ | 338 |
| Сообщения по каналу ЕДИНИЦА ОБОРУДОВАНИЯ | 338 |
| Сообщения по каналу М-РЕСУРС | 339 |
| Сообщения по каналу D-РЕСУРС | 339 |
| Другие виды сообщений | 340 |
| Генерация сообщений с помощью переменной MESSAGE | 340 |
| Запись в отчет тревог сообщений оператора | 340 |
| Запись аргументов CALL.STRING в отчет тревог | 341 |
| Архивы SIAD | 342 |
| Уменьшение объема SIAD при записи с нарушением порядка временных меток | 343 |
| Уменьшение записей канала FLOAT в SIAD | 343 |
| Выборка и обработка данных SIAD | 344 |
| Временной интервал выборки | 345 |
| Обработка данных локального архива по каналу | 348 |
| Быстрая выборка данных из локального архива | 350 |
| Выборка данных из локального архива по каналу | 352 |
| Расширенная быстрая выборка данных из локального архива | 353 |
| Принудительная запись в SIAD | 354 |
| Дифференциальный срез локального архива | 355 |
| Запись аргументов канала CALL в SIAD | 357 |
| Архивирование в регистратор | 358 |
| Индивидуальный архив | 362 |
| Запрос удаленного индивидуального архива | 364 |
| Дополнительные средства | 368 |
| Перенаправление архива в базу данных | 368 |
| Глава 10 | 371 |
| Генерация документов | 371 |
| Использование разработанных шаблонов | 372 |
| Редактирование шаблонов документов | 374 |
| Редактор шаблонов документов (отчетов) | 374 |
| Задание свойств документа | 376 |
| Форматирование текста | 377 |

| | |
|---|------------|
| Форматирование списков | 377 |
| Использование таблиц в шаблоне документа | 379 |
| Конфигурирование обычной таблицы..... | 379 |
| Конфигурирование таблицы архивных значений | 381 |
| Вставка объектов в шаблон документа | 386 |
| Вставка значения переменной..... | 386 |
| Вставка горизонтальной линии | 386 |
| Вставка рисунка | 387 |
| Вставка выражения времени | 387 |
| Вставка тренда..... | 388 |
| Вставка круговой диаграммы..... | 391 |
| Вставка гистограммы | 392 |
| Вставка отчета тревог..... | 392 |
| Вставка документа..... | 396 |
| Особенности взаимодействия MPB с документом/экраном | 398 |
| Особенности извлечения данных монитором по запросу документа/экрана..... | 398 |
| Особенности отображения данных в документе/графике..... | 399 |
| Использование встроенных шаблонов | 404 |
| Глава 11..... | 405 |
| Разработка драйверов. Интерфейс TCOM..... | 405 |
| Драйверы обмена с контроллерами..... | 406 |
| Драйвер t13..... | 407 |
| Драйверы t11 и t12..... | 411 |
| Удаленный адрес и разновидности драйверов | 412 |
| TCOM5. Драйвер t11 | 415 |
| TCOM5. Функции драйвера t11, вызываемые MPB | 415 |
| TCOM5. Заголовок драйвера t11 | 419 |
| TCOM5. Примеры драйверов t11 | 421 |
| TCOM6. Драйвер t11 | 425 |
| TCOM6. Функции драйвера t11, вызываемые MPB | 425 |
| TCOM6. Заголовок драйвера t11 | 427 |
| Алгоритм взаимодействия с драйвером t11 | 430 |
| Алгоритм вызова драйвера t11 | 430 |
| Алгоритм обработки данных <i>BLOCKDATA11</i> | 433 |
| Алгоритм обработки данных <i>DATA11</i> | 434 |
| Алгоритм формирования блоковых запросов | 434 |
| Ограничения драйвера t11 | 436 |
| TCOM5. Драйвер t12 | 437 |
| TCOM5. Функции модуля описания протокола | 437 |
| TCOM5. Функции модуля описания интерфейса..... | 440 |
| TCOM5. Пример драйвера t12..... | 442 |
| TCOM6. Драйвер t12 | 445 |
| TCOM6. Функции модуля описания интерфейса..... | 445 |
| TCOM6. Функции модуля описания протокола | 446 |
| TCOM6. Заголовок драйвера t12 | 449 |
| Алгоритм взаимодействия с драйвером t12 | 452 |
| Использование для разработки драйверов других компиляторов | 455 |
| Драйверы обмена с УСО в WINDOWS | 457 |
| Каналы для вызова драйвера RWH | 457 |

| | |
|--|------------|
| Функции драйвера | 457 |
| Первый вызов драйвера | 457 |
| Последний вызов драйвера | 458 |
| Инициализация аналоговых сигналов | 458 |
| Инициализация дискретных сигналов | 458 |
| Формирование аналоговых сигналов | 458 |
| Опрос аналоговых сигналов | 459 |
| Формирование дискретных сигналов | 459 |
| Опрос дискретных сигналов | 459 |
| Опрос атрибутов каналов | 460 |
| Формирование атрибутов каналов | 460 |
| Шаблон драйвера для WINDOWS | 461 |
| Глава 12 | 463 |
| T-FACTORY | 463 |
| Каналы T-FACTORY | 464 |
| Канал класса M-РЕСУРС | 464 |
| Редактор канала M-РЕСУРС | 464 |
| Атрибуты канала M-РЕСУРС | 466 |
| Канал CALL с типом вызова MRESOURCE_1 | 471 |
| Канал класса ЕДИНИЦА ОБОРУДОВАНИЯ | 472 |
| Редактор канала ЕДИНИЦА ОБОРУДОВАНИЯ | 472 |
| Атрибуты канала ЕДИНИЦА ОБОРУДОВАНИЯ | 477 |
| Канал класса ПЕРСОНАЛ | 484 |
| Редактор канала ПЕРСОНАЛ | 484 |
| Атрибуты канала ПЕРСОНАЛ | 487 |
| Канал класса D-РЕСУРС | 491 |
| Редактор канала D-РЕСУРС | 491 |
| Атрибуты канала D-РЕСУРС | 494 |
| Глава 13 | 503 |
| TRACE MODE 6 в решении отраслевых задач | 503 |
| Электроэнергетика | 504 |
| АСКУЭ | 504 |
| Отчеты АСКУЭ | 505 |
| Приложения | 509 |
| Используемые сокращения | 510 |
| Задание параметров работы мониторов | 511 |
| Типовые средства редактирования | 535 |
| Типовые инструменты редактирования | 535 |
| Типовые операции редактирования | 537 |
| Сочетания клавиш в ИС | 539 |
| Форматы | 541 |
| Формат Си вывода чисел | 541 |
| Формат Си вывода даты и времени | 541 |
| Формат IP-адреса | 542 |
| Функции общего назначения | 544 |
| Копирование архивов и отчета тревог | 544 |
| Номер SubNum | 544 |
| Подтипы каналов | 552 |

| | |
|--|------------|
| Подтип 0 | 552 |
| Подтип 1 | 552 |
| Подтип 2 | 556 |
| Подтип 3 | 557 |
| Подтип 8 | 558 |
| Подтипы 9 и 100 | 565 |
| Подтип 10 | 566 |
| Подтипы 11 и 12 | 566 |
| Подтип 13 | 566 |
| Подтип 14 | 566 |
| Подтип 15 | 566 |
| Подтипы 22-26 | 567 |
| Подтип 64 | 567 |
| Подтип 65 | 567 |
| Подтип 66 | 568 |
| Подтип 67 | 568 |
| Подтип 69 | 568 |
| Подтип 70 | 569 |
| Подтип 71 | 569 |
| Подтипы 102 и 103 | 569 |
| Подтип 108 | 569 |
| Подтип 109 | 569 |
| Подтип 254 | 569 |
| Библиотеки компонентов | 570 |
| Поставляемая пользовательская библиотека компонентов | 570 |
| Объединение пользовательских библиотек компонентов | 570 |

Глава 7

Программирование алгоритмов

Программирование алгоритмов в TRACE MODE 6

Для программирования алгоритмов функционирования разрабатываемого проекта АСУ в TRACE MODE 6 включены языки **Техно ST**, **Техно SFC**, **Техно FBD**, **Техно LD** и **Техно IL**. Данные языки являются модификациями языков **ST** (Structured Text), **SFC** (Sequential Function Chart), **FBD** (Function Block Diagram), **LD** (Ladder Diagram) и **IL** (Instruction List) стандарта IEC61131-3.

Программы и некоторые их компоненты (функции, шаги и переходы SFC и т.п.) могут быть разработаны на любом из встроенных языков в соответствующем редакторе, при этом языки для программы и ее компонентов выбираются независимо.

Для создания и редактирования свойств аргументов, переменных, функций и структурных типов программы, а также для использования в программе функций из внешних библиотек в интегрированную среду разработки проекта встроены специальные табличные редакторы.


TRACE MODE 6 имеет также средства для отладки программ.

Основным языком программирования TRACE MODE 6 является **Техно ST**. Программы, разработанные на языках **Техно LD**, **Техно SFC** и **Техно FBD**, перед компиляцией транслируются в **Техно ST**. **IL**-программы перед компиляцией частично транслируются в **ST**, частично – в ассемблер. Отсюда следует, например, что ключевые слова **Техно ST** являются таковыми и для всех других языков.

Операции с программами

Подключение программы к проекту

Использование программы возможно только после ее успешной компиляции. Для компиляции программы нужно выполнить одно из следующих действий:

- выполнить команду **Компилировать** из меню **Программа** (или нажать клавишу **F7** или нажать ЛК на иконке  панели инструментов отладчика) – по этой команде создается только код для отладки программы в ИС (см. **Отладка программ**). Отладочный код сохраняется в поддиректории, создаваемой в директории **%TRACE MODE 6 IDE% \ tmp**. Если компилятор обнаруживает ошибки, он выводит соответствующие сообщения в окне, которое в этом случае открывается автоматически (см. **Окно 'Сооб-**

щения'). Если компиляция прошла успешно, окно сообщений не открывается;

- выполнить экспорт проекта (см. **Сохранение проекта для запуска**) – по этой команде в папке узла, содержащего канал вызова программы (см. **Выполнение программы в реальном времени**), создается как отладочный, так и исполняемый код (см. **Файлы узла, создаваемые при экспорте**). При обнаружении ошибок в программе выводится сообщение о невозможности ее экспорта.

Выполнение программы в реальном времени

Для выполнения программы в реальном времени в узле должен быть создан канал класса CALL с типом вызова **Program**, настроенный на вызов шаблона программы.

Подобный канал CALL типа INPUT обрабатывается со своим периодом пересчета в соответствующем потоке (см. **Период пересчета канала**).

Подобный канал CALL типа OUTPUT обрабатывается при подаче 1 в его атрибут 39, **EXEC**. В частности, для обработки такого канала из графики можно использовать функцию управления **Выполнить** (см. **Функции управления ГЭ**).

В атрибут 87, **CS** канала CALL вызова программы записывается время (в миллисекундах) между вызовами программы (см. **Атрибуты каналов, отображаемые профайлером**). В атрибут 92, **I2** записывается код ошибки (см. **Коды диагностируемых ошибок**). Количество аргументов канала вызова программы ограничено; в случае 4-байтовых аргументов – 1024 (т.е. под все аргументы не может быть выделено памяти больше, чем 4к – см. **Определение переменных и констант**).

Если атрибут **Параметр** (34, **FPrint**) канала вызова программы (**MATH**) равен 255:

- если к некоторому аргументу **MATH** (**MATH.arg**) привязан атрибут (124, **ArgSize**) канала CALL (**call**):
 - если $\text{MATH.R} > 0$, в **MATH.arg** передается значение аргумента с номером $(\text{MATH.R} - 1)$ канала **call**;
 - если $\text{MATH.R} = 0$, в течение одного такта пересчета в **MATH.arg** последовательно передаются все аргументы канала **call**, и после каждой такой передачи программа обрабатывается;
- к аргументу **MATH** может быть привязан атрибут (124, **ArgSize**) канала CALL.ROOT (**root**), к аргументам которого привязаны каналы CALL (**call_i**). В этой конфигурации по алгоритму, описанному выше, обрабатываются аргументы канала **call**, привязанного к аргументу с номером **root.R** канала **root**;

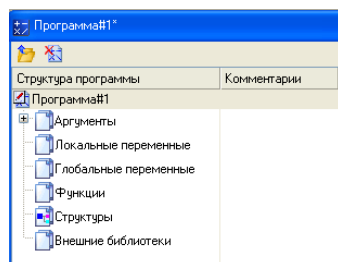
- если атрибут **MATH.124** привязан к каналу **call** (CALL.Screen, CALL.Document(Report), CALL.ChGroupReq или CALL.Vector), аргументы **MATH** копируются в аргументы **call**. Начальный аргумент для такой перепривязки задается полубайтом 1 (0x30) атрибута **Параметр** канала **call**.

Если к каналу **call** (CALL.Screen или CALL.Document(Report)) привязан атрибут **MATH.124**, передается столбец аргументов **MATH**, а если атрибут **MATH.140** – строка.



Окно структуры программы

В данном окне в виде дерева отображается структура программы. Чтобы открыть это окно, нужно дважды нажать ЛК на имени программы в навигаторе проекта.

Структурное дерево включает в себя все программные компоненты и используется для навигации по программе. При нажатии ЛК на любом элементе дерева автоматически открывается соответствующий редактор. Созданный в табличном редакторе компонент (элемент) автоматически добавляется к структурному дереву.

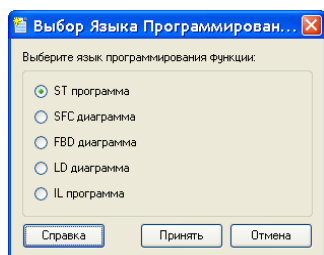


Окно структуры программы имеет панель следующих инструментов:


-  – переход на уровень определения программы, функции и т.п., к которой относится выделенный элемент;
-  – удаление тела выделенной программы, функции и т.п. (см. **Определение функции и функции-блока** в разделе **Пользовательские функции Техно ST**).

Выбор языка программирования

Язык программирования может быть независимо задан для основной программы, функции-блока, функции и шага SFC. Язык выбирается в следующем диалоге:



Этот диалог автоматически появляется на экране при нажатии ЛК на имени вновь созданной программы или ее компонента (для которого язык может быть задан независимо) в окне структуры программы. После выбора языка программа (компонент) открывается в соответствующем редакторе.

Изменить язык можно только после удаления тела программы (компонента) (см. **Определение функции и функции-блока** в разделе **Пользовательские функции Техно ST**). Для этого нужно нажать ЛК на иконке  панели инструментов в окне структуры программы, после чего диалог выбора языка автоматически появляется на экране.

Типовые инструменты редактирования программ

Все редакторы программ имеют наборы типовых инструментов для создания и удаления, поиска и работы с буфером обмена (см. **Типовые средства редактирования**). Состав этих наборов варьируется в зависимости от типа редактора.

Инструменты, специфичные для редакторов **SFC**, **FBD** и **LD**, рассматриваются в разделах, посвященных работе в этих редакторах.

Масштабирование диаграмм


Для изменения размеров диаграмм **SFC**, **FBD** и **LD** используются клавиши «←» и «→» дополнительной клавиатуры при удержании клавиши **CTRL**.


Максимально возможные размеры диаграмм определяются настройками соответствующих редакторов (см. **Настройка редакторов и отладчика**).


Инструменты для работы с закладками



Редакторы **ST**, **IL**, **FBD**, **SFC** и **LD** имеют панель следующих инструментов для работы с закладками:

 – установить/снять закладку (**CTRL+F2**);

 – перейти к предыдущей закладке (**SHIFT+F2**);

 – перейти к следующей закладке (**F2**);

 – удалить все закладки.

Закладки используются для быстрых переходов по программе. Чтобы установить закладку, нужно поместить курсор в нужную строку программы (в текстовом редакторе) или на нужный элемент диаграммы (в редакторе **SFC**, **FBD** или **LD**) и нажать кнопку . В тестовых редакторах закладки обозначаются значком , в редакторах **SFC**, **LD** и **FBD** элемент диаграммы с закладкой выделяется цветом (см. также **Отладка программ**).

Создание элементов программ с помощью табличных редакторов

Табличные редакторы используются для создания следующих компонентов и элементов программ:

- аргументы;
- локальные переменные;
- глобальные переменные;
- функции-блоки (подпрограммы) и функции;
- структурные типы.

Кроме того, с помощью табличных редакторов конфигурируются обращения к функциям из внешних библиотек.

Перечисленные компоненты и элементы, наряду с листингами **ST** и **IL** и диаграммами **LD**, **SFC** и **FBD**, образуют ветви дерева в окне структуры программы.

Для входа в соответствующий табличный редактор нужно в окне структуры программы нажать ЛК на любом из перечисленных выше элементов.

Особенности редактирования

Для создания/удаления строк и поиска в табличных редакторах используется типовая панель инструментов (см. **Типовые средства редактирования**).

Для перехода к редактированию отдельной ячейки таблицы нужно дважды нажать ЛК на этой ячейке. Редактирование ячейки производится либо путем непосредственного ввода с клавиатуры, либо путем выбора нужного значения из списка.

При задании числа в качестве разделителя целой и дробной части используется точка.

Если в ячейку столбца **[] Массив** ввести число, равное количеству элементов массива, то в этой ячейке отобразится диапазон индексов элементов (начиная с 0). Например, для двумерного массива при вводе **9, 8** отобразится **0 .. 8, 0 .. 7**.

Некоторые элементы (например, переменные), заданные в табличных редакторах, автоматически добавляются в листинги текстовых программ в виде соответствующих конструкций языка. Такие конструкции выделены серым цветом и недоступны для редактирования с помощью клавиатуры.

Доступные типы данных (столбец **Тип данных**) для программ на всех языках одинаковы (см. **Определение переменных и констант**).

Начальное значение (столбец **Начальное значение**) может быть задано в любой из форм, определенных для **Техно ST** (см. **Числовые константы Техно ST** и **Строковые константы Техно ST**).

Табличный редактор аргументов основной программы

Аргументы основной программы, так же как и аргументы любого шаблона, задаются в соответствующем табличном редакторе (см. **Табличный редактор аргументов**).

Табличный редактор аргументов программного компонента

Вид табличного редактора аргументов программного компонента (функции или функции-блока) показан на следующем рисунке.

| Имя | Тип ввода/выхода | Тип данных | Начальное значение | Комментарий |
|-----------|------------------|------------|--------------------|-------------|
| N ARG_000 | Вход | INT | 0 .. 8 | N array |
| N ARG_001 | Вход/Выход | INT | V 256 | |
| N ARG_002 | Выход | INT | | |
| | Вход | BOOL | | |
| | Вход | SINT | | |
| | Вход/Выход | USINT | | |
| | Вход/Выход | INT | | |
| | | UINT | | |
| | | DINT | | |
| | | UDINT | | |
| | | TIME | | |
| | | DATE | | |
| | | TIME_ | | |

В этом редакторе задается имя аргумента, его тип (**ВХОД**, **ВЫХОД** или **ВХОД/ВЫХОД**), тип данных, начальное значение и комментарий. Если в поле **[] (Границы массива)** строки аргумента задать число, аргумент интерпретируется как массив.

Заданные на рисунке параметры равнозначны следующей конструкции **Техно ST** (см. **Операторы Техно ST**):

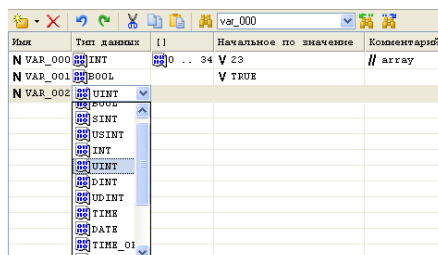
```

VAR_INPUT ARG_000: ARRAY OF INT[ 0 .. 8 ];
END_VAR //array
VAR_INOUT ARG_001 : INT := 256; END_VAR
VAR_OUTPUT ARG_002 : INT; END_VAR

```

Табличный редактор переменных

Вид табличного редактора переменных показан на следующем рисунке.



В этом редакторе задается имя переменной, ее тип данных, начальное значение и комментарий. Если в поле [] строки переменной задать число, переменная интерпретируется как массив.

Если показанные на рисунке переменные являются локальными, то заданные параметры равнозначны следующей конструкции **Техно ST** (см. **Операторы Техно ST**):

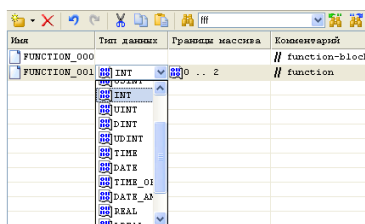
```

VAR VAR_000: ARRAY OF INT[ 0 .. 34 ] := 23;
END_VAR //array
VAR VAR_001: BOOL := TRUE; END_VAR
VAR VAR_002: UINT; END_VAR

```

Табличный редактор функций и функций-блоков

Вид табличного редактора функций и функций-блоков показан на следующем рисунке.



В этом редакторе задается имя функции (функции-блока) и комментарий.

Если указан тип возвращаемого значения, определяется функция, если тип возвращаемого значения не указан, определяется функция-блок.

Если в поле **Массив** строки функции задать число, функция возвращает

массив. Для функции-блока поле **Массив** недоступно.

Показанные на рисунке параметры соответствуют следующим конструкциям **Техно ST** (см. **Пользовательские функции Техно ST**):

строка 1:

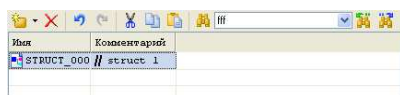
```
FUNCTION_BLOCK FUNCTION_001
END_FUNCTION_BLOCK
```

строка 2:

```
FUNCTION FUNCTION_002 : ARRAY OF INT[ 0 .. 2 ]
END_FUNCTION
```

Табличный редактор структурных типов

В этом редакторе задается имя создаваемого структурного типа и комментарий (см. **Структуры Техно ST**).

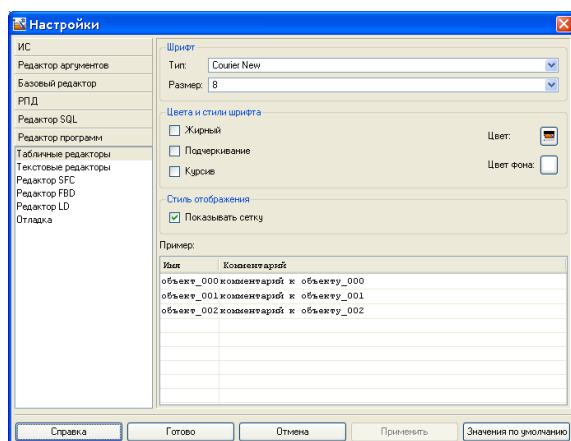


Табличные редакторы для работы с внешними функциями

Для конфигурирования вызовов функций из внешних библиотек используются специальные табличные редакторы (см. **Внешние библиотеки функций**).

Настройка редакторов и отладчика

Параметры редакторов и отладчика программ настраиваются на вкладке **Редактор программ** (см. **Задание общих настроек ИС**):





При выборе редактора в левой части диалога набор инструментов вкладки соответствующим образом изменяется.

Для всех редакторов имеются типовые инструменты для задания таких стандартных характеристик шрифта, как семейство, размер, вид начертания (жирный, курсив, подчеркнутый) и цвет. Типовым является также задание цвета фона рабочего поля редактора.

При изменении размеров шрифта изменяются размеры элементов SFC-, FBD- и LD-диаграмм, а также шаг сетки.

Выбор цвета производится в стандартном диалоге, который выводится на

экран при нажатии кнопок  (цвет выбранного элемента программы) и  (цвет фона) диалога настройки редактора.

Вкладка содержит кнопку восстановления параметров, заданных по умолчанию.

Настройка табличных редакторов

В диалоге настройки параметров табличных редакторов с помощью типовых инструментов задаются параметры шрифта и цвет фона рабочего поля. Диалог снабжен окном предварительного просмотра.

Настройка текстовых редакторов

В диалоге настройки параметров текстовых редакторов с помощью типовых инструментов задается цвет фона рабочего поля, а также параметры шрифта для каждого из следующих элементов текстовой программы: обычный текст, комментарий, число, строковое значение, тип данных, ключевое слово и метка. Перед заданием параметров шрифта нужный элемент нужно выделить в списке нажатием ЛК.

Настройка SFC-редактора

В диалоге настройки параметров SFC-редактора с помощью типовых инструментов задается цвет фона рабочего поля, а также такие параметры отображения SFC-шагов и SFC-переходов (в том числе при выделении), как цвет контура, цвет заполнения, цвет и другие характеристики шрифта. Перед заданием параметров элемент нужно выделить в списке нажатием ЛК.

В диалоге можно также задать толщину линий переходов и шаг размещения SFC-шагов и SFC-переходов на диаграмме.

Если флаг **Плоский вид** не установлен, SFC-шаги и SFC-переходы имеют объемный вид:



При установке флага **Плоский вид** SFC-шаги и SFC-переходы выглядят следующим образом:

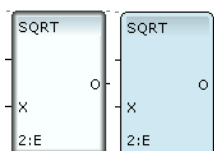


Настройка FBD-редактора

В диалоге настройки параметров FBD-редактора с помощью типовых инструментов задается цвет фона рабочего поля, а также такие параметры отображения элементов FBD-диаграммы (в том числе при выделении), как цвет контура, цвет заполнения, цвет и другие характеристики шрифта (для FBD-блоков) и цвет линий (для входов/выходов блоков и межблочных связей). Перед заданием параметров элемент нужно выделить в списке нажатием ЛК.

В диалоге можно также задать толщину линий и шаг размещения FBD-блоков на диаграмме (шаг сетки).

Если флаг **Плоский вид** не установлен, FBD-блоки имеют объемный вид, в противном случае – плоский:



При установке флага **Показать сетку** сетка отображается в рабочем поле редактора.

Настройка LD-редактора

В диалоге настройки параметров LD-редактора с помощью типовых инструментов задается цвет фона рабочего поля, а также такие параметры отображения элементов LD-диаграммы (в том числе при выделении), как цвет блоков, характеристики шрифта и цвет шин и связей. Перед заданием параметров элемент нужно выделить в списке нажатием ЛК.

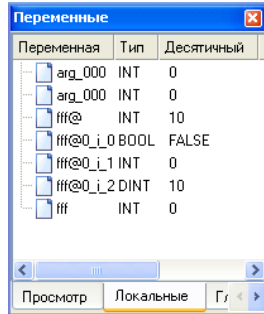
При установке флага **Показать сетку** сетка отображается в рабочем поле редактора.

В диалоге можно также задать толщину линий и шаг размещения LD-блоков на диаграмме (шаг сетки).

Настройка отладчика

В диалоге настройки отладчика могут быть заданы следующие параметры:

- **Глубина стека** – задание глубины стека вызовов функций в программе;
- **Показать внутренние переменные** – при установке этого флага в окне просмотра переменных дополнительно выводятся текущие значения функций, их аргументов и переменных:



- **Отладка в цикле** – при установке этого флага происходит за-цикливание процесса отладки. В пошаговом режиме после выполнения всей программы отладчик переходит к первой команде программы. В непрерывном режиме после выполнения всей программы отладчик приостанавливает свою работу; в этот момент можно перейти в другой режим отладки. Эта функция имеет дополнительную опцию – **Задержка между циклами** (в секундах).

Описание языка Техно ST

Лексическая структура языка Техно ST

В алфавит языка входят:

- прописные и строчные буквы латинского алфавита;
- цифры 0,1,...9;
- специальные знаки:
+ - * / < = > ! : & | ^ ~ % () [] , ; #

Из символов алфавита формируются лексемы языка:

- идентификаторы;
- ключевые слова;
- числовые и строковые константы;
- символьные операторы (знаки операций);
- разделители;
- комментарии.

Идентификаторы Техно ST

Идентификаторы могут состоять из заглавных и строчных латинских букв, цифр и знака подчеркивания '_'. Первым символом идентификатора не может быть цифра. Длина идентификатора не ограничена.

Идентификаторы не чувствительны к регистру, т.е. 'AAA' и 'aaa' являются идентичными идентификаторами.

Имена констант, переменных, функций и т.п., задаваемые пользователем, должны удовлетворять правилам задания идентификаторов.

Ключевые слова Техно ST

Ключевые (служебные) слова – это идентификаторы, зарезервированные в языке для специального использования.

Список ключевых слов языка **Техно ST**:

and, array, bool, break, by, byte, case, constant, continue, date, date_and_time, dint, do, dt, dword, else, elsif, end_case, end_for, end_function, end_function_block, end_if, end_program, end_repeat, end_struct, end_type, end_var, end_while, exit, false, for, function, function_block, goto, handle, if, int, lreal, mod, not, of, or, program, real, repeat, return, rol, ror, shl, shr, sint, string,

struct, time, time_of_day, to, tod, true, type, uint, until, usint, var, var_arg, var_global, var_inout, var_input, var_output, while, word, xor.

Кроме того, к ключевым словам относятся имена функций C, которые могут быть использованы в ST-программе (см. **Стандартные функции C в ST-программе**).

Ключевые слова нечувствительны к регистру, т.е. **while** и **WHILE** являются идентичными ключевыми словами.

Ключевые слова **Техно ST** являются таковыми и для всех других языков; их нельзя использовать в качестве пользовательских идентификаторов (например, в качестве имен переменных) в любых программах.

Разделители Техно ST

В качестве разделителей, или знаков пунктуации, в языке **Техно ST** используются следующие лексемы:

+ - * ** / < <= <> << > >= >> ! != =
 == : := & | ^ ~ % () [] . .. , ;

Комментарии Техно ST

Комментарии бывают двух видов: строчные и блочные.

Строчный комментарий начинается с символов **//** и продолжается до конца строки.

Блочный комментарий начинается с символов **/*** и продолжается до символов ***/**.

Вложенные комментарии не допускаются.

Синтаксис Техно ST

Для описания структуры программы и операторов в **Техно ST** приняты следующие терминологические соглашения:

- **выражение** – последовательность операндов, разделителей и символьных операторов, задающая вычисление без присвоения результата;
- **предложение** – последовательность лексем, определяющая выполнение логически законченного промежуточного действия. Таким действием может быть присвоение переменной результата вычислений, вызов функции-блока и т.п. Операторы (кроме символьных) также образуют предложения.

На основании этих соглашений программа или ее компонент на языке **Техно ST** определяется как последовательность предложений.

Каждое предложение должно завершаться точкой с запятой. Исключением из этого правила являются операторы определения переменных, для завершения которых точка с запятой не используется.

Длина строки программы не ограничивается, лексемы разделяются произвольным числом пробелов, знаков табуляции или символов перевода строки.

Основная точка входа в программу

Основная точка входа в программу определяется следующей конструкцией:

```
program
    {определение аргументов}
    {список предложений}
end_program
```

Необязательное выражение {определение аргументов} задается аналогично выражению {определение переменной} для операторов определения переменной (см. раздел **Операторы Техно ST**). В дальнейшем конструкция **program...end_program** называется основной программой.

Функции, глобальные переменные и структурные типы не могут быть определены в основной программе.

Основная точка входа создается автоматически при создании программы. Если для программы выбран язык **ST** или **IL**, конструкция **program ... end_program** отображается в листинге. Если для программы выбран язык **SFC**, **LD** или **FBD**, основная точка создается во внутреннем представлении и недоступна для просмотра.

Переменные и константы Техно ST

Под объектом в **Техно ST** понимается некоторая область памяти, которой присвоено имя (идентификатор). Переменная (константа) – это частный случай объекта как именованной области памяти. Отличительной чертой переменной (константы) является возможность связывать с ее именем различные значения, совокупность которых определяется типом переменной (константы).

При определении значения переменной в соответствующую ей область памяти помещается некоторый код. Если этот процесс происходит во время компиляции программы, он называется инициализацией переменной, если во время выполнения программы – присвоением значения.

Определение переменных и констант

Вид константы или переменной (глобальная, локальная) задается оператором, с помощью которого данная переменная (константа) определяется (см. **Операторы определения переменных** в разделе **Операторы Техно ST**). Синтаксис операторов определения переменных предполагает обязательное указание типа данных:

```
//определение локальной строковой
//переменной myVar
VAR myVar: STRING; END_VAR
```

Тип данных определяет размер выделяемой памяти. Для указания типа в **Техно ST** определены следующие ключевые слова (в круглых скобках указано соответствие типу данных C):

BOOL (bool) – булево значение размерностью 1 байт (**true (1)** или **false (0)**);

SINT (__int8) – целое со знаком размерностью 1 байт (**-128 ... 127**);

USINT (unsigned __int8) – целое без знака размерностью 1 байт (**0 ... 255**);

INT (short) – целое со знаком размерностью 2 байта (**-32768 ... 32767**);

UINT (unsigned short) – целое без знака размерностью 2 байта (**0 ... 65535**);

DINT (long) – целое со знаком (4 байта) (**-2147483648 ... 2147483647**);

UDINT (unsigned long) – целое без знака (4 байта) (**0 ... 4294967295**);

TIME, DATE, TIME_OF_DAY(*), DATE_AND_TIME – соответствуют **DINT**. Значения переменных этих типов задаются аналогично соответствующим временным константам (см. **Числовые константы Техно ST**);

(*) Начиная с версии 6.08, тип данных **TIME_OF_DAY** не поддерживается.

REAL (float) – вещественное число (4 байта) (максимальное значение **3.402823466e+38**);

LREAL (double) – вещественное число (8 байт) (максимальное значение **1.7976931348623158e+308**);

STRING (WCHAR*) – 256 символов в кодировке utf-8 (см. также **Строковые константы Техно ST**);

HANDLE – см. **Специальная обработка аргументов HANDLE**.

Кроме указанных типов, переменной может быть присвоен структурный тип, созданный пользователем. Такая переменная является конкретным объектом указанного типа (см. **Структуры Техно ST**).

При определении переменной может быть задано ее значение:

```
VAR i: INT:=0; END_VAR
```

Если при определении переменной ее значение не задано, то этой переменной по умолчанию присваивается следующее начальное значение:

числовая переменная – 0;

переменная типа **BOOL** – **FALSE**;

переменная типа **STRING** – пустая строка;

переменная типа **HANDLE** – 16#00000000 (0 в формате HEX);

переменная типа **TIME**, **DATE** или **DATE_AND_TIME** – 0.

При определении константы задание ее значения обязательно:

```
VAR CONSTANT myConst: INT:=13; END_VAR
```

В отличие от переменной, значение константы в программе изменять нельзя.

Особенности присвоения значений переменным

При присвоении значения переменной типа TYPE1 переменной типа TYPE2 нужно учитывать следующее:

- присвоение корректно только в том случае, если тип TYPE2 включает в себе все числа типа TYPE1:

```
VAR a: REAL := -1564.343; END_VAR
```

```
VAR b: USINT := 50; END_VAR
```

```
  a = b; //корректная операция
```

```
  b = a; //некорректная операция
```

- присвоение корректно, если один из типов – **BOOL**, а другой – любой численный. Логическое значение TRUE соответствует единице, FALSE – нулю; нуль соответствует FALSE, любое ненулевое значение, в том числе отрицательное, соответствует TRUE:

```
VAR a: BOOL; END_VAR
```

```
VAR b: SINT := -50; END_VAR
```

```
  a = b; //a = TRUE, корректная операция
```

```
  b = a; //b = 1, корректная операция
```

Числовые константы Техно ST

Десятичные целочисленные константы состоят из ненулевой цифры, за которой следует последовательность десятичных цифр:

123, 456, 7890

Двоичные целочисленные константы начинаются с префикса **2#**, за которым следуют цифры 0 или 1:

2#1001, 2#1100

Восьмеричные целочисленные константы начинаются с префикса **8#**, за которым следуют цифры от 0 до 7:

8#777, 8#0123

Шестнадцатеричные константы начинаются с префикса **16#**, за которым следуют цифры или буквы **a...f**. Буквы можно задавать как в нижнем, так и в верхнем регистре (**A...F**):

16#123, 16#EA7

Вещественные константы состоят из целой и дробной части, разделенной точкой. Либо целая, либо дробная часть может отсутствовать. Числа могут задаваться в формате с плавающей точкой, при этом они сопровождаются суффиксом E с указанием десятичного порядка:

1.23, 123., .123, 0.123E3, .123e-3, 123.E+5

Временные интервалы состоят из префикса **t#** или **time#**, за которым следует запись в виде

<дни>d<часы>h<минуты>m<секунды>s<миллисекунды>ms

Любая составляющая может быть опущена (например, запись **t#1h10s** является корректной и означает 1 час 10 секунд). Временной интервал приводится к целочисленному виду, означающему количество миллисекунд в заданном временном интервале, а возвращается целое число секунд.

Дата состоит из префикса **d#** или **date#**, за которым следует запись в виде **yyyy-mm-dd** (год, месяц, день). Приводится к целочисленному виду, означающему количество секунд, прошедшее с 0 часов 1 января 1970 года до 0 часов заданной даты.

Константа "Дата и время" состоит из префикса **dt#** или **date_and_time#**, за которым следует запись в виде **yyyy-mm-dd-hh:mm:ss** (год, месяц, день, час, минута, секунда). Приводится к целочисленному виду, означающему количество секунд, прошедшее с 0 часов 1 января 1970 года до данных даты и времени.

Строковые константы Техно ST

Строковые константы представляют собой набор символов, заключенных

в одинарные или двойные кавычки: **'первая строка'**, **"вторая строка"**. В строке недопустимы управляющие символы, включая переводы строки, а также кавычки и символ **\$**.

Для размещения в строках произвольных символов применяется механизм эскейп-последовательностей, начинающихся с символа **\$**. Определены следующие последовательности:

- \$r** – возврат каретки, код 16#0D;
- \$n** – перевод строки, код 16#0A;
- \$t** – табуляция, код 16#09;
- \$uXXXX** – UNICODE-символ ('X' – шестнадцатеричная цифра);
- \$x** – символ **x** ('x' – любой символ).

Пример

"Строка с кавычкой: **\$'**, символом **\$u0410** и переводом строки **\$n**"

Особенности вычислений

Целочисленность результата арифметических вычислений в программе имеет высший приоритет – даже в том случае, когда этот результат присваивается переменной с плавающей точкой.

Пусть, например, в программе объявлена переменная **float**:

```
VAR VAR_000 : REAL; END_VAR
```

Тогда:

```
VAR_000 = 2 / 10           //VAR_000 = 0
VAR_000 = 2. / 10         //VAR_000 = 0.2
VAR_000 = 2. / 10 + 2 /10 //VAR_000 = 0.2
```

Операторы языка Техно ST

Символьные операторы

Символьные операторы **Техно ST** представляют собой знаки операций, выполняемых над операндами. В качестве операндов могут выступать:

- имена констант;
- имена переменных;
- имена переменных – элементов массивов;
- вызовы пользовательских функций;
- вызовы библиотечных функций;

- выражения, заключенные в скобки;
- уточненные имена элементов структур.

Арифметические операторы

К данным операторам относятся:

унарный '-' – смена знака операнда;

унарный '+' – пустая операция;

'+' – сложение; если один из операндов - строка, то другой операнд также преобразуется в строку, затем производится конкатенация строк;

'-' – вычитание;

'*' – умножение (только для числовых значений);

'/' – деление;

'%' или **mod** – взятие остатка (деление по модулю);

** – возведение в степень.

Побитовые операторы

В данную группу входят следующие операторы:

'&' – побитовое "И";

'|' – побитовое "ИЛИ";

'^' или **xor** – побитовое "исключающее ИЛИ".

унарная '~' – поразрядная инверсия;

'<<' или **shl** – сдвиг влево на указанное число разрядов (только для целочисленных значений);

'>>' или **shr** – сдвиг вправо на указанное число разрядов (без сохранения знака) (только для целочисленных значений);

rol – циклический сдвиг влево на указанное число разрядов (только для целочисленных значений);

ror – циклический сдвиг вправо на указанное число разрядов (без сохранения знака) (только для целочисленных значений).

Операторы сравнения

Все операторы сравнения возвращают **true**, если условие выполняется, иначе – **false**.

'==' – проверка на равенство;
'!=' или '<>' – проверка на неравенство;
'<' – меньше;
'>' – больше;
'<=' – меньше или равно;
'>=' – больше или равно.

Логические операторы

К данным операторам относятся:

&& или **and** – возвращает **true**, если оба операнда не равны нулю, иначе - **false**. Если первый операнд ложен, второй операнд не вычисляется;

|| или **or** – возвращает **true**, если любой из операндов не равен нулю, иначе - **false**. Если первый операнд истинен, второй операнд не вычисляется.

унарный **!** или **not** – возвращает **true**, если операнд ложен, иначе - **false**.

Операторы присваивания

Синтаксис:

{операнд} = {выражение}

или

{операнд} := {выражение}

В качестве операнда может выступать:

- имя переменной;
- имя переменной-элемента массива (индексированная переменная);
- уточненное имя переменной объекта.

Приоритет символьных операторов Техно ST

В приведенном ниже списке символьные операторы расположены в порядке убывания приоритета:

!, not, ~, унарный '-', унарный '+'
<<, >>, shl, shr, rol, ror
&, |, ^, xor

`*`, `/`, `%`, `mod`
`+`, `-`
`==`, `!=`, `<>`, `<`, `<=`, `>`, `>=`
`&&`, `and`
`||`, `or`
`=`, `:=`

Выражения, заключенные в скобки, вычисляются в первоочередном порядке.

Операторы Техно ST

Определены следующие операторы, образующие предложения **Техно ST**:

`return`
`if`
`case`
`while`
`repeat`
`for`
`break`
`exit`
`continue`
операторы определения переменных;
оператор индексирования элементов массива;
`goto`

Оператор return

Определены 2 варианта задания данного оператора:

`return {выражение}`

Действие: выход из функции **Техно ST**. Значением функции является значение {выражения};

`return`

Действие: выход из функции-блока **Техно ST**.

Пример

```
RETURN (2 + ARG_000 ** 2);
```

Оператор if-then-else

Данный оператор начинается с ключевого слова **if** и заканчивается ключевым словом **end_if**. Определены 3 варианта задания данного оператора:

Вариант 1

```
if {выражение} then {последовательность предложений} end_if
```

Действие: если {выражение} истинно, выполняется {последовательность предложений}, иначе никаких действий не производится.

Вариант 2

```
if {выражение} then {последовательность предложений1}  
else {последовательность предложений2} end_if
```

Действие: если {выражение1} истинно, выполняется {последовательность предложений1}, иначе выполняется {последовательность предложений2}.

Вариант 3

```
if {выражение1} then {последовательность предложений1}  
elseif {выражение2} then {последовательность предложений2}  
...  
elseif {выражениеN} then {последовательность предложенийN}  
else {последовательность предложений} end_if
```

Действие: выполняется первая по порядку {последовательность предложений}, для которой соответствующее {выражение} истинно. Если все {выражения} ложны, выполняется {последовательность предложений}, следующая за ключевым словом **else**.

Количество блоков "**elseif** {выражение} **then** {последовательность предложений}" не ограничено.

Пример

В результате выполнения следующего кода переменной VAR_000 присваивается значение 200. Выполняется только одно (первое по порядку) действие, для которого условие истинно, поэтому действие, следующее за конструкцией **ELSIF...THEN**, выполнено не будет, несмотря на то, что условие VAR_002 < 1 истинно:

```
VAR VAR_000 : INT; END_VAR
VAR VAR_002 : REAL := 0.5; END_VAR
  IF VAR_002 < 2 THEN
    VAR_000 = 200;
  ELSIF VAR_002 < 1 THEN
    VAR_000 = 500;
  ELSE
    VAR_000 = 300;
  END_IF;
```

Оператор case

Определены 2 варианта задания данного оператора.

Вариант 1

```
case {выражение} of
  {список значений}: {последовательность предложений}
  ...
  {список значений}: {последовательность предложений}
end_case
```

Вариант 2

```
case {выражение} of
  {список значений}: {последовательность предложений}
  ...
  {список значений}: {последовательность предложений}
else {последовательность предложений}
end_case
```

Список значений представляет собой набор целых чисел или набор диапазонов целых чисел, разделенных запятой. Диапазон задается в виде

{нижняя граница} .. {верхняя граница}

Действие: если результат вычисления {выражения} принадлежит множествам, заданным {списками значений}, выполняется соответствующая {последовательность предложений}. Если результат вычисления {выражения} не принадлежит ни одному из заданных множеств, выполняется {последовательность предложений}, следующая за ключевым словом **else**.

Пример

В результате выполнения следующего кода VAR_001=500:

```
VAR VAR_000 : INT; END_VAR
VAR VAR_001 : INT; END_VAR
CASE VAR_000 + 4 OF
  0 .. 2 : VAR_001 = 200;
  3, 4, 5 : VAR_001 = 500;
END_CASE;
```

Оператор while

Синтаксис:

```
while {выражение} do {последовательность предложений} end_while
```

Действие: пока {выражение} истинно, выполняется {последовательность предложений}.

Пример

После выполнения следующего кода VAR_001=16:

```
VAR VAR_000 : INT := 10; END_VAR
VAR VAR_001 : INT; END_VAR
WHILE VAR_000 > 2 DO VAR_000 = VAR_000
- 1;
  VAR_001 = VAR_001 + 2;
END_WHILE;
```

Оператор repeat

Синтаксис:

```
repeat {последовательность предложений} until {выражение} end_repeat
```

Действие: пока {выражение} истинно, выполняется {последовательность предложений}. Если {выражение} ложно, {последовательность предложений} выполняется 1 раз.

Пример

После выполнения следующего кода VAR_001=20:

```
VAR VAR_000 : INT :=10; END_VAR  
VAR VAR_001 : INT; END_VAR  
REPEAT VAR_001 = VAR_001 + 2; VAR_000 =  
VAR_000 + 1; UNTIL VAR_000 < 20 END_REPEAT;
```

Оператор for

Синтаксис:

```
for {инициализация переменной цикла} to {выраже-  
ние1} by {выражение2} do {последовательность пред-  
ложений}  
  
end_for
```

Инициализация переменной цикла имеет вид:

```
{имя переменной} := {выражение}
```

Действие: пока значение переменной цикла меньше или равно значению {выражения1} выполняется {последовательность предложений}. По завершении каждого цикла к переменной цикла прибавляется значение {выражения2}; если оно не задано, прибавляется 1.

С помощью оператора **for** нельзя построить цикл с убывающим счетчиком. Для создания таких циклов нужно использовать операторы **while** и **repeat**.

Пример

После выполнения следующего кода VAR_001=22:

```
VAR VAR_000 : INT :=10; END_VAR  
VAR VAR_001 : INT; END_VAR  
FOR VAR_000 = 10 TO 20 DO VAR_001 = VAR_001  
+ 2; END_FOR;
```

Операторы break и exit

Операторы **break** и **exit** эквивалентны.

Синтаксис:

```
break  
  
exit
```

Действие: выход за пределы цикла. В случае вложенных циклов выход

осуществляется только из текущего цикла и не затрагивает внешние.

Оператор `continue`

Синтаксис:

```
continue
```

Действие: переход в конец цикла, т.е. выражения, следующие за оператором **continue** до конца цикла, не выполняются.

Операторы определения переменных

Операторы определения переменных могут быть заданы вручную (кроме операторов определения глобальных переменных) или с помощью табличного редактора.

В языке **ST** определены следующие операторы данного типа:

```
var
```

```
    {определение переменной}
```

```
    ...
```

```
    {определение переменной}
```

```
end_var
```

```
var_global
```

```
    {определение переменной}
```

```
    ...
```

```
    {определение переменной}
```

```
end_var
```

```
var_arg
```

```
    {определение переменной}
```

```
    ...
```

```
    {определение переменной}
```

```
end_var
```

```
var_input
```

```
    {определение переменной}
```

```
...
    {определение переменной}
end_var

var_output
    {определение переменной}
...
    {определение переменной}
end_var

var_inout
    {определение переменной}
...
    {определение переменной}
end_var
```

После ключевого слова **end_var** точка с запятой не ставится.

Действие: определяет новую переменную. При использовании совместно с **constant** задает константу.

Оператор **var ... end_var** используется для создания локальных переменных и структур; может использоваться в основной программе или ее компоненте (функции).

Оператор **var_global ... end_var** используется для создания глобальных переменных; может использоваться вне основной программы и ее компонентов (функций).

Оператор **var_arg(var_input) ... end_var** используется для определения аргументов (основной программы или ее функций), передаваемых по значению. Определение аргумента с помощью этого оператора равнозначно заданию аргумента типа **вход** в табличном редакторе.

Оператор **var_output(var_inout) ... end_var** используется для определения аргументов (основной программы или ее функций), передаваемых по ссылке. Определение аргумента с помощью оператора **var_output...end_var** равнозначно заданию в табличном редакторе аргумента типа **выход**, а определение аргумента с помощью оператора **var_inout...end_var** равнозначно заданию аргумента типа **вход/выход**.

Создание аргументов вручную с помощью указанных операторов может использоваться только в отладочных програм-

мах – для таких аргументов нельзя задать привязку. Аргументы рабочей программы следует создавать с помощью табличного редактора.

Выражение **{определение переменной}** имеет вид:

{имя переменной} : {тип переменной} ;

{имя переменной} : {тип переменной} := {выражение} ;

{имя переменной} : array [] of {тип переменной} ;

{имя переменной} : array [{размерности массива}] of {тип переменной} ;

{имя переменной} : array [{размерности массива}] of {тип переменной} := {начальные значения} ;

Выражения **{размерности массива}** задаются в виде диапазонов изменения индексов массива, разделенных запятой.

Диапазон изменения индексов массива имеет вид

{нижняя граница} .. {верхняя граница}

или

{размер массива}

обозначающий диапазон от **0** до **{размер массива}-1**. В случае, если размерность массива не указана, он считается пустым и ожидается его инициализация в ходе выполнения программы.

Выражения **{начальные значения}** имеют вид списка начальных значений элементов массива, разделенных запятой. Каждое начальное значение имеет вид выражения, вычисление которого дает реальное начальное значение, или конструкции

{целочисленная константа} ({выражение})

где **{целочисленная константа}** задает количество элементов, которым присваивается это значение. При присвоении начальных значений элементам массива первым изменяется последний индекс массива.

Область действия имени переменной определяется по следующим правилам:

- **глобальные** переменные действуют в рамках программы и сохраняют свое значение между вызовами программы. В частности, глобальными являются переменные FBD- и LD-блоков;
- **локальные** переменные и **аргументы** действуют в рамках объекта (программы, функции, структуры), в котором определены.

При привязке переменной по имени она ищется в следующем порядке: локальные, аргументы функции, переменные-члены структуры, глобаль-

ные.

Оператор индексирования элементов массива

Синтаксис:

```
{имя} [ {индекс 1}, ... {индекс N}]
```

где **{имя}** – имя переменной или функции, возвращающей массив, а **{индекс k}** – целое неотрицательное число или целочисленная переменная (кроме **UINT** и **USINT**). Количество индексов зависит от размерности массива.

Действие: возвращает ссылку на элемент массива, которая может быть использована в левой и правой части оператора присваивания.

Оператор goto

Синтаксис:

```
goto {метка строки}
```

Действие: безусловный переход к строке кода с указанной меткой. Оператор **goto** и метка, на которую этот оператор ссылается, должны находиться в одном и том же программном компоненте (программе, функции и т.п.). Метка должна начинаться с буквы и отделяться от кода программы двоеточием:

```
...
goto myLabel2;
...
myLabel2:
    END_PROGRAM
```

Функции Техно ST

Стандартные функции C в ST-программе

В ST-программе может быть использован ряд стандартных функций Си:

```
sin(), cos(), tan(), asin(), exp(), log()
```

Специальные функции в ST-программе

В ST-программе могут быть использованы следующие специальные функции:

```
//чтение байта из порта с номером port_num
```

```
    USINT inp(UINT port_num)
//чтение слова из порта с номером port_num
    UINT inpw(UINT port_num)
//запись байта в порт с номером port_num
    outp(UINT port_num, USINT value)
//запись слова в порт с номером port_num
    outpw(UINT port_num, UINT value)
//чтение атрибута канала (целое со знаком,
//4 байта)
    DINT getAttributeI(UDINT ch_id, UINT
    attr_id)
//чтение атрибута канала (вещественное, 4 байта)
    REAL getAttributeF(UDINT ch_id, UINT
    attr_id)
//установка атрибута канала (целое со знаком,
//4 байта)
    setAttributeI(UDINT ch_id, UINT attr_id,
    DINT value)
//установка атрибута канала (вещественное,
//4 байта)
    setAttributeF(UDINT ch_id, UINT attr_id,
    REAL value)
```

Функции чтения и записи в порт поддерживаются только в Микро МРВ для DOS.

В качестве **ch_id** может выступать число, равное ID канала, или аргумент, привязанный к атрибуту 118, **ID** канала.

В качестве **attr_id** и **value** могут выступать числа или аргументы с соответствующими числовыми значениями.

Пользовательские функции Техно ST

В **Техно ST** определены две разновидности пользовательских функций: собственно функция и функция-блок. Функция-блок играет роль подпрограммы, т.е. не возвращает значений.

Функции и функции-блоки создаются с помощью табличного редактора. Если при создании указан тип данного компонента, создается функция, в противном случае – функция-блок.

Определение функции и функции-блока

Синтаксис:

```
function {имя} : {тип возвращаемого значения}
    {определение аргументов}
    {список предложений}
end_function
```

```
function_block {имя}
    {определение аргументов}
    {список предложений}
end_function_block
```

Эти конструкции создаются автоматически при создании функции и функции-блока в табличном редакторе.

Необязательное выражение {определение аргументов} задается аналогично выражению {определение переменной} для операторов определения переменной (см. **Операторы определения переменных** в разделе **Операторы Техно ST**). Тело функции составляет {список предложений}.

Если при определении аргумента задано его значение, он считается опциональным. Такой аргумент может не указываться в вызове функции; в этом случае используется значение, заданное аргументу при его описании. После определения опционального аргумента не может следовать определение аргумента, для которого значение не задается (такой аргумент обязательно указывается в вызове функции).

Функция может возвращать значение структурного типа, а также массив.

Вызов функции и функции-блока

Синтаксис:

```
{имя функции} ({аргументы})
```

где {аргументы} – последовательность выражений, разделенных запятыми, или

```
{имя функции} ()
```

если аргументы у функции или функции-блока отсутствуют.

Число аргументов в вызове функции (функции-блока) должно быть равно числу аргументов, заданных для этой функции (функции-блока).

Передача аргумента по значению или по ссылке задается определением

аргумента функции (см. **Операторы определения переменных** в разделе **Операторы Техно ST**). Строки и массивы всегда передаются по ссылке. В случае, когда соответствующий аргумент определяет передачу по значению, передача все равно осуществляется по ссылке, а аргумент считается константой.

Термин **по значению** обозначает передачу значения аргумента в вызываемую функцию. При этом функция оперирует с копией переменной и не может изменить ее истинное значение. Термин **по ссылке** обозначает передачу адреса аргумента в вызываемую функцию. При этом функция оперирует с самой переменной и может изменить ее значение.

Вызываемая функция или функция-блок может быть запрограммирована на любом из встроенных языков (функцию нельзя запрограммировать на **Техно SFC**) – см. также **Создание пользовательских функциональных блоков**.

Следующие функции (функции-блоки) могут быть вызваны в основной программе только однократно:

- содержащие глобальные переменные программы;
- содержащие FBD-блоки с внутренними переменными (см. **Редактирование FBD-программ**).

Другие функции могут вызываться в основной программе многократно.

Пример

Данная программа выполняет возведение в квадрат с использованием ST-функции. При этом показано отличие передачи значения аргумента по ссылке и по значению:

```
FUNCTION SecondDegree: LREAL
//VAR_ARG определяет передачу значения
//переменной loc_var1
    VAR_ARG xx: REAL; END_VAR
//VAR_INOUT определяет передачу адреса
//переменной loc_var2
    VAR_INOUT yy: REAL; END_VAR
//следующее действие не изменяет
//значение loc_var1
    xx = xx+1;
//следующее действие изменяет значение loc_var2
    yy = yy+1;
    RETURN (xx+yy)**2;
END_FUNCTION
```

```
PROGRAM
  VAR
    loc_var1:INT;
    loc_var2:INT;
    d : LREAL;
  END_VAR
  loc_var1 = 2;
  loc_var2=1;
  // передача loc_var1 и loc_var2 в функцию,
  // способ передачи задается в функции.
  // Переменной d присваивается значение,
  // определенное в функции оператором RETURN
  d := SecondDegree(loc_var1, loc_var2);
END_PROGRAM
```

Пример

Данный пример демонстрирует особенности использования функции-блока.

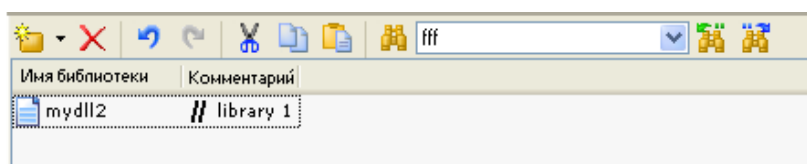
```
FUNCTION_BLOCK Sec_Degree
  VAR_ARG
    xx: REAL;
    yy: REAL;
  END_VAR
  VAR_OUTPUT
    result: LREAL;
  END_VAR
  result =(xx+yy+1)**2;
END_FUNCTION_BLOCK
PROGRAM
  VAR
    loc_var1:INT:=2;
    loc_var2:INT:=1;
    d : LREAL;
  END_VAR
  //Передача значений loc_var1 и loc_var2
  //в функцию-блок. Переменной d присваивается
  //значение переменной result функции-блока.
  //Чтобы такое присвоение работало корректно,
  //переменная result должна быть определена в
  //функции-блоке оператором VAR_OUTPUT или
```

```
//VAR_INOUT
    Sec_Degree(loc_var1, loc_var2, d);
END_PROGRAM
```

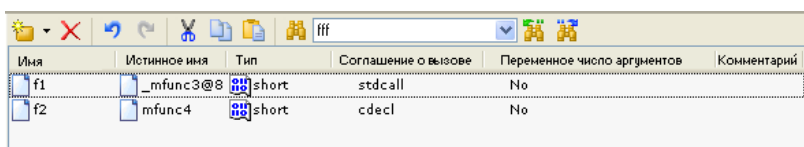
Внешние библиотеки функций

В ST-программе допускается использование функций из динамических библиотек (DLL). Библиотеки должны располагаться в директории, из которой запускается монитор, или в директории, путь к которой указан в переменной PATH операционной системы. Для конфигурирования вызовов внешних функций пользователя используются специальные табличные редакторы.

Имя подключаемой библиотеки и комментарий к ней указываются в редакторе, показанном на рисунке ниже (имя библиотеки указывается без расширения). Этот табличный редактор открывается при нажатии ЛК на группе **Внешние библиотеки** в окне структуры программы.



Библиотечные функции, которые предполагается вызывать из программы, описываются в табличном редакторе, который открывается при нажатии ЛК на имени внешней библиотеки в окне структуры программы:



При описании внешней функции в этом редакторе должны быть заданы следующие параметры:

- **Внешнее имя** – имя, под которым внешняя функция будет вызываться из программы (задается разработчиком программы). Под этим именем внешняя функция добавляется в структуру программы;
- **Истинное имя** – имя функции, экспортируемое библиотекой;
- **Тип возвращаемого значения** – тип данных, возвращаемых внешней функцией (см. **Определение переменных и констант**);
- **Соглашение о вызове** – соглашение о вызове внешней функции;
- **Переменное число аргументов** – указание, является ли внешняя функция функцией переменного числа аргументов (возможные

опции в этом поле – **да** и **нет**);

- **Комментарий** – необязательный комментарий к функции (задается разработчиком программы).

Аргументы внешних функций описываются в табличном редакторе, который открывается при нажатии ЛК на группе **Аргументы** функции в окне структуры программы:

| Имя | Тип данных | Указатель | Комментарий |
|-----------|------------|-----------|-------------|
| N ARG_000 | long | Нет | |
| N ARG_001 | char | Да | // arg_1 |
| N ARG_002 | unsigned | Да | |
| N ARG_003 | short | Нет | |
| N ARG_004 | unsigned | Да | |
| | long | | |
| | unsigned | | |
| | float | | |
| | double | | |
| | STRING | | |
| | HANDLE | | |

В этом редакторе для каждого аргумента указываются следующие параметры:

- **Тип данных** – тип данных аргумента, выбирается из списка;
- **Указатель** – если аргумент внешней функции является указателем, в этом поле должно быть указано **да**, если переменной – **нет**.

При взаимодействии с внешней библиотекой строка передается в формате Unicode (16 бит). Переменная TRACE MODE должна иметь тип данных STRING, указатель на строку в DLL – WCHAR. Для перевода из WCHAR в CHAR в DLL может использоваться функция WideCharToMultiByte.

Если при загрузке узла в MPB DLL не найдена, в протокол профайлера (см. **Профайлеры**) записывается сообщение вида **WRN_LOAD:Templates load error 22,ID=0 Type=1**.

Массивы Техно ST

Массив – это совокупность объектов одного типа, имеющая общее имя (идентификатор). Объекты, составляющие массив, называются элементами этого массива.

Для задания массивов используются операторы определения переменных, для обращения к элементам массивов – оператор индексирования, в связи с чем элементы массивов в **Техно ST** называются также индексированными переменными.

В **Техно ST** определены одномерные и многомерные массивы.

Одномерные массивы

Одномерный массив – это массив переменных; для обращения к элементу одномерного массива достаточно указания одного индекса. В приведенном ниже примере показаны различные варианты задания одномерных массивов.

Пример

```
PROGRAM
    VAR
        //Массив 7 переменных типа REAL.
        //Элементы массива по умолчанию инициализируются
        //нулевыми значениями:
        nn: array [7] of REAL;
        //Массив 3 переменных типа REAL.
        //Первые два элемента инициализируются
        //заданными значениями
        //(n[0]=1.5, n[1] = 10), n[2] по умолчанию
        //инициализируется нулевым значением:
        n: array [3] of REAL := 1.5, 2*5;
    END_VAR

    VAR
        //Массив 4 строковых переменных.
        //Первые три элемента инициализируются
        //заданным значением
        //(m[0] = m[1] = m[2] = "OK"), m[3] по умолчанию
        //инициализируется пустой строкой:
        m: array [4] of STRING := 3("OK");
    END_VAR
    //Задание значений элементам массивов:
    n[2] = 7*2.5;
    m[3] = "ERROR";
END_PROGRAM
```

Многомерные массивы

Многомерный массив – это массив массивов. Определение многомерного массива в общем случае должно содержать сведения о типе, размерности и количествах элементов каждой размерности. Например, трехмерный

массив, заданный конструкцией

```
kk: array [4,3,2] of INT;
```

представляет собой массив 4 элементов, каждый из которых – двумерный массив с размерами 3 на 2. Данный массив содержит 24 элемента типа **INT**, которые в приведенном ниже перечне расположены в порядке возрастания адресов (слева направо):

```
kk[0,0,0], kk[0,0,1], kk[0,1,0], kk[0,1,1],
kk[0,2,0], kk[0,2,1], kk[1,0,0], kk[1,0,1],
kk[1,1,0], kk[1,1,1], kk[1,2,0], kk[1,2,1],
kk[2,0,0], kk[2,0,1], kk[2,1,0], kk[2,1,1],
kk[2,2,0], kk[2,2,1], kk[3,0,0], kk[3,0,1],
kk[3,1,0], kk[3,1,1], kk[3,2,0], kk[3,2,1]
```

Пример

В данном примере показаны различные варианты задания многомерных массивов.

```
PROGRAM
  VAR
    //двумерный массив переменных
    //INT с заданием начальных значений
    //(ll[0,0]=1, ll[0,1]=2, ll[0,2]=3,
    // ll[1,0]=4, [1,1]=5, ll[1,2]=6):
    ll: array [2,3] of INT := 1,2,3,4,5,6;
  END_VAR
  VAR
    //двумерный массив строковых
    //переменных с указанием диапазонов индексов и
    //заданием начального значения первых четырех
    //элементов
    //(pp[5,9]=pp[5,10]=pp[5,11]="OK",
    //pp[6,9]= "NO",
    //pp[6,10]= ...=pp[7,11]=""):
    pp: array[5 .. 7, 9 .. 11] of
      STRING:=3("OK"), "NO";
  END_VAR
  //задание значений элементов массивов
  ll[1,2] = 17*5;
  pp[6,10] = "ERROR";
END_PROGRAM
```

Структуры Техно ST

В отличие от массива, структура – это совокупность объектов, имеющих различный тип.

Для создания структур нужно вначале с помощью табличных редакторов определить структурные типы, а также переменные, аргументы и функции, которые являются членами этих структурных типов.

Для справки: во внутреннем представлении языка **Техно ST** определение структурного типа задается следующей конструкцией:

```
type
    {имя_структурного_типа} : STRUCT
        //определения переменных –
        //членов структуры
        //(см. "Операторы определения
        //переменных")
        //определения функций –
        //членов структуры
        //(см. "Пользовательские функции
        //Техно ST")
    end_struct;
end_type
```

Параметр **{имя_структурного_типа}**, а также переменные и функции – члены структурного типа – задаются с помощью табличных редакторов.

Определив структурный тип, можно определять конкретные структуры данного типа (объекты). Такими объектами могут быть переменные или функции.

Объекты создаются с помощью табличных редакторов. Для этого в основной программе или ее компоненте создается переменная (функция) типа **{имя_структурного_типа}**.

Для обращения к элементам объекта используется так называемое уточненное имя:

```
{имя объекта} . {имя элемента}
```

В уточненном имени не поддерживается адресация к элементу массива объекта.

Пример

Пусть в программе создан структурный тип с именем **myType**, членами которого являются следующие переменные и функция:

```
mvar1: REAL:=3.14;
mvar2: STRING := "OK";
    FUNCTION SecondDegree: LREAL
        VAR_ARG xx: REAL; END_VAR
        RETURN xx**2;
    END_FUNCTION
```

Создание и использование объекта типа **myType** иллюстрирует следующий код:

```
PROGRAM
    VAR
        d: LREAL;
        myS : myType; //объект myS типа myType
    END_VAR
        d=myS.SecondDegree(25); //возвращает 625
        d=myS.mvar1; //возвращает 3.14
//изменение значения mvar1 объекта myS
        myS.mvar1 = 10;
    END_PROGRAM
```

Таким образом, структурный тип является только шаблоном структур. С именем типа не связан никакой конкретный объект, поэтому это имя нельзя использовать, например, для формирования уточненного имени элемента:

```
//ошибочная конструкция
myType.mvar1
```

Описание языка Техно IL

Синтаксис Техно IL

Программа на языке **Техно IL** представляет собой последовательность **инструкций**. Каждая инструкция должна начинаться с новой строки и должна содержать **оператор** с опциональным **модификатором** и, для некоторых операций, один или более **операндов**, разделенных пробелами. Между инструкциями могут располагаться пустые строки. Компилятор не чувствителен к регистру, т.е. инструкции **add var_002** и **ADD VAR_002** равнозначны.

Примеры IL-инструкций

```
ADD VAR_000 2.6
LT VAR_000 VAR_001
JMPC label1
GT VAR_001 20
JMPC label2
LD 278
label1: CAL FUNCTION_000 (VAR_000, VAR_001)
label2: ST VAR_001
```

Под **аккумулятором** в **Техно IL** понимается хранилище текущего результата вычислений (в этом качестве выступает один из регистров процессора). Далее в описании языка **Техно IL** значение аккумулятора обозначается словом **result**. Функция на языке **Техно IL** возвращает **result**.

Техно IL поддерживает одноадресный и двухадресный режимы записи инструкций, которые оперируют с двумя операндами. В первом случае первым операндом является аккумулятор, который опускается при записи, во втором случае указываются два операнда.

Пример

В данном примере представлена запись процедуры $a = a + b$ в одноадресном и двухадресном режиме. Одноадресный режим:

```
LD a    //result = a
ADD b    //result = result + b
ST a    //a = result
```

Двухадресный режим позволяет записать ту же операцию компактнее:

```
ADD a b  // a = a + b
```

В IL-программе могут использоваться метки и комментарии. Правила их задания аналогичны правилам **Техно ST** (см. **Комментарии Техно ST** и описание оператора **goto** в разделе **Операторы Техно ST**).

Операнды Техно IL

В качестве операндов операторов **Техно IL** могут выступать переменные и константы всех типов, определенных в языке **Техно ST** (см. **Переменные и константы Техно ST**). В некоторых случаях в качестве операнда может выступать число.

Для операторов перехода и вызова функции операндом является соответственно имя метки и имя функции.

Переменные IL-программы задаются аналогично переменным **Техно ST** (см. **Операторы Техно ST**).

Операторы и модификаторы Техно IL

Модификаторы Техно IL

Модификаторы **Техно IL** – это литеры **N**, **C** и **X**, которые могут быть приписаны справа к имени ряда операторов.

Модификатор **N** обозначает логическое отрицание операнда. Например, инструкция

```
AND a
```

интерпретируется как **result = result AND a**, а инструкция

```
ANDN a
```

интерпретируется как **result = result AND NOT a**.

Для операторов **JMP**, **CAL** и **RET**:

- модификатор **C** обозначает, что инструкция выполняется в том случае, если результат предыдущей операции сравнения истинен;
- модификатор **X** обозначает, что инструкция выполняется в том случае, если **result = TRUE**.

Операторы обмена с аккумулятором

| Синтаксис | Допустимый модификатор | Действие |
|------------|------------------------|-------------------|
| LD operand | N | result := operand |
| ST operand | N | operand := result |

Знак "!=" в таблице обозначает операцию присваивания.

В качестве операнда может использоваться численная или булева переменная. В качестве операнда оператора **LD** может использоваться число.

Отличное от нуля значение аккумулятора интерпретируется как TRUE, нулевое – как FALSE, поэтому значение аккумулятора может быть присвоено как численной, так и булевой переменной.

Пример

```

VAR VAR_000 : INT := 10; END_VAR
VAR VAR_001 : BOOL := TRUE; END_VAR
VAR VAR_002 : BOOL; END_VAR
LD 8 //result := 8
ST VAR_000 //VAR_000 := 8
ST VAR_002 //VAR_002 := TRUE
LD 0 //result := 0
ST VAR_001 //VAR_001 := FALSE
LD VAR_001 //result := FALSE
ST VAR_002 //VAR_002 := FALSE
ST VAR_000 //VAR_000 := 0
    
```

Логические операторы Техно IL

| Синтаксис | Допустимый модификатор | Действие |
|-----------------------|------------------------|---|
| S operand | | operand := TRUE (см. примечание) |
| R operand | | operand := FALSE (см. примечание) |
| AND operand1 operand2 | N | result := operand1 := operand1 AND operand2 |
| OR operand1 operand2 | N | result := operand1 := operand1 OR operand2 |
| XOR operand1 operand2 | N | result := operand1 := operand1 XOR operand2 |

Примечание. Оператор выполняется только тогда, когда **result = TRUE**.

В качестве операндов могут использоваться булевы переменные. Вторым операндом может быть число (но не численная переменная), которое интерпретируется следующим образом: не равно 0 – TRUE; равно 0 – FALSE.

Выполнение операторов **R** и **S** не изменяет значения аккумулятора.

Пример

```

VAR VAR_001 : BOOL := TRUE; END_VAR
VAR VAR_002 : BOOL; END_VAR
    
```

```

VAR VAR_004 : INT := 0; END_VAR
LD 1 //result:=1
S VAR_002 //VAR_002:=TRUE
R VAR_002 //VAR_002:=FALSE
AND VAR_001 VAR_002 //result:=VAR_001:=FALSE
LD 1 //result:=1
S VAR_001 //VAR_001:=TRUE
OR VAR_002 VAR_001 //result:=VAR_002:=TRUE
XOR VAR_002 VAR_001 //result:=VAR_002:=FALSE
OR VAR_002 10 //result:=VAR_002:=TRUE

```

Арифметические операторы Техно ИЛ

| Синтаксис | Действие |
|-----------------------|---|
| ADD operand1 operand2 | result := operand1 := operand1 + operand2 |
| SUB operand1 operand2 | result := operand1 := operand1 - operand2 |
| MUL operand1 operand2 | result := operand1 := operand1 * operand2 |
| DIV operand1 operand2 | result := operand1 := operand1 : operand2 |

В качестве операндов используются численные переменные, в качестве второго операнда может использоваться число.

Арифметические операторы не допускают использования модификаторов.

Пример

```

VAR VAR_000 : REAL := 20; END_VAR
VAR VAR_001 : LREAL := 30; END_VAR
ADD VAR_000 10 //result := VAR_000 := 30
MUL VAR_001 9 //result := VAR_001 := 270
SUB VAR_001 VAR_000 //result := VAR_001 := 240
DIV VAR_001 VAR_000 //result := VAR_001 := 8

```

Операторы сравнения Техно ИЛ

| Синтаксис | Действие |
|----------------------|---|
| GT operand1 operand2 | result := TRUE, если operand1 > operand2 |
| GE operand1 operand2 | result := TRUE, если operand1 >= operand2 |
| EQ operand1 operand2 | result := TRUE, если operand1 == operand2 |
| NE operand1 operand2 | result := TRUE, если operand1 <> operand2 |
| LE operand1 operand2 | result := TRUE, если operand1 <= operand2 |
| LT operand1 operand2 | result := TRUE, если operand1 < operand2 |

В качестве операндов используются численные переменные, в качестве

второго операнда может использоваться число.

Операторы сравнения не допускают использования модификаторов.

Операторы сравнения, как правило, предшествуют операторам **JMPC**, **CALC** и **RETC**. Если результат сравнения ложен, инструкции сравнения не изменяют значения аккумулятора, а последующий оператор **JMPC**, **CALC** или **RETC** игнорируется (даже если **result = TRUE**).

Пример

```
VAR VAR_000 : INT := 20; END_VAR
VAR VAR_001 : INT := 30; END_VAR
VAR VAR_002 : BOOL; END_VAR
LD 1 //result := TRUE
GT VAR_000 VAR_001 //результат сравнения ложен,
//т.к. (20<30), аккумулятор сохраняет свое
//значение (TRUE)
RETC //RETC игнорируется
LD VAR_002 //result := FALSE
LT VAR_000 VAR_001 //результат сравнения истинен
//result := TRUE
CALC fff(VAR_000) //вызов функции произойдет
```

Операторы перехода и вызова функции Техно II

| Синтаксис | Допустимый модификатор | Действие |
|---------------------------------|------------------------|---|
| JMP имя_метки | C, X | переход к строке с указанной меткой |
| CAL имя_функции(val1, ... valN) | C, X | вызов функции или функции-блока |
| RET | C, X | выход из программы, функции или функции-блока |

Операторы перехода выполняются, если строка с указанной меткой находится в том же программном компоненте.

CAL и **CALL** являются равнозначными операторами. В круглых скобках через запятую указываются значения, передаваемые в функцию. Между именем функции и круглыми скобками пробел необязателен. Число передаваемых в функцию значений должно быть равно числу аргументов, заданных для этой функции.

При выполнении оператора **RET** и его разновидностей функция возвращает значение **result**.

Определены следующие модификации данных операторов:

JMP, CAL, RET – соответственно оператор безусловного перехода, безусловного вызова и безусловного выхода.

JMPX, CALX, RETX – соответственно оператор условного перехода, условного вызова и условного выхода. Инструкция, содержащая любой из этих операторов, выполняется только тогда, когда **result=TRUE**, в противном случае игнорируется.

JMPC, CALC, RETC – соответственно оператор условного перехода, условного вызова и условного выхода. Эти операторы следуют непосредственно за оператором сравнения. Инструкция, содержащая любой из этих операторов, выполняется только тогда, когда результат предыдущей операции сравнения истинен (см. выше **Операторы сравнения Техно IL**), в противном случае игнорируется.

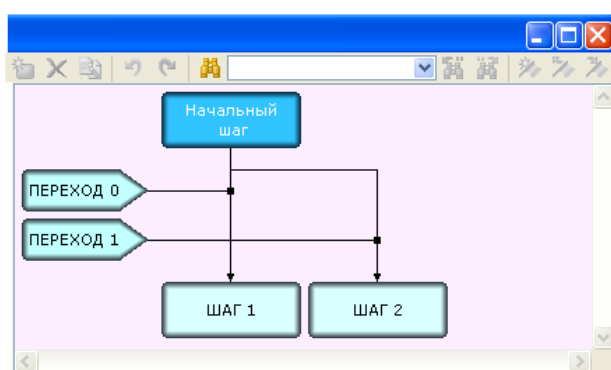
Редактирование SFC-программ

Язык **Техно SFC** позволяет создавать программы в виде алгоритма, состоящего из SFC-шагов и SFC-переходов.

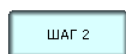
Для SFC-шагов задаются выполняемые действия, для SFC-переходов – условия переходов между шагами, поэтому в дальнейшем SFC-переход иногда называется SFC-условием.

Для перехода от одного шага к другому SFC-условие, действующее на этом переходе, должно быть истинным (т.е. возвращать TRUE или 1).

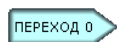
Вид программы в SFC-редакторе показан на рисунке:



Для отображения алгоритма в редакторе используются следующие элементы:



– SFC-шаг. Число указывает номер шага;



– SFC-условие. Число указывает номер перехода.



Имя элемента можно изменить. Для перехода в режим редактирования этого параметра нужно дважды нажать ЛК на элементе; для выхода из режима редактирования нужно нажать ОК.

Направление перехода от одного шага к другому указывает линия со стрелкой. Линия, соединяющая SFC-условие с линией перехода между шагами, указывает, на каком переходе действует данное условие.

SFC-программа может выступать в роли основной программы и функции-блока. Запрограммировать функцию на языке **Техно SFC** нельзя.

Выделение элементов алгоритма SFC

Для выделения элемента алгоритма нужно с помощью мыши установить

на него курсор, вид которого при этом меняется с  на , и нажать ЛК.


При выделении SFC-шага выделяется только сам шаг.

При выделении SFC-условия выделяются само условие, линия, соединяющая условие с линией перехода между шагами, и линия перехода. Аналогичное действие выполняется при нажатии ЛК на линии, соединяющей условие с линией перехода между шагами.

При нажатии ЛК на линии перехода между шагами выделяются данная линия перехода и старшее по номеру условие, действующее на этом переходе.

Повторное нажатие ЛК снимает выделение.


Задание SFC-шагов и SFC-условий

Чтобы перейти к редактированию SFC-условия или действия, выполняемого на SFC-шаге, нужно нажать ЛК на имени шага (условия) в окне структуры программы или нажать кнопку  панели инструментов SFC-редактора.

Для вновь созданного SFC-шага или SFC-условия после описанной процедуры автоматически появляется диалог, в котором следует выбрать язык программирования, после чего SFC-шаг (SFC-условие) открывается в соответствующем редакторе.

Редактирование алгоритма SFC

При создании SFC-программы первый SFC-шаг с заданным по умолчанию именем создается автоматически. Чтобы добавить или удалить элементы алгоритма, нужно нажать ЛК на соответствующих кнопках панели инструментов SFC-редактора:

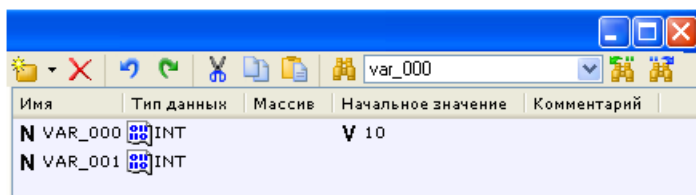
- Для добавления шага и условия нужно выделить шаг, вслед за которым должны располагаться добавляемые элементы, и нажать кнопку .
- Для добавления SFC-условия нужно нажать ЛК на начальном шаге и, удерживая кнопку нажатой, переместить курсор на конечный шаг, после чего кнопку мыши отпустить.

Данное свойство можно использовать для создания цикла, если в качестве начального и конечного шагов использовать один и тот же шаг.

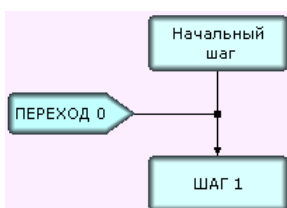
Пример создания цикла в SFC-программе

В данном примере в SFC-программе создается цикл. В качестве счетчика цикла выступает переменная VAR_000.

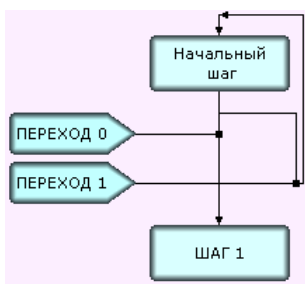
Создадим новую программу, выберем для нее язык SFC и зададим следующие переменные:



Выделим на диаграмме начальный шаг и нажмем кнопку . Диаграмма SFC примет следующий вид:



Нажмем ЛК на начальном шаге и, удерживая кнопку нажатой, сместим курсор в пределах графического изображения шага и отпустим кнопку мыши. Диаграмма примет следующий вид:



Используя язык **Техно ST**, зададим шаги и условия следующим образом:

```

SFC_STEP "Начальный шаг"
  VAR VAR_000 : INT := 10; END_VAR
  VAR VAR_001 : INT ; END_VAR
  VAR_000 = VAR_000 + 1; //увеличение
                        //счетчика на 1
  VAR_001 = VAR_001 + 2; //выполняемое в
                        //цикле действие
END_SFC_STEP
SFC_TRANSITION "ПЕРЕХОД 0" FROM( INITIAL_STEP )
TO( STEP_1 )
  VAR VAR_000 : INT := 10; END_VAR
  VAR VAR_001 : INT ; END_VAR
  
```

```

    VAR_000 == 20
END_SFC_TRANSITION
SFC_TRANSITION "ПЕРЕХОД 1" FROM( INITIAL_STEP )
TO( INITIAL_STEP )
    VAR VAR_000 : INT := 10; END_VAR
    VAR VAR_001 : INT ; END_VAR
    VAR_000 < 20
END_SFC_TRANSITION
SFC_STEP "ШАГ 1"
    VAR VAR_000 : INT := 10; END_VAR
    VAR VAR_001 : INT ; END_VAR
    VAR_001=200; //действие после выхода
                //из цикла
END_SFC_STEP


```

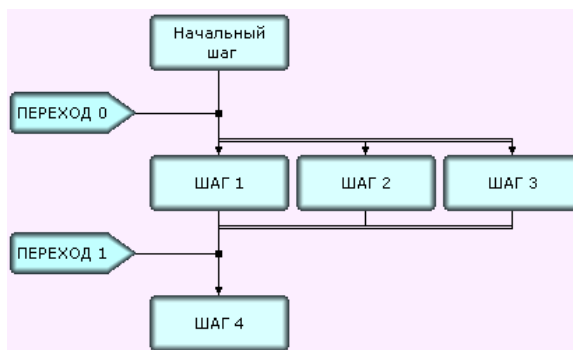
В этом примере выполнение шага **Начальный шаг** повторяется 10 раз (пока счетчик растет от 10 до 20). Данный алгоритм работает аналогично следующему коду **Техно ST**:

```

VAR VAR_000 : INT :=10; END_VAR
VAR VAR_001 : INT; END_VAR
REPEAT VAR_001 = VAR_001 + 2; VAR_000 =
VAR_000 + 1; UNTIL VAR_000 < 20 END_REPEAT;

```

- Для добавления дополнительного (параллельного) шага, который будет выполняться по созданному ранее условию, нужно выделить это условие или выходящую из него линию, и нажать кнопку . Данное свойство используется для более структурированного представления алгоритма. Например, при выполнении следующей диаграммы условие **ПЕРЕХОД 1** будет проверяться только после выполнения шагов **ШАГ 1**, **ШАГ 2** и **ШАГ 3**.



Параллельные шаги должны быть обязательно связаны с последующим условием, причем с одним и тем же. Чтобы реализовать такую связь, нужно нажать ЛК на параллельном шаге и, удерживая кнопку нажатой, переместить курсор на условие или линию пере-

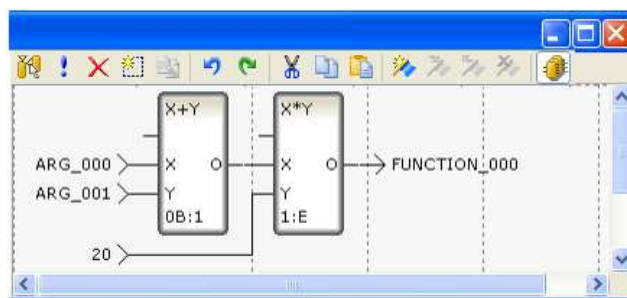
хода, на котором это условие действует.

Для удаления параллельного шага нужно выделить его и нажать кнопку **X** или клавишу **Del**.

- Для удаления SFC-условия нужно выделить его и нажать кнопку **X**. Единственное условие, действующее на переходе между шагами, удалить нельзя.
- Для удаления шага и всех связанных с ним SFC-условий нужно выделить этот шаг и нажать кнопку **X**. Удалить можно только те шаги, которые не имеют переходов к другим шагам (из таких шагов не выходят линии со стрелкой).

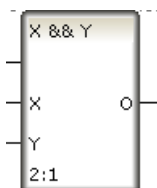
Редактирование FBD-программ

FBD-программа представляет собой цепочку (диаграмму) последовательно выполняемых **функциональных блоков**. На рисунке показан вид программы, состоящей из двух блоков, в FBD-редакторе.



Функциональный блок – это графическое изображение вызова встроенной функции **Техно FBD** (FBD-блока) или функции (функции-блока), определенной пользователем.

Вид FBD-блока показан на следующем рисунке.



В верхней части блока выводится обозначение функции, выполняемой блоком (**X && Y** на рисунке). Именованные отрезки слева (**X** и **Y**), обозначают входы блока (аргументы, переменные или константы функции). Отрезок без имени слева обозначает вход, управляющий выполнением блока (в дальнейшем – вход **RUN**). Блок выполняется, если **RUN=0** (значение по умолчанию).

Отрезки, примыкающие к блоку справа, обозначают выходы блока (возвращаемые функцией значения).

Кроме входов/выходов, некоторые встроенные FBD-блоки имеют внутренние переменные, недоступные пользователю. Переменные FBD-блока (входы/выходы и внутренние) являются глобальными, т.е. сохраняют свое значение между вызовами программы, в том числе при **RUN=1**.

В нижней части блока выводится его номер и, после двоеточия, номер следующего выполняемого блока (**2:1** на рисунке). Номера блоков задаются последовательно при их размещении в рабочем поле редактора; номера следующих выполняемых блоков определяются автоматически при

соединении входов и выходов блоков (образовании диаграммы). На блоке, который выполняется первым в программе, после его номера отображается символ **В**; на блоке, который выполняется последним, – символ **Е**.


FBD-программа может выступать в роли основной программы, функции и функции-блока.

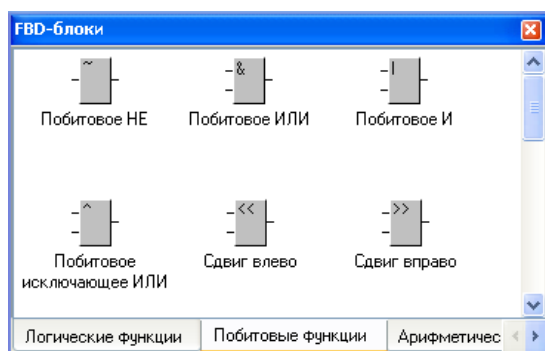
Для создания FBD-программы и подключения ее к проекту нужно выполнить следующие операции:

- разместить необходимые функциональные блоки в рабочем поле FBD-редактора;
- соединить нужные входы и выходы блоков, образовав единую диаграмму;
- задать аргументы, переменные и константы программы;
- привязать входы/выходы FBD-диаграммы к аргументам, переменным и константам программы;
- скомпилировать программу.

Размещение FBD-блоков в рабочем поле редактора

Рабочее поле FBD-редактора с помощью сетки разбито на участки. На одном участке можно разместить один функциональный блок.


Выбрать нужный функциональный блок для размещения можно с помощью специального навигатора, показанного на рисунке. Открыть/закрыть окно навигатора функциональных блоков можно с помощью кнопки  панели инструментов FBD-редактора.



В нижней части навигатора находятся кнопки выбора группы блоков (**Логические**, **Арифметические** и т.п.), в рабочем поле навигатора отображаются блоки, которые входят в выбранную группу.



Чтобы получить справку о блоке, нужно дважды нажать ЛК на его изображении в навигаторе.

Чтобы поместить блок на участок рабочего поля FBD-редактора, нужно

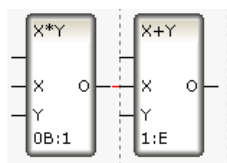
нажать ЛК на изображении блока в навигаторе и, удерживая кнопку нажатой, переместить курсор на нужный участок, после чего кнопку отпустить (метод drag-and-drop). Если курсор принимает вид , размещение блока в выбранной области невозможно.

Редактирование диаграммы FBD-блоков

Для выделения элемента FBD-диаграммы (функционального блока, входа или выхода блока, связи между блоками, графического изображения привязки входа/выхода FBD-диаграммы к аргументу/переменной программы, метки) нужно с помощью мыши установить на него курсор, вид которого

при этом меняется с  на , и нажать ЛК. Выделенный элемент обозначается цветом, заданным в параметрах FBD-редактора. Для выделения группы элементов нужно с помощью мыши обвести их контурным прямоугольником (нажать ЛК в некоторой точке рабочего поля редактора и, удерживая кнопку нажатой, переместить курсор в направлении диагонали будущего прямоугольника, после чего кнопку отпустить).

Создание связей между блоками производится методом drag-and-drop (выделить вход/выход блока, нажать ЛК на изображении этого входа/выхода и, удерживая кнопку нажатой, переместить курсор на изображение выхода/входа другого блока, после чего кнопку отпустить). Созданная связь обозначается на диаграмме линией:



Блоки можно перемещать на другие участки рабочего поля FBD-редактора методом drag-and-drop, при этом созданные связи сохраняются.


Информация о блоке или связи может быть получена из всплывающей подсказки:


| Информация по FBD блоку | |
|-------------------------|------------------------|
| ID блока: | 0 |
| Позиция: | 1:1 |
| Функция: | X*Y |
| Описание: | Сложение |
| Следующий: | 3 (X) |
| Выходы | |
| 0 (O) | -> FBD блок 1:1 (X+YX) |
| 0 (O) | -> FBD блок 2:1 (X+YX) |
| 0 (O) | -> ARG_002 |
| 0 (O) | -> ARG_002 |

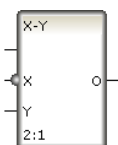
| Информация по связи FBD диаграммы | |
|-----------------------------------|----------------------|
| Ис: | FBD блок 0:0 (X+Y:O) |
| В: | FBD блок 2:1 (X+YX) |


Кроме панели инструментов, для редактирования диаграмм FBD-редактор снабжен набором контекстных меню, доступных по нажатию ПК после выделения элемента диаграммы. Помимо типовых инструментов для ре-

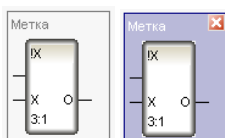
дактирования, работы с буфером обмена и закладками, панель инструментов и меню включают следующие команды:

 **Привязать** – перейти в режим привязки выделенного входа/выхода к аргументу/переменной программы;

 **Инвертировать** – инвертировать вход/выход блока. На рисунке показано обозначение инвертирования входа **X**:



 **Создать метку** – при выполнении этой команды для участка размещения выделенного блока создается метка (рисунок слева):




После выделения метки (рисунок справа) возможно ее удаление с помощью стандартного инструмента или контекстного меню.

Для перехода к редактированию метки нужно дважды нажать на ней ЛК.

 **Редактировать** – редактировать код пользовательского блока;


Перейти – меню перехода к следующему выполняемому блоку или к привязанному входу/выходу;

 **Справка** – открыть окно справки с описанием блока.

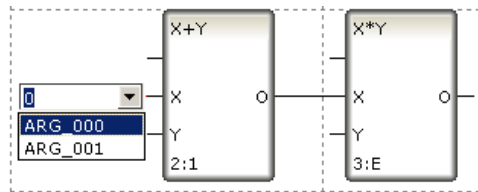
Привязка входов и выходов FBD-диаграммы

Входы и выходы функциональных блоков могут выступать в качестве входов и выходов FBD-диаграммы. Они могут быть привязаны к аргументам или переменным, заданным для данной FBD-программы с помощью табличных редакторов, а также к глобальным переменным.

FBD-диаграмма должна иметь по меньшей мере одну привязку.

Для привязки входа/выхода FBD-диаграммы нужно выделить этот вход/выход и нажать кнопку  панели инструментов или нажать ПК и выполнить команду **Привязать** из контекстного меню. При этом возле выделенного входа/выхода выводится окно со списком доступных для привязки аргументов/переменных, показанное на рисунке ниже. Для вхо-

да в этом окне можно также задать постоянное значение (с помощью клавиатуры).



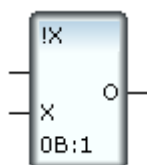
Функция (SFC-условие) на языке **Техно FBD** не возвращает значений, если ни один из выходов FBD-диаграммы не привязан к имени функции (SFC-условия).

Описание FBD-блоков

Раздел 'Логические'

На вход блоков этого раздела можно подавать числовые значения, а также значения типа **BOOL**.

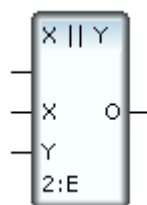
НЕ (!X)



$$O = \text{NOT } X$$

$O=1$, если $X=0$, во всех остальных случаях $O=0$.

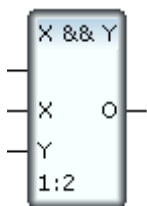
ИЛИ (X || Y)



$$O = X \text{ OR } Y$$

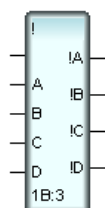
$O=0$, если одновременно $X=0$ и $Y=0$, во всех остальных случаях $O=1$.

И (X && Y)



$$O = X \text{ AND } Y$$

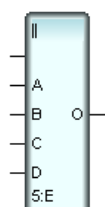
$O=1$, если X и Y одновременно отличны от нуля, во всех остальных случаях $O=0$.

4 НЕ (!)

$$!A = \text{NOT } A; !B = \text{NOT } B; !C = \text{NOT } C; !D = \text{NOT } D$$

Выход равен 1, если соответствующий вход равен 0, во всех остальных случаях выход равен 0.

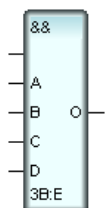
Если вход не определен, его значение принимается равным 0.

Логическое сложение четырех элементов (||)

$$O = A \text{ OR } B \text{ OR } C \text{ OR } D$$

O=1, если хотя бы один из входов отличен от нуля. **O=0**, если **A=B=C=D=0**.

Если вход не определен, его значение принимается равным 0.

Логическое умножение четырех элементов (&&)

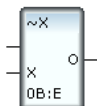
$$O = A \text{ AND } B \text{ AND } C \text{ AND } D$$

O=1, если все входы одновременно отличны от 0, во всех остальных случаях **O=0**.

Если вход не определен, его значение принимается равным 0.

Раздел 'Побитовые'

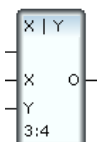
Побитовое НЕ ($\sim X$)



Если входу присвоено численное значение с помощью клавиатуры, блок выполняет 32-разрядную побитовую операцию **NOT** (16-разрядную, если это задано в настройках компилятора). Например, $O=16\#FFFFFFFE$ при $X=1$.

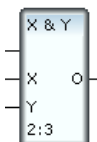
Если вход привязан к переменной, то тип этой переменной определяет разрядность выхода.

Побитовое ИЛИ ($X | Y$)



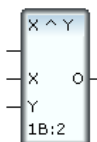
Блок выполняет побитовую операцию **OR**. Например, $O=3$ при $X=2$ и $Y=3$.

Побитовое И ($X \& Y$)

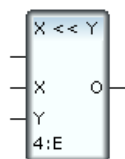


Блок выполняет побитовую операцию **AND**. Например, $O=2$ при $X=2$ и $Y=3$.

Побитовое исключающее ИЛИ ($X \wedge Y$)



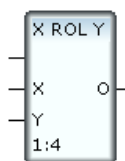
Блок выполняет побитовую операцию **XOR**. Например, $O=1$ при $X=2$ и $Y=3$.

Сдвиг влево ($X \ll Y$)

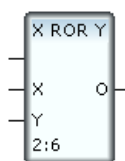
Блок смещает значение входа **X** влево на число разрядов, заданное значением входа **Y**. Справа число дополняется нулевыми разрядами. Например, $0 = 16\#30$ при $X = 16\#3$ и $Y = 4$.

Сдвиг вправо ($X \gg Y$)

Блок смещает значение входа **X** вправо на число разрядов, заданное значением входа **Y**. Слева число дополняется нулевыми разрядами. Например, $0 = 16\#3$ при $X = 16\#30$ и $Y = 4$.

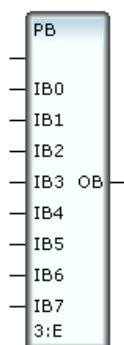
Циклический сдвиг влево ($X \text{ ROL } Y$)

Блок смещает значение входа **X** влево на число разрядов, заданное значением входа **Y**. Справа число дополняется разрядами, которые при сдвиге "выбывают" слева.

Циклический сдвиг вправо ($X \text{ ROR } Y$)

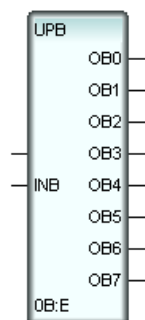
Блок смещает значение входа **X** вправо на число разрядов, заданное значением входа **Y**. Слева число дополняется разрядами, которые при сдвиге "выбывают" справа.

Упаковка битов (PB)



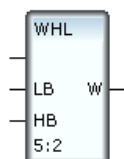
Блок формирует значение битов выхода **OB** по значению соответствующих входов. При ненулевом значении входа соответствующий ему бит выхода равен 1, в противном случае – 0. Вход **IB0** формирует бит 0 выходного значения, **IB7** – бит 7.

Распаковка битов (UPB)

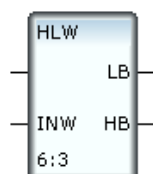


Значения выходов этого блока равны значениям соответствующих битов байта 0 значения входа **INB** (выход **OB0** соответствует биту 0). Например, при **INB=16#FF08** выход **OB3=1**, а все остальные выходы равны нулю.

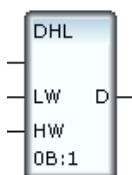
Упаковка байтов (WHL)



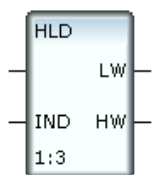
Байт 0 выхода **W** этого блока равен значению байта 0 входа **LB**, байт 1 выхода **W** равен значению байта 0 входа **HB**.

Распаковка байтов (HLW)

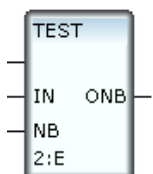
Значение выхода **LB** данного блока равно значению байта 0 входа **INW**, значение выхода **HB** – значению байта 1 входа **INW**.

Упаковка слов (DHL)

Значение слова 0 выхода **D** этого блока равно значению слова 0 входа **LW**. Значение слова 1 выхода **D** равно значению слова 0 входа **HW**.

Распаковка слов (HLD)

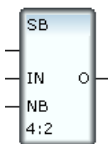
Значение выхода **LW** этого блока равно значению слова 0 входа **IND**, значение выхода **HW** – значению слова 1 входа **IND**.

Чтение бита (TEST)

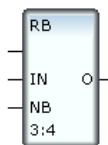
Этот блок передает на выход значение одного из битов значения входа **IN**. Номер передаваемого бита задает вход **NB** (0 соответствует биту 0). Например, **ONB=1** при **IN=16#FFAA** и **NB=3**.

Установка бита (SB)

Данный блок модифицирует значение входа **IN**, устанавливая в 1 один из его битов, и передает модифицированное значение на выход **O**. Вход **NB** задает номер изменяемого бита (0 соответствует биту 0).

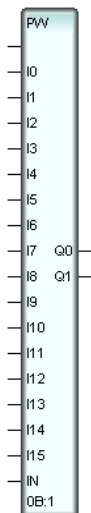


Сброс бита (RB)



Данный блок модифицирует значение входа **IN**, устанавливая в 0 один из его битов, и передает модифицированное значение на выход **O**. Вход **NB** задает номер изменяемого бита (0 соответствует биту 0).

Упаковка битов в слова (PW)



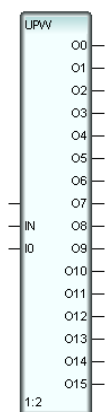
Выходы – 32-разрядные, входы **I0...I15** – биты для упаковки, вход **IN** – управляющий:

- $IN \langle 1, 3, 7 \text{ и } 15$ – соответствующий бит на выходе **Q0** устанавливается в том случае, если $I_i \ \& \ IN \langle 0$;

- **IN=1** – выходное слово в **Q0** формируется из младших битов **I0-I15**;
- **IN=3** – начиная с **I0**, 2 младшие бита каждого входа последовательно записываются в 2 младшие бита **Q0**. Перед каждой записью значение **Q0** сдвигается на 2 разряда влево (упаковка в 2 слова);
- **IN=7** – начиная с **I0**, 3 младшие бита каждого входа последовательно записываются в 3 младшие бита **Q0**. Перед каждой записью значение **Q0** сдвигается на 3 разряда влево;
- **IN=15** – начиная с **I0**, 4 младшие бита каждого входа последовательно записываются в 4 младшие бита **Q0**. Перед каждой записью значение **Q0** сдвигается на 4 разряда влево.

Q1 индицирует число изменений **Q0**.

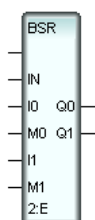
Распаковка слов в биты (UPW)



Вход **I0** – 32-разрядное значение для распаковки, вход **IN** – управляющий:

- если **IN=1, 3, 5, 6, 7** или **15**, то соответствующий выход равен 1, если **I0 & IN <> 0**;
- если **IN** принимает другие значения, выходы равны **I0 & IN**.

Комбинирование битов (BSR)

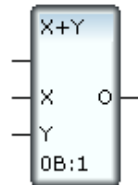


Алгоритм работы блока:

- $Q0=Q1=IN$;
- если $I0 \neq 0$, в $Q0$ устанавливаются биты, которые установлены в $M0$;
- если $I0=0$, в $Q1$ устанавливаются биты, которые установлены в $M0$;
- если $I1 \neq 0$, в $Q0$ устанавливаются биты, которые установлены в $M1$;
- если $I1=0$, в $Q1$ устанавливаются биты, которые установлены в $M1$.

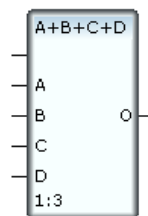
Раздел 'Арифметические'

Сложение двух элементов (X+Y)



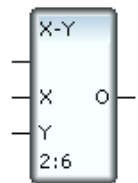
$$O = X + Y$$

Сложение четырех элементов (A+B+C+D)



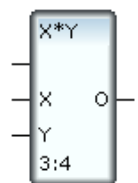
$$O = A + B + C + D$$

Вычитание (X-Y)



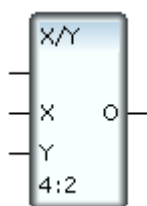
$$O = X - Y$$

Умножение (X*Y)



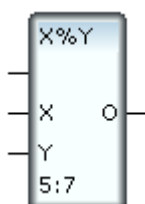
$$O = X * Y$$

Деление (X/Y)



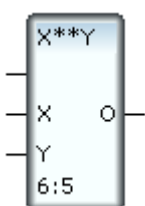
$O = x/y$

Остаток от деления (X%Y)



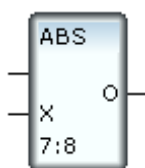
Значение выхода **O** равно остатку от деления значения входа **X** на значение входа **Y**.

*Возведение в степень (X**Y)*

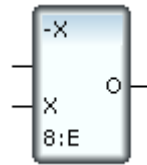


$O = x^y$

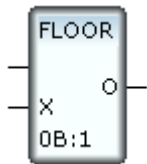
Абсолютное значение (ABS)



$O = |x|$

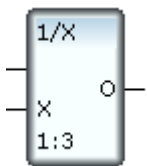
Инверсия знака (-X)

$$O = -X$$

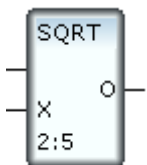
Целая часть (FLOOR)

На выход **O** передается целая часть значения входа **X**.

Не следует путать функцию этого блока с округлением до целого (для округления используется блок **NDGT** из раздела **Алгебраические функции**).

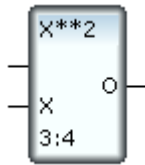
Обратная величина (1/X)

$$O = 1/X$$

Квадратный корень (SQRT)

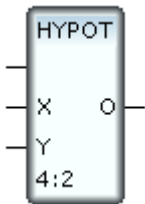
$$O = \sqrt{X}$$

Возведение в квадрат (X2)**



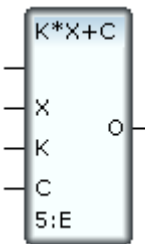
$$O = X^2$$

Сумма квадратов (HYPOT)



$$O = X^2 + Y^2$$

Масштабирование (K*X+C)



$$O = K * X + C$$

Раздел 'Тригонометрические'

Функции прямого тригонометрического преобразования интерпретируют значение своего аргумента (значение входа **ARG** для блоков **SIN**, **COS** и **TAN**; отношение **DVD/DVS** для блока **_ATAN** и значение входа **IA** для блока **PDT**) как угол в радианах.

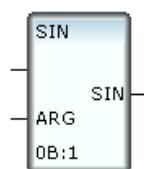
Блоки **ASIN**, **ACOS** и **ATAN** и **_ATAN** возвращают главное значение соответствующих функций в радианах:

$$-\frac{\pi}{2} \leq \arcsin x \leq \frac{\pi}{2};$$

$$-\frac{\pi}{2} \leq \arctg x \leq \frac{\pi}{2};$$

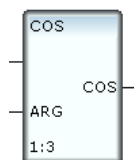
$$0 \leq \arccos x \leq \pi$$

Синус (SIN)



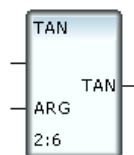
$$\text{SIN} = \sin(\text{ARG})$$

Косинус (COS)



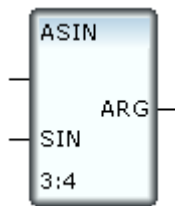
$$\text{COS} = \cos(\text{ARG})$$

Тангенс (TAN)



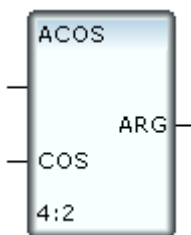
$$\text{TAN} = \text{tg}(\text{ARG})$$

Арксинус (ASIN)



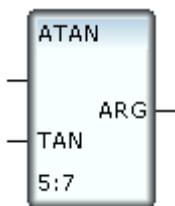
ARG = Arcsin(SIN)

Арккосинус (ACOS)



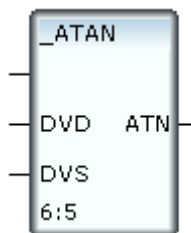
ARG = Arccos(COS)

Арктангенс (ATAN)

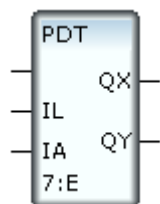


ARG = Arctg(TAN)

Арктангенс отношения (_ATAN)



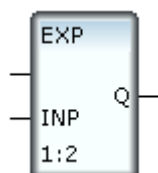
ATN = Arctg(DVD/DVS)

Преобразование полярных координат в декартовы (PDT)

$$QX = IL * \cos(IA) ; QY = IL * \sin(IA)$$

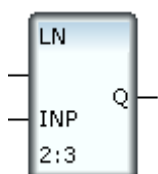
Раздел 'Алгебраические'

Экспонента (EXP)



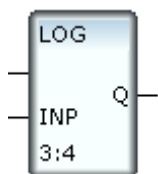
$$Q = e^{\text{INP}}$$

Натуральный логарифм (LN)



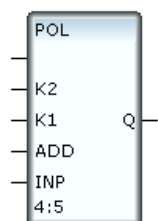
$$Q = \ln(\text{INP})$$

Десятичный логарифм (LOG)



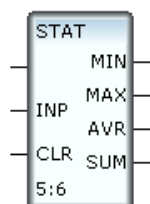
$$Q = \lg(\text{INP})$$

Квадратный трехчлен (POL)



$$Q = K2 \cdot \text{INP}^2 + K1 \cdot \text{INP} + \text{ADD}$$

Статистика (STAT)



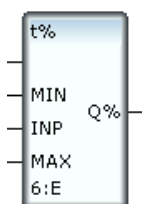
Этот блок определяет характеристики сигнала, поданного на вход **INP**.

При **CLR=0** на выходе **MIN** формируется минимальное значение анализируемого сигнала, на выходе **MAX** – максимальное, на выходе **AVR** – среднее.

Значение выхода **SUM** увеличивается на величину входа **INP** на каждом такте пересчета блока.

При подаче на вход **CLR** значения, отличного от 0, на всех выходах устанавливается значение, поданное на вход **INP**.

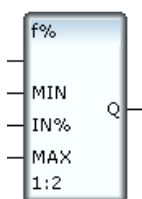
Перевод в проценты (t%)



Блок вычисляет отношение входной величины (**INP**) к заданному диапазону (в процентах).

$$Q = \frac{INP - MIN}{MAX - MIN} * 100\%$$

Перевод из процентов (f%)

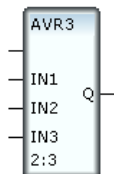


Блок вычисляет значение, заданное в процентах по отношению к установленному диапазону.

$$Q = \frac{(\text{MAX} - \text{MIN}) * \text{IN}\%}{100\%} + \text{MIN}$$

Значение входа **IN%** задается в процентах.

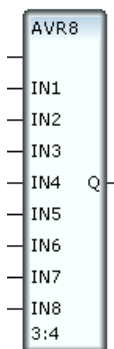
Среднее по трем точкам (AVR3)



$$Q = (\text{IN1} + \text{IN2} + \text{IN3}) / 3$$

Значение неопределенного входа принимается равным нулю.

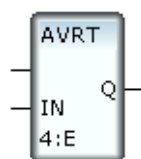
Среднее по восьми точкам (AVR8)



$$Q = \frac{1}{n} \sum_{i=1}^n \text{IN}_i$$

Значение неопределенного входа принимается равным нулю.

Скользящее среднее (AVRT)



Этот блок вычисляет среднее значение входа **IN** за четыре последних такта пересчета:

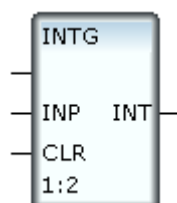
$$Q_k = \frac{1}{4} \sum_{i=k-3}^k IN_i$$

где

Q_k – текущее значение среднего (на k -ом такте пересчета блока).

IN_i – значение входа на i -ом такте пересчета.

Определенный интеграл (INTG)



Если $CLR=0$, то

$$INT_k = \Delta t * \sum_{i=1}^k \frac{INP_i + INP_{i-1}}{2}$$

где

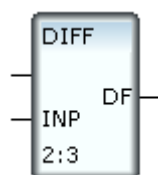
INT_k – текущее значение интеграла (на k -ом такте пересчета).

Δt – период пересчета блока в секундах.

INP_i – значение входа INP на i -ом такте пересчета блока (INP_0 – значение при запуске пересчета).

При подаче на вход CLR отличного от нуля значения a процесс интегрирования прерывается, при этом $INT = a - 1$.

Производная по двум точкам (DIFF)



$$DF_i = \frac{INP_i - INP_{i-1}}{\Delta t}$$

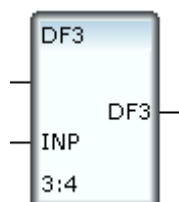
где

DF_i – текущее значение производной (на i -ом такте пересчета).

Δt – период пересчета блока в секундах.

INP_i – значение входа **INP** на i -ом такте пересчета блока (INP_0 – значение при запуске пересчета).

Производная по трем точкам (DF3)



$$DF3_i = \frac{1.5 * INP_i - 2 * INP_{i-1} + 0.5 * INP_{i-2}}{\Delta t}$$

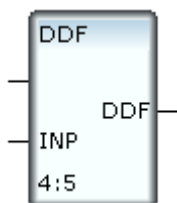
где

$DF3_i$ – текущее значение производной (на i -ом такте пересчета).

INP_i – значение входа **INP** на i -ом такте пересчета блока (INP_0 – значение при запуске пересчета).

Δt – период пересчета блока в секундах.

Вторая производная (DDF)



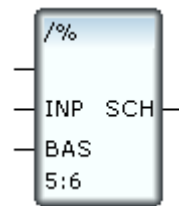
$$DDF_i = \frac{INP_i - 2 * INP_{i-1} + INP_{i-2}}{(\Delta t)^2}$$

где

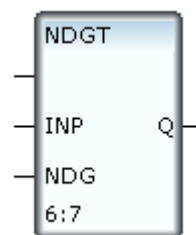
DDF_i – текущее значение второй производной (на i -ом такте пересчета).

INP_i – значение входа **INP** на i -ом такте пересчета блока (INP_0 – значение при запуске пересчета).

Δt – период пересчета блока в секундах.

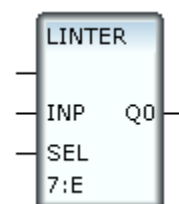
Нагрузка (%)

$$SCH = \frac{INP - BAS}{BAS} * 100\%$$

Округление (NDGT)

На выходе **Q** формируется округленное значение входа **INP**. Значение входа **NDG** задает точность округления:

- 2 – до сотен;
- 1 – до десятков;
- 0 – до целых;
- 1 – до первого разряда после запятой;
- 2 – до второго разряда после запятой;
- 3 – до третьего разряда после запятой;
- 4 – до четвертого разряда после запятой

Линейная интерполяция (LINTER)

Для корректной работы этого блока должен быть создан по крайней мере

один канал CALL OUTPUT с типом вызова **TableFunction** (9) (см. **Канал класса CALL**). Алгоритм работы блока зависит от атрибута **Параметр** канала CALL.

Параметр=0

Аргументы канала CALL задают в табличном виде некоторую функцию $y(x)$. Четные аргументы, начиная с нулевого, определяют абсциссы, последующие нечетные – соответствующие ординаты точек. Значения четных аргументов (абсцисс) должны монотонно возрастать. Номер таблицы задается начальным значением канала CALL.

Байт 0 неотрицательного значения входа **SEL** блока LINTER указывает номер используемой таблицы (1...32), байт 1 определяет алгоритм работы блока:

- если значение байта 1 равно 0:
 - если $INP = X_k$, то $Q0 = Y_k$ (X_k, Y_k – табличная точка);
 - если $INP < X_0$, то $Q0 = Y_0$ (здесь X_0 – наименьшее значение аргумента табличной функции);
 - если $INP > X_M$, то $Q0 = Y_M$ (здесь X_M – наибольшее значение аргумента табличной функции);
 - если INP принимает некоторое промежуточное значение между ближайшими к нему табличными X_k и X_{k+1} ($X_k < INP < X_{k+1}$), то блок выполняет линейную интерполяцию:

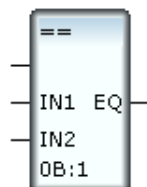
$$Q0 = Y_k + \frac{Y_{k+1} - Y_k}{X_{k+1} - X_k} (INP - X_k)$$

- если значение байта 1 равно 1, то целое неотрицательное значение **INP** интерпретируется как порядковый номер точки в таблице (начиная с 0) и выход блока принимает табличное значение абсциссы этой точки. Если **INP** задает несуществующую точку, значение выхода блока не изменяется;
- если значение байта 1 равно 2, то целое неотрицательное значение **INP** интерпретируется как порядковый номер точки в таблице (начиная с 0) и выход блока принимает табличное значение ординаты этой точки. Если **INP** задает несуществующую точку, значение выхода блока не изменяется;
- если значение байта 1 равно 3, то целое неотрицательное значение **INP** интерпретируется как порядковый номер аргумента канала CALL (начиная с 0) и выход блока принимает соответствующее табличное значение. Если **INP** задает несуществующий аргумент канала CALL, значение выхода блока не изменяется.

Во всех случаях значение выхода не изменяется, если в узле нет таблицы с номером, заданным **SEL**.

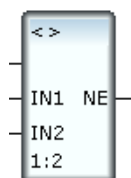
Раздел 'Функции сравнения'

Равенство (==)



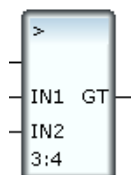
Если **IN1 = IN2**, то **EQ = 1 (TRUE)**, в противном случае **EQ = 0 (FALSE)**.

Неравенство (<>)



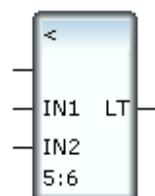
Если **IN1 <> IN2**, то **NE = 1 (TRUE)**, в противном случае **NE = 0 (FALSE)**.

Больше (>)



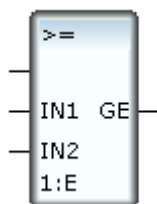
Если **IN1 > IN2**, то **GT = 1 (TRUE)**, в противном случае **GT = 0 (FALSE)**.

Меньше (<)



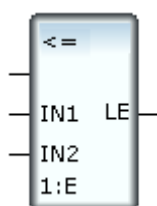
Если **IN1 < IN2**, то **LT = 1 (TRUE)**, в противном случае **LT = 0 (FALSE)**.

Больше или равно (\geq)



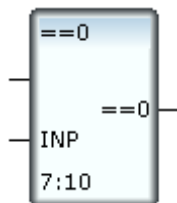
Если **IN1** \geq **IN2**, то **GE** = 1 (TRUE), в противном случае **GE** = 0 (FALSE).

Меньше или равно (\leq)



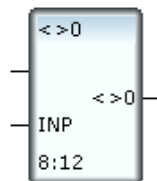
Если **IN1** \leq **IN2**, то **LE** = 1 (TRUE), в противном случае **LE** = 0 (FALSE).

Равенство нулю ($=0$)

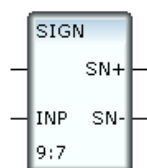


Если **INP** = 0, то блок возвращает 1 (TRUE), в противном случае – 0 (FALSE).

Неравенство нулю ($\neq 0$)

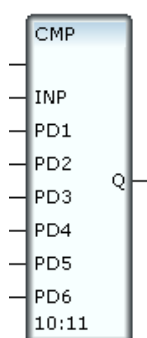


Если **INP** \neq 0, то блок возвращает 1 (TRUE), в противном случае – 0 (FALSE).

Знаковая функция (SIGN)

Если $INP \geq 0$, то $SN+ = 1$, а $SN- = 0$.

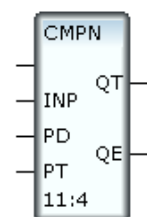
Если $INP < 0$, то $SN+ = 0$, а $SN- = 1$.

Анализ на равенство (CMP)

Блок сравнивает значение входа INP со значениями входов PD_k ($k=1,2,\dots,6$ – номер входа).

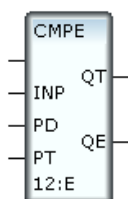
На выход Q передается наименьший из номеров входов PD_k , чьи значения равны INP .

Если значения всех PD_k отличны от INP , значение выхода не изменяется.

Анализ несовпадения (CMPN)

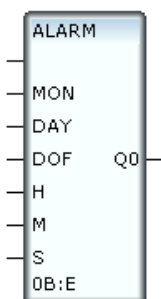
Если $INP \neq PD$, то QT увеличивается на 1 на каждом такте пересчета блока. Если $QT > PT$, то QE принимает значение 1, при этом увеличение значения QT прекращается, даже если $INP \neq PD$. Выходы QT и QE принимают значение 0 при любом изменении INP .

Анализ совпадения (CMPE)



Блок работает аналогично блоку **CMPN**. В отличие от **CMPN**, блок **CMPE** анализирует совпадение значения **INP** с уставкой, поданной на вход **PD**. Еще одним отличием является сброс выходов **QT** и **QE** в 0 при любом изменении как **INP**, так и **PD**.

Управление по астрономическому времени (ALARM)

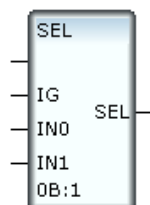


Данный блок формирует на выходе 1 при совпадении текущего астрономического времени с временем, заданным входами блока. Единица на выходе удерживается в течение одной секунды, затем выход обнуляется.

Вход **MON** задает номер месяца, **DAY** – день месяца, **DOF** – день недели (1 - 7, первый день недели - воскресенье), **H** – часы, **M** – минуты и **S** – секунды. Если хотя бы один из этих параметров не соответствует текущему астрономическому времени, **Q0** = 0. При формировании выходного сигнала не учитываются входы со значением -1.

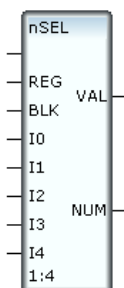
Раздел 'Функции выбора'

Выбор из двух (SEL)



$SEL = IN_k$, ($k = 0,1$), если $IG = k$.

Выбор из пяти (nSEL)

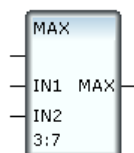


Выход **NUM** этого блока равен номеру одного из входов **I0...I4**, значение которого передается на выход **VAL**. Коммутацией управляют входы **REG** и **BLK**. Если **REG** = 0, на выход **VAL** передается наименьшее из входных значений, если **REG** = 1 – наибольшее.

Первые 5 битов числа, поданного на вход **BLK**, определяют участие входов в выборе. Если бит равен 0, соответствующий ему по номеру вход рассматривается, 1 – игнорируется.

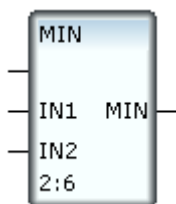
Если входом **BLK** заблокированы все 5 сигнальных входов, **NUM** = -2.

Выбор максимального (MAX)



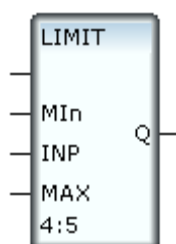
Этот блок возвращает большее из входных значений.

Выбор минимального (MIN)



Этот блок возвращает меньшее из входных значений.

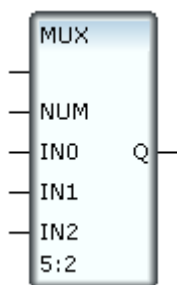
Ограничение (LIMIT)



Этот блок клипширует входной сигнал (вход **INP**), если его значение выходит за границы заданного диапазона:

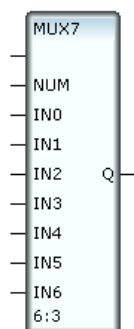
- Q = MAX**, если **INP > MAX**
- Q = MIN**, если **INP < MIN**
- Q = INP**, если **MIN <= INP <= MAX**

Выбор из трех (MUX)



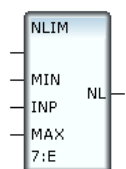
Q = IN_k, ($k = 0,1,2$), если **NUM = k**.

Если **NUM** принимает другое значение, то на текущем такте пересчета значение выхода **Q** не изменяется.

Выбор из семи (MUX7)

$Q = IN_k$, ($k = 0, 1 \dots 6$), если $NUM = k$.

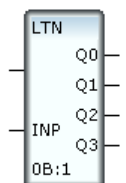
Если NUM принимает другое значение, то на текущем такте пересчета значение выхода Q не изменяется.

Интервал (NLIM)

$NL = 1$, если $INP > MAX$ (на вход INP подается анализируемое значение)

$NL = 0$, если $MIN \leq INP \leq MAX$

$NL = 2$, если $INP < MIN$

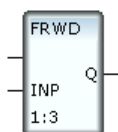
Запаздывание (LTN)

Данный блок реализует звено чистого запаздывания:

$$\begin{aligned} Q0_i &= INP_i; & Q1_i &= INP_{i-1}; & Q2_i &= INP_{i-2}; \\ Q3_i &= INP_{i-3}; \end{aligned}$$

где i – номер текущего такта пересчета блока.

Предсказание (FRWD)



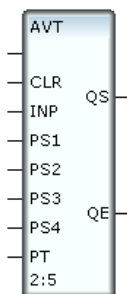
Этот блок реализует функцию экстраполяции входного значения по первой и второй производным. Выходу **Q** присваивается предполагаемое значение входа **INP** на следующем такте пересчета. Вычисление значения выхода осуществляется по следующей формуле:

$$Q_i = INP_i + \left(\frac{INP_i - INP_{i-1}}{\Delta t} + \frac{INP_i - 2 * INP_{i-1} + INP_{i-2}}{(\Delta t)^2} \Delta t \right) \Delta t =$$

$$= 3 * INP_i - 3 * INP_{i-1} + INP_{i-2}$$

где **i** – номер текущего такта пересчета.

Анализ и управление состоянием (AVT)



По результатам анализа значения, поданного на вход **INP**, блок последовательно изменяет состояние управляемого объекта.

При подаче 1 на вход **CLR** выходы блока обнуляются, и он прекращает свою работу.

При подаче 0 на вход **CLR** блок переходит в рабочий режим, при этом на один такт пересчета бит 0 выхода **QS** принимает значение 1.

На первом этапе работы блок сравнивает значения входов **INP** и **PS1**. Если **INP <> PS1** в течение **PT** тактов пересчета, биты 0 и 8 выхода **QE** принимают значение 1 (**QE=16#101**). Установившиеся значения выходов остаются неизменными до тех пор, пока значения **INP** и **PS1** не станут равными. Когда **INP=PS1**, бит 1 **QS** принимает значение 1 на один такт пересчета, а бит 1 **QS** принимает постоянное значение 1 (т.е. на один такт пересчета **QS** принимает значение 16#102, а затем – постоянное значение 16#100). Выход **QE** вначале обнуляется, а по окончании текущего интервала из **PT** тактов пересчета его биты 0 и 9 принимают постоянное значение 1 (**QE=16#201**). На этом первый этап работы блока завершается,

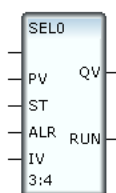
а значения выходов (**QS**=16#100 и **QE**=16#201) остаются неизменными до тех пор, пока не выполнится условие **INP=PS2**.

Следующие этапы аналогичны первому, т.е. когда **INP** становится равным **PS<k+1>**, где **k** = 1,2,3 – номер последнего завершеного этапа, **QS** изменяет свое значение: бит с номером (**k+1**) принимает значение 1 на один такт пересчета, а байт 1 принимает постоянное значение **k+1**. Выход **QE** вначале обнуляется, а по окончании текущего интервала из **PT** тактов пересчета его биты 0 и **k+9** (бит (**k+1**) байта 1) принимают постоянное значение 1 (по завершении последнего (четвертого) этапа и последующем истечении **PT** тактов пересчета **QE**=2).

Таким образом:

- в байте 1 **QS** формируется код нового состояния объекта (1,2,3,4);
- импульс в **i**-ом (**i**=1,2,3,4) бите **QS** указывает на завершение **i**-ого этапа;
- ненулевое значение **QE** индицирует превышение времени исполнения этапа, заданное **PT**. При этом:
 - номер установленного в 1 бита байта 1 **QE** равен номеру последнего завершеного этапа (после завершения четвертого этапа байт 1 **QE** принимает значение 0);
 - байт 0 **QE** равен 2 после завершения этапа 4 и 1 – после завершения любого другого этапа.

Управление выработкой (SEL0)



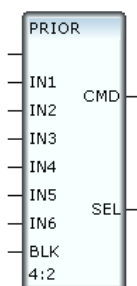
Этот блок управляет несколькими (до 8) однотипными агрегатами для получения заданной выработки.

На вход **PV** данного блока подается задание на выработку. Вход **ST** используется для ввода данных о состоянии агрегатов. Каждый его бит описывает состояние соответствующего агрегата (0 - выключен, 1 - включен). Вход **ALR** используется для указания работоспособности агрегатов. Каждый бит этого входа описывает соответствующий агрегат (0 - работоспособен, 1 - неисправен). На последний вход (**IV**) подается производительность одного агрегата.

На выходе **QV** формируется значение предполагаемой выработки. Оно вычисляется как сумма производительностей агрегатов, для которых сформированы команды на включение. Эти команды формируются на вы-

ходе **RUN**. Каждый бит данного выхода управляет соответствующим агрегатом (0 - выключить, 1 - включить). Формирование этих команд осуществляется с учетом текущего состояния и работоспособности агрегатов для обеспечения следующего условия: предполагаемая выработка должна быть не меньше заданной.

Выбор по приоритету (*PRIOR*)



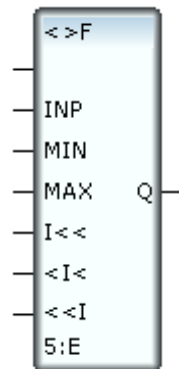
Данный блок имеет шесть входов для команд управления (**IN1...IN6**) и один вход блокирования команд (**BLK**).

Из входов **IN1...IN6** выбирается наименьший по номеру вход, который не заблокирован и имеет ненулевое значение. Значение такого входа передается на выход **CMD**, а его номер – на выход **SEL**.

Блокировать можно только входы **IN3...IN6**. Управляет блокировкой вход **BLK**. Его биты 0-3 блокируют команду 1, поданную на соответствующий вход, биты 4-7 – команду 2, биты 8-11 – команду 3. Младшему биту в каждой четверке соответствует вход **IN3**, а старшему – **IN6**. Блокирование осуществляется по равенству бита единице. Например, чтобы заблокировать команду 3, поданную на вход **IN4**, нужно подать на вход **BLK** значение 2^9 (512).

Назначение других битов **BLK**:

- Бит 12 (0x1000) – если 1: обнулить **CMD**;
- Бит 13 (0x2000) – если 1: обнулить **SEL**;
- Бит 14 (0x4000) –
- Бит 15 (0x8000) – заблокировать все команды по **IN1** и **IN2**;
- Бит 16 (0x10000) – если 1: заблокировать битами 8-11 команду -1 (вместо команды 3);
- Бит 17 (0x20000) – если 1: копировать 0 на выход при подаче 4 на соответствующий вход;
- Бит 18 (0x40000) – если 1: умножить **CMD** на 0x400 (для ZDV и KLP);
- Бит 19 (0x80000) – если 1: обнулить выходы и отключить блок.

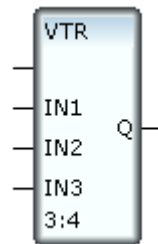
Управление по интервалу (<>F)

Данный блок позволяет формировать произвольные значения в зависимости от интервала, в который попадает анализируемое значение, поданное на вход **INP**.

Если **INP** < **MIN**, то **Q** = **I<<**

Если **MIN** <= **INP** <= **MAX**, то **Q** = **<I<**

Если **INP** > **MAX**, то **Q** = **<<I**

Измерение по трем точкам (VTR)

Значение выхода этого блока вычисляется по следующей формуле:

$$Q_i = (IN1 + IN2 + IN3 + Q_{i-1} - MIN - MAX) / 2$$

где

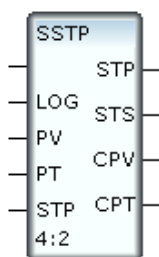
i – номер текущего такта пересчета;

IN<k> – значения входов (квазипостоянные величины);

MIN – минимальное из **IN1**, **IN2**, **IN3** и **Q_{i-1}**;

MAX – максимальное из **IN1**, **IN2**, **IN3** и **Q_{i-1}**.

Таким образом, выходное значение формируется плавно, без скачка.

Конечный автомат (SSTP)

Данный блок переводит управляемый объект в состояние, которое выбирает в соответствии со значениями своих входов.

Входы блока:

- LOG** – управляющий вход. Допустимые значения – 0 и 1;
- PT** – уставка таймера (задается в тактах пересчета);
- PV** – уставка счетчика событий. Под событием здесь понимается изменение значения входа **LOG** с 0 на 1;
- STP** – код следующего состояния. При равенстве этого параметра 0 код следующего состояния равен коду текущего состояния, увеличенному на единицу.

Выходы блока:

- CPT** – текущее значение таймера (в тактах пересчета). Выход **CPT** обнуляется при переходе в новое состояние;
- CPV** – текущее значение счетчика событий. Выход **CPV** обнуляется при переходе в новое состояние;
- STP** – код текущего состояния;
- STS** – код условия, по которому выбрано текущее состояние.

Переход в новое состояние, заданное входом **STP**, производится при выполнении любого из двух следующих условий:

- **CPV=PV**, т.е. если значение входа **LOG** изменилось с 0 на 1 **PV** раз;
- **CPT=PT**, т.е. если истекло заданное время.

Выход **STS** принимает следующие значения:

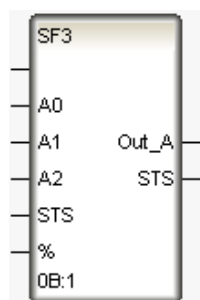
- 0 – если **LOG=0** и еще не произошло ни одного перехода;
- 1 – если **LOG=0**, а текущее состояние выбрано по условию **CPV=PV**;
- 2 – если **LOG=0**, а текущее состояние выбрано по условию **CPT=PT**;
- 16 – если **LOG=1** и еще не произошло ни одного перехода;

17 – если **LOG=1**, а текущее состояние выбрано по условию **CPV=PV**;

18 – если **LOG=1**, а текущее состояние выбрано по условию **CPT=PT**.

Подача отрицательного значения на вход **PV**, **PT** или **STP** останавливает работу блока и формирует на выходе **CPV**, **CPT** и **STP** соответственно значение **|PV|**, **|PT|** и **|вход STP|**. После снятия отрицательных значений со входов блок обрабатывает свои функции в соответствии с установленными значениями на выходах.

Выбор аналогового сигнала (SF3)



Биты 0-2 входа **STS** – достоверность соответственно значений **A0-A2** (0 – достоверно, 1 – недостоверно).

Если **A0-A2** недостоверны, **Out_A** не меняется и устанавливается бит 0 выхода **STS**.

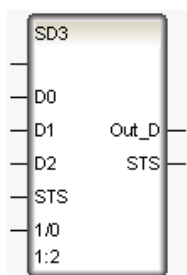
Если из **A0-A2** достоверно только одно значение, оно записывается в **Out_A**.

Если из **A0-A2** достоверны два значения, в **Out_A** записывается их среднее арифметическое.

Если **A0-A2** достоверны:

- если бит 8 входа **STS** не установлен, **Out_A=M**, где **M** – медиана (вычисляется по **A0-A2**). Кроме того, вычисляются величины $d_i = (A_i - M)/M$. При $d_i > \%$ устанавливаются соответственно биты 8, 9 и 10 выхода **STS**;
- если бит 8 входа **STS** установлен, блок работает по аналогичному алгоритму, но дополнительно вычисляется сумма значений **A0-A2**, из которой затем вычитаются значения, для которых $d_i > \%$. Скорректированная таким образом сумма делится на **N** (число значений, для которых $d_i < \%$), и полученный результат записывается в **Out_A**.

Если установлен бит 9 входа **STS**, значение битов 12-15 выхода **STS** индицирует **N**.

Выбор цифрового сигнала (SD3)

Биты 0-2 входа **STS** – достоверность соответственно значений **D0-D2** (0 – достоверно, 1 – недостоверно).

Если **D0-D2** недостоверны, **Out_D** не меняется и устанавливается бит 0 выхода **STS**.

Если из **D0-D2** достоверно только одно значение, оно записывается в **Out_D**.

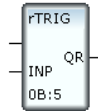
Если из **D0-D2** достоверны два значения (**D_k** и **D_m**):

- если бит 0 входа **1/0** равен 0, то **Out_D = D_k OR D_m**
- если бит 0 входа **1/0** равен 1, то **Out_D = D_k AND D_m**

Если **D0-D2** достоверны, то **Out_D = S = (D0 & D1) | (D0 & D2) | (D1 & D2)** (**S** равно 0 или 1 в зависимости от того, каких значений больше в **D0-D2**). Далее **S** сравнивается с **D0-D2**, при неравенстве устанавливаются соответственно биты 8-10 выхода **STS**. Если установлен бит 9 входа **1/0**, значение битов 12-15 выхода **STS** индицирует число значений, равных **S**.

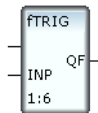
Раздел 'Триггеры и счетчики'

Импульс по переднему фронту (rTRIG)



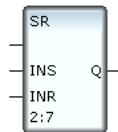
Этот блок формирует прямоугольный импульс длиной в один такт пересчета при изменении значения младшего разряда **INP** с 0 на 1.

Импульс по заднему фронту (fTRIG)



Выход **QF** принимает значение 1 на один такт пересчета при изменении значения входа **INP** с любого положительного на 0.

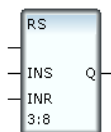
Триггер с приоритетом по установке (SR)



Выход **Q** изменяет свое значение с 0 на 1 при изменении значения входа **INS** с 0 на любое положительное (при этом значение входа **INR** может быть любым неотрицательным).

Для сброса выхода в 0 нужно подать 0 на вход **INS** и любое положительное значение на вход **INR**.

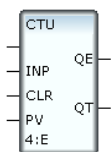
Триггер с приоритетом по сбросу (RS)



Выход **Q** изменяет свое значение с 0 на 1, если **INS** > 0 и **INR** = 0.

Для сброса выхода в 0 нужно подать любое положительное значение на вход **INR**.

Счетчик (CTU)



Функцией данного блока является подсчет количества тактов пересчета, в течение которых значение контролируемой величины (вход **INP**) было от-
лично от 0, и сравнение этого количества с заданной уставкой. Вход **CLR**
используется для смещения текущего значения счетчика (выход **QT**), вход
PV – для задания уставки, а выход **QE** показывает результат сравнения
значения счетчика с уставкой.

Ниже индекс *i* обозначает номер текущего такта пересчета блока.

При $CLR_i = 0$ и $QT_i < PV_i$:

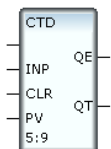
если $INP_i < 0$, то $QT_i = QT_{i-1} + 1$, в противном случае $QT_i = QT_{i-1}$

При $CLR_i > 0$:

$QT_i = CLR_i - 1$ и вне зависимости от PV_i и INP_i счетчик останавли-
вается.

Если $QT_i > PV_i$, то $QE_i = 1$ (при этом счетчик останавливается), в против-
ном случае $QE_i = 0$.

Обратный счетчик (CTD)



Этот блок аналогичен блоку **CTU**, но реализует обратный счетчик.

Контролируемая величина подается на вход **INP**. Вход **CLR** используется
для смещения текущего значения счетчика (выход **QT**), вход **PV** – для за-
дания начального значения счетчика, выход **QE** показывает результат
сравнения значения счетчика с 0.

Ниже индекс *i* обозначает номер текущего такта пересчета блока.

При $CLR_i = 0$:

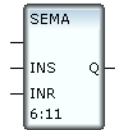
если $INP_i < 0$, то $QT_i = QT_{i-1} - 1$, в противном случае $QT_i = QT_{i-1}$

При $CLR_i < 0$:

$QT_i = PV_i - (CLR_i - 1)$ и вне зависимости от INP_i счетчик остано-
вливается

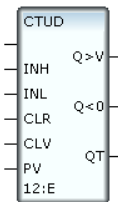
Если $QT_i < 0$, то $QE_i = 1$ (при этом счетчик останавливается), в противном случае $QE_i = 0$.

Семафор (SEMA)



Данный блок аналогичен триггеру с приоритетом по установке (блок **SR**). Отличие заключается в том, что установка 1 на выходе осуществляется с задержкой на один такт. Сброс выхода в 0 осуществляется на том же такте, на котором одновременно $INR < 0$ и $INS = 0$.

Комбинированный счетчик (CTUD)



Этот блок сочетает в себе функции нарастающего и убывающего счетчиков.

Наличие ненулевого значения на входе **INH** увеличивает значение счетчика на 1 (выход **QT**), наличие ненулевого значения на входе **INL** – уменьшает. Вход **INH** имеет более высокий приоритет, чем **INL**, поэтому при ненулевых значениях обоих входов блок работает как нарастающий счетчик.

Вход **PV** задает максимальное значение для нарастающего счетчика и начальное значение для убывающего счетчика.

Ниже индекс *i* обозначает номер текущего такта пересчета блока.

Нарастающий счетчик:

При $CLR_i = 0$ и $QT_i < PV_i$:

если $INH_i < 0$, то $QT_i = QT_{i-1} + 1$, в противном случае $QT_i = QT_{i-1}$

При $CLR_i > 0$:

$QT_i = 0$ и счетчик останавливается.

Если $QT_i > PV_i$, то $Q>V_i = 1$ (при этом счетчик останавливается), в противном случае $Q>V_i = 0$.

Убывающий счетчик:

При $CLV_i = 0$:

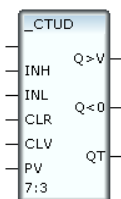
если $INL_i < 0$, то $QT_i = QT_{i-1} - 1$, в противном случае $QT_i = QT_{i-1}$

При $CLV_i < 0$:

$QT_i = PV_i$ и счетчик останавливается

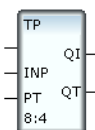
Если $QT_i < 0$, то $Q<0_i = 1$ (при этом счетчик останавливается), в противном случае $Q<0_i = 0$.

Комбинированный счетчик 2 (*_CTUD*)



Этот блок аналогичен блоку **CTUD** за одним исключением: он останавливается, если одновременно $INH < 0$ и $INL < 0$.

Импульс произвольной длительности (*TP*)

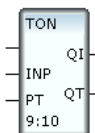


Этот блок предназначен для формирования импульсов единичной амплитуды и заданной длительности.

Импульс формируется на выходе **QI** при изменении значения входа **INP** с 0 на любое положительное. Длительность импульса в тактах пересчета задает вход **PT**. Выход **QT** индицирует число тактов, прошедших с начала формирования импульса.

Перед формированием очередного импульса нужно обнулить выход **QT** (для этого нужно подать 0 на вход **INP**).

Задержка на включение (*TON*)

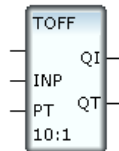


При изменении значения входа **INP** с 0 на любое ненулевое выход **QI** принимает значение 1 с задержкой в **PT** тактов пересчета, при этом выход

QT индицирует число тактов, прошедших с момента изменения **INP**.

Чтобы обнулить выходы **QI** и **QT**, нужно подать 0 на вход **INP**.

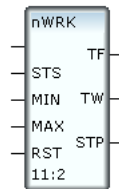
Задержка на выключение (TOFF)



Если $INP < 0$, то $QI = 1$, $QT = 0$.

При изменении значения входа **INP** с ненулевого на 0 выход **QI** принимает значение 0 с задержкой в **PT** тактов пересчета, при этом выход **QT** индицирует число тактов, прошедших с момента изменения **INP**.

Сторожевой таймер мотора (nWRK)



Этот блок контролирует время работы устройства типа «двигатель», управляемого блоком **MOTOR**. Блок **nWRK** может быть использован для выключения мотора через определенный промежуток времени.

Вход **STS** блока **nWRK** нужно соединить с выходом **STS** блока **MOTOR**. При $STS=1$ или 17 (мотор работает) увеличивается значение выходов **TF** и **TW**. Эти выходы показывают соответственно общее время работы мотора и время работы с момента последнего включения. Единицы измерения времени задаются значением битов 4-7 входа **RST**: 0 – дни, 1 – часы, 2 – минуты, 3 – секунды.

Если бит 0 **RST** равен 1, обнуляются выходы, указанные последующим битами при их равенстве 1: бит 1 – выход **TF**, бит 2 – выход **TW**, бит 3 – выход **STP**.

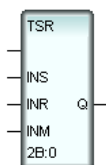
Если значение выхода **TW** (время работы с момента последнего включения) превышает значение входа **MAX**, то выходу **STP** присваивается значение 1. Этот сигнал может использоваться для выключения мотора.

Если $STP=0$, **TW** принимает значение 0 на том же такте пересчета, на котором **STS** становится отличным от 1 или 17.

Если $STP=1$, а **STS** становится отличным от 1 или 17 (2, 18, 10 или 26), то

STP=0 на этом же такте пересчета, а **TW** принимает значение 0 спустя **MIN** (если **STS** принимает другие значения, то **STP=0** и **TW=0** на этом же такте пересчета). Значение входа **MIN** не должно превышать 127.

32 триггера SR (TSR)



Соответствующие биты 32-разрядных **INS**, **INR** и **Q** образуют 32 триггера с приоритетом по установке (см. описание блока **SR**).

Вход **INM** – 32-разрядная маска используемых триггеров (0 – триггер используется, 1 – не используется).

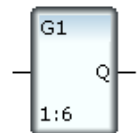
Раздел 'Генераторы'

Меандр (G01)



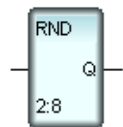
Данный блок генерирует прямоугольный сигнал с максимальным значением 1.

Бегущая единица (G1)



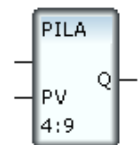
При работе этого генератора его 8-битовый выход последовательно принимает значения $0, 2^0, 2^1, \dots, 2^7, 0, 2^0$ и т.д. Значение выхода изменяется на каждом такте пересчета.

Случайная величина в диапазоне [0, 1] (RND)



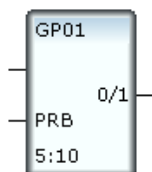
Этот блок генерирует случайную величину с равномерным законом распределения в диапазоне $[0, 1]$.

Пилообразный сигнал (PILA)



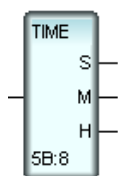
Данный блок генерирует пилообразный сигнал с максимальным значением, задаваемым входом **PV**. На каждом такте пересчета выход увеличивается на 1.

Единица с заданной вероятностью (GP01)



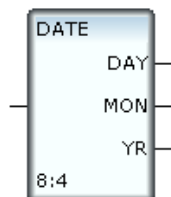
На выходе этого функционального блока генерируется 0 или 1, причем вероятность генерации единицы задается значением входа **PRB**. На вход **PRB** подается целое число в диапазоне от 0 до 1000. Этим границам соответствуют значения вероятности 0 и 1.

Астрономическое время (TIME)



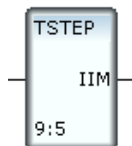
Значение выхода **S** этого блока равно текущей астрономической секунде, **M** - минуте, **H** - часу.

Астрономическая дата (DATE)

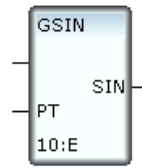


На выходе **DAY** этого блока генерируется текущее значение дня месяца, на выходе **MON** – номер месяца, а на выходе **YR** – текущий год.

Период вызова программы (TSTEP)



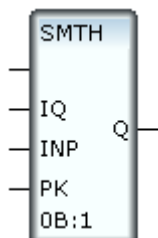
Этот блок измеряет период вызова программы в миллисекундах.

Синусоидальный сигнал (GSIN)

Этот функциональный блок генерирует синусоидальный сигнал единичной амплитуды. Период колебаний (в секундах) задается значением входа **PT**.

Раздел 'Управление'

Экспоненциальное сглаживание (SMTH)

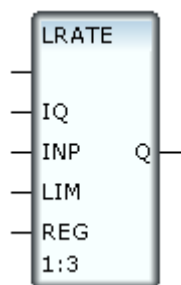


Этот блок выполняет суммирование с весом:

$$Q = IQ * (1 - 1/PK) + INP/PK$$

Если выход **Q** соединен со входом **IQ**, блок выполняет экспоненциальное сглаживание сигнала, поданного на вход **INP**. При этом коэффициент сглаживания (вход **PK**) должен быть больше 1.

Ограничение скорости (LRATE)



Этот блок с соединенными выходом **Q** и входом **IQ** ограничивает скорость изменения сигнала, поданного на вход **INP**. Максимально допустимое изменение сигнала задается входом **LIM**. Выходное значение формируется следующим образом.

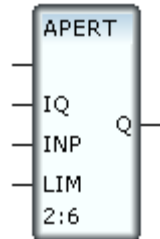
```
IF |INP-IQ|>LIM THEN Q:=IQ+SIGN(INP-IQ)*LIM
ELSE Q:=INP
```

Вход **REG** задает режим работы блока:

- 0 – ограничение скорости в соответствии с описанным выше алгоритмом;
- 1 – запрет увеличения;

- 2 – запрет уменьшения;
- 3 – фиксация значения выхода.

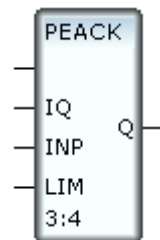
Апертура (APERT)



Этот блок с соединенными выходом **Q** и входом **IQ** фильтрует небольшие изменения (например, шумы дискретизации) сигнала, поданного на вход **INP**. Величина порога задается входом **LIM**. Алгоритм формирования выходного значения выглядит следующим образом.

```
IF |INP-IQ|<LIM THEN Q:=IQ
ELSE Q:=INP
```

Фильтрация пиков (PEACK)

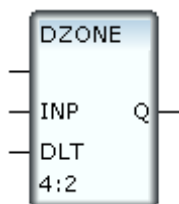


Этот блок с соединенными выходом **Q** и входом **IQ** фильтрует случайные броски сигнала, поданного на вход **INP**. Максимально допустимое изменение сигнала задается входом **LIM**. Выходное значение формируется следующим образом (*i* обозначает текущий такт пересчета).

```
IF |INPi-IQi|<=LIM THEN Qi:=INPi
ELSE IF |INPi-1 - IQi-1|<=LIM THEN Qi:=IQi
ELSE Qi:=INPi
```

Таким образом, значение входа **INP** передается на выход, если его изменение на текущем такте пересчета не превысило **LIM**, в противном случае выход не изменяется. Если на следующем такте пересчета разность **INP** и **Q** опять превысила **LIM**, то на выход передается значение **INP**.

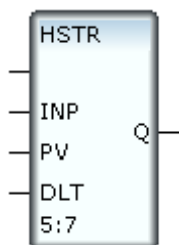
Зона нечувствительности (DZONE)



Данный блок работает как блок **APERT** с **IQ = 0**:

```
IF |INP| < DLT THEN Q := 0
ELSE Q := INP
```

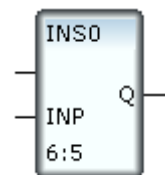
Гистерезис (HSTR)



Функцией данного блока является формирование единичного выходного сигнала при переходе входного значения (**INP**) через заданную границу (**PV**) с учетом значения гистерезиса (**DLT**).

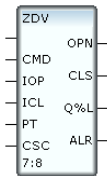
```
IF INP > PV + DLT THEN Q := 1
IF INP < PV - DLT THEN Q := 0
```

Вставка нуля (INS0)



Этот блок воспроизводит на выходе входное значение. Однако при каждом изменении входного значения величина выхода обнуляется на один такт и только на следующем становится равной входу. Как следствие, выходное значение всегда равно нулю, если входной сигнал изменяется на каждом такте пересчета.

Управление задвижкой (ZDV)



Данный блок управляет устройством типа «задвижка». Он анализирует следующие цифровые сигналы (0 или 1): сигналы концевых выключателей открытия, закрытия и муфты, сигнал электрической части (привода) и сигнал о наличии электрического напряжения (например, на соленоиде).

Для управления блоком могут использоваться как потенциальные, так и импульсные команды; в последнем случае команда обрабатывается по переднему фронту импульса.

Вход **CMD** предназначен для подачи команд управления. Потенциальные команды:

- 0 – остановить;
- 1 – открыть;
- 2 – закрыть;
- 17 – открыть и сбросить ошибки;
- 18 – закрыть и сбросить ошибки;
- 20 – остановить и сбросить ошибки.

Импульсные команды:

- импульс в бите 0 **CMD** – открыть;
- импульс в бите 1 **CMD** – закрыть;
- импульс в бите 2 **CMD** – остановить.

При одновременной подаче импульсов в биты **CMD**: **CMD**=3 – закрыть, **CMD**=5 или 7 – остановить.

При выполнении команд открытия/закрытия на выходе **OPN** формируется сигнал открытия (бит 0 равен 1), на выходе **CLS** – сигнал закрытия (бит 0 равен 1).

На вход **IOP** подаются следующие сигналы:

- бит 0 (0x1) – сигнал концевого выключателя открытия (если бит 6 **CSC** не установлен: 1 – открыто полностью, 0 – закрыто или открыто не полностью);
- бит 3 или 8 (0x8 или 0x100) – сигнал электрической части (привода) (0 – не работает, 1 – работает);
- бит 2 или 9 (0x4 или 0x200) – сигнал о наличии напряжения: 0 – норма, 1 – авария.

На вход **ICL** подаются следующие сигналы:

- бит 0 (0x1) – сигнал концевого выключателя закрытия (если бит 6 **CSC** не установлен: 1 – закрыто полностью, 0 – открыто или закрыто не полностью);
- бит 1 (0x2) – сигнал концевого выключателя муфты (если бит 6 **CSC** не установлен: 1 – закрыто полностью, 0 – открыто или закрыто не полностью);
- биты 3 и 8 (0x8 и 0x100) – аналоги битов 3 и 8 **IOP**;
- бит 2 или 9 (0x4 или 0x200) – аналоги битов 2 и 9 **IOP**.

Если на **ICL** ничего не подано, то $\text{ICL} = \text{IOP} \& 6$.

Вход **PT** используется для задания времени хода задвижки (в секундах). Его значение должно быть немного больше реального времени хода. По значению **PT** и времени, прошедшему с момента подачи команды, блок вычисляет текущее положение задвижки и выводит полученное значение на выход **Q%L**:

$$Q\%L_n = Q\%L_0 \pm 100 * (n-1) * t / PT$$

где $Q\%L_0$ – положение задвижки при подаче команды, $Q\%L_n$ – положение задвижки спустя n тактов пересчета после подачи команды, t – период пересчета в секундах, знак «+» соответствует закрытию, знак «-» – открытию (т.е. выход индицирует процент закрытия задвижки). В приведенной формуле учтено, что при подаче команды блок останавливается на 1 такт для сброса управляющего сигнала на выходе **OPN** или **CLS** (даже если этот сигнал не сформирован).

Ко времени хода добавляется значение, заданное ключом **VLVADDM**=<число секунд> в файле *.cnf.

Сигналы конечных выключателей имеют более высокий приоритет по сравнению с вычисленным положением задвижки, т.е. если при открытии (закрытии) включился концевой выключатель открытия (концевые выключатели закрытия и муфты), блок **ZDV** считает задвижку полностью открытой (закрытой) и $Q\%L=0$ (100) на следующем такте пересчета.

Установленные в 1 биты входа **CSC** задают следующие режимы работы блока:

Режим вычисляется как результат побитового сложения (OR) **CSC** и значения ключа **VLVSTSM**=<число> в файле *CNF.

- бит 0 (0x1) – блокирование контроля концевого выключателя открытия;
- бит 1 (0x2) – блокирование контроля концевого выключателя закрытия;
- бит 2 (0x4) – блокирование контроля концевого выключателя муфты;

- бит 3 (0x8) – принудительное присваивание выходу **Q%L** значения 0 («открыто») при выходе из аварийной ситуации (т.е. при установленном бите 5);
- бит 4 (0x10) – принудительное присваивание выходу **Q%L** значения 100 («закрыто») при выходе из аварийной ситуации (т.е. при установленном бите 5);
- бит 5 (0x20) – выход из аварийной ситуации. При возникновении аварийной ситуации при открытии/закрытии соответствующие выходы (**OPN** и **CLS**) принимают значение 0 (процедура прерывается), и блок **ZDV** останавливается. Если в режиме остановки ошибку устранить (т.е. задать корректное сочетание значений входов), выполнение процедуры не возобновится, и признак аварии, сформированный на выходе **ALR**, сохранится. Для сброса признака аварии и возобновления выполнения процедуры нужно установить данный бит в 1. При установке бита 5 возможна одновременная установка бита 3 (0x8) или бита 4 (0x10) для выполнения соответствующих функций (одновременная установка битов 3-5 равнозначна установке одного бита 5). При выходе из аварийной ситуации:
 - если задвижка была открыта/закрыта по времени, а затем остановлена при закрытии/открытии, то автоматически **Q%L=50**;
 - если задвижка была закрыта по времени, а затем возникла ошибка 4, то автоматически **Q%L= %sdv** (**%sdv** – процент закрытия, соответствующий таймауту концевого выключателя);
 - если задвижка была открыта по времени, а затем возникла ошибка 5, то автоматически **Q%L= 100-%sdv**;
 - после ошибки 6 или 2 – задвижка автоматически устанавливается в положение «открыто»;
 - после ошибки 7 или 3 – задвижка автоматически устанавливается в положение «закрыто»;
 - после ошибки 10 – автоматически **Q%L=Q%L+2**;
 - после ошибки 11, 12 или 13 – автоматически **Q%L=Q%L-2**;
 - после других ошибок – попытка (1 такт) идентификации положения задвижки;
- бит 6 (0x40) – инверсные сигналы концевых выключателей открытия, закрытия и муфты;
- бит 7 (0x80) – перевод в режим отслеживания состояния задвижки по концевым выключателям (аварийные ситуации не контролируются). Этот режим индицируется значением 15 байта 1 (0x0F00) выхода **ALR**;
- бит 8 (0x100) – см. ниже описание ошибки 14;
- бит 9 (0x200) – импульсные управляющие сигналы открытия (бит 0 **OPN**), закрытия (бит 0 **CLS**) и остановки (бит 2 или 9 **OPN**), длина импульса определяется битами 12-14. При отключении концевой

го выключателя соответствующий импульс сбрасывается;

- бит 10 (0x400) – блокирование команды открытия;
- бит 11 (0x800) – блокирование команды закрытия;

Если команда открытия (закрытия) блокируется битом 10 (11) **STS** в момент выполнения, задвижка останавливается.

- биты 12-14 (маска 0x7000) – эти биты задают длину управляющего импульса. Длина импульса в секундах рассчитывается по следующей формуле:

$$0.5 + [(CSC \& 0x7000) \gg 12] + K$$

K задается с помощью ключа **VLVAIMP** в файле *.cnf. По умолчанию **K** = 2с, эта константа не может превышать время, соответствующее 31 такту пересчета блока;

- бит 15 (0x8000) – 1 – управляющий сигнал остановки вырабатывается в бите 2 **OPN**, 0 – в бите 9 **OPN**. В режиме импульсных управляющих сигналов (установлен бит 9):
 - генерация импульса остановки зависит от бита 22;
 - в течение импульса остановки команды открытия/закрытия игнорируются;
- бит 16 (0x10000) – индикация процента открытия задвижки вместо процента закрытия на выходе **Q%L**;
- бит 17 (0x20000) – если задвижка не находится в одном из крайних положений, выход **Q%L** индицирует время (в секундах), оставшееся до закрытия/открытия. В крайних положениях задвижки **Q%L=0**;
- бит 18 (0x40000) – запрет сброса сигнала на **OPN/CLS** (например, для задвижки, которая должна удерживаться в крайнем положении);
- бит 20 (0x100000) – при наличии сигнала концевого выключателя закрытия наличие сигнала муфты не анализируется (считается, что он есть);
- бит 21 (0x200000) – разрешение анализа сигнала электрической части (привода). Наличие сигнала анализируется спустя 1 такт пересчета после подачи команды, а если задвижка находится в интервале (**%s_{dv}**, 100-**%s_{dv}**) – спустя 2 такта;
- бит 22 (0x400000) – в режиме импульсных управляющих сигналов (установлен бит 9) – разрешение генерации управляющего импульса остановки в случае диагностирования положения ОТКРЫТО/ЗАКРЫТО по концевому выключателю.

Байт 0 (0xFF) выхода **ALR** (этот байт показывает состояние задвижки) может принимать следующие значения:

- 1 – открывается;
- 2 – закрывается;

- 4 – открыта;
- 8 – закрыта;
- 17 (0x11) – остановка при открытии;
- 18 (0x12) – остановка при закрытии;
- 33 (0x21) – открывается в пределах таймаута концевого выключателя (*);
- 34 (0x22) – закрывается в пределах таймаута концевого выключателя;
- 49 (0x31) – остановка при открытии в пределах таймаута концевого выключателя;
- 50 (0x32) – остановка при закрытии в пределах таймаута концевого выключателя.

Байт 1 (0xFF00) выхода **ALR** указывает на возникновение аварийных ситуаций (аварийные ситуации не контролируются, если они связаны с сигналами, контроль которых заблокирован):

- 0 – норма;
- 1 – одновременно присутствуют сигналы конечных выключателей открытия и закрытия;
- 2 – при открытии не отключился концевой выключатель закрытия или муфты по истечении времени, заданного с помощью блока **SdV** (*);
- 3 – при закрытии не отключился концевой выключатель открытия по истечении времени, заданного с помощью блока **SdV** (*);
- 4 – остановка при открытии по времени (истекли **PT** секунд, а сигнала концевого выключателя открытия нет);
- 5 – остановка при закрытии по времени (истекли **PT** секунд, а сигнала концевого выключателя закрытия и/или муфты нет);
- 6 – остановка (возможно, задвижка закрыта внешними средствами);
- 7 – остановка (возможно, задвижка открыта внешними средствами);
- 8 – наличие сигнала электрической части (привода) при неподвижной задвижке;
- 9 – нет напряжения;
- 10 – задвижка открыта, но сигнала концевого выключателя открытия нет;
- 11 – задвижка закрыта, но сигнала концевого выключателя закрытия нет;
- 12 – при закрытой задвижке пропал сигнал концевого выключателя муфты, а сигнал концевого выключателя закрытия остался;
- 13 – при закрытой задвижке пропали сигналы конечных выключателя

телей закрытия и муфты.

- 14 – задвижка движется, а сигнала электрической части (привода) нет. В этой ситуации: если бит 8 (0x100) **CSC** не установлен, сигнал открытия/закрытия не сбрасывается (считается, что задвижка движется), в противном случае сигнал сбрасывается;
- 15 – это значение указывает, что блок находится в режиме отслеживания.

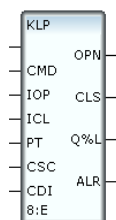
(*) Таймаут (время ожидания отключения) концевого выключателя в отсутствие блока **SdV** равно $0.1 * PT$ секунд.

Порядок диагностики аварийных ситуаций (при обнаружении ошибки дальнейшая диагностика не выполняется; если анализ соответствующего концевого выключателя заблокирован, пункт пропускается):

- задвижка остановлена, последнее положение – «открыто»:
 - сигнал напряжения;
 - сигналы конечных выключателей открытия и закрытия одновременно;
 - внешнее закрытие;
 - сигнал электрической части;
 - сигнал концевого выключателя открытия;
- задвижка остановлена, последнее положение – «закрыто»:
 - сигнал напряжения;
 - сигналы конечных выключателей открытия и закрытия одновременно;
 - инверсия сигналов конечных выключателей открытия и закрытия (внешнее открытие);
 - сигнал электрической части;
 - сигнал концевого выключателя закрытия;
 - сигнал концевого выключателя муфты;
 - сигналы конечных выключателей закрытия и муфты одновременно;
- задвижка открывается, время хода истекло:
 - сигнал напряжения;
 - сигнал концевого выключателя открытия;
- задвижка открывается из положения «закрыто» в течение времени, которое меньше времени ожидания отключения концевого выключателя закрытия, сигнал электрической части отсутствует:
 - сигнал напряжения;
 - генерируется ошибка 14, после чего проверяется отключение концевого выключателя закрытия. Если в требуемый момент времени нет отключения концевого выключателя закрытия, генерируется ошибка 2;

- задвижка закрывается, время хода истекло:
 - сигнал напряжения;
 - сигнал концевого выключателя закрытия;
- задвижка закрывается из положения «открыто» в течение времени, которое меньше времени ожидания отключения концевого выключателя открытия, сигнал электрической части отсутствует:
 - сигнал напряжения;
 - генерируется ошибка 14, после чего проверяется отключение концевого выключателя открытия. Если в требуемый момент времени нет отключения концевого выключателя открытия, генерируется ошибка 3.

Управление клапаном (KLP)



Блок **KLP** представляет собой модификацию блока **ZDV** для управления устройством типа «регулирующий клапан». Блок анализирует следующие цифровые сигналы (0 или 1): сигналы конечных выключателей открытия и закрытия, сигнал электрической части (привода) и сигнал о наличии электрического напряжения. Кроме того, блок анализирует аналоговый сигнал в диапазоне 0-100, показывающий реальное положение клапана (0 – полностью открыт, 100 – полностью закрыт).

Вход **CMD** задает направление движения клапана: положительная величина – открытие, отрицательная – закрытие. При выполнении этих команд на выходе **OPN** формируется сигнал открытия (значение 1), на выходе **CLS** – сигнал закрытия (значение 1). В режиме отслеживания неотрицательное значение входа **CMD** (0-100) задает положение, которое должен занять клапан.

Вход **PT** используется для задания времени хода клапана (в секундах). Ко времени хода добавляется значение, заданное ключом **KLPADDM**=<число секунд> в файле *.cnf. Если контроль положения клапана блокирован, блок вычисляет текущее положение клапана и выводит полученное значение на выход **Q%L** (аналогично **ZDV**).

На вход **CDI** подается реальное положение клапана (аналоговый сигнал в диапазоне 0-100, соответствующий проценту закрытия).

Если контроль положения клапана не блокирован, блок не

вычисляет положение, и $Q\%L=CDI$.

На вход **IOP** подаются следующие сигналы:

- бит 0 (0x1) – сигнал концевого выключателя открытия (1 – открыто полностью, 0 – закрыто или открыто не полностью);
- бит 3 или 8 (0x8 или 0x100) – сигнал электрической части (привода) (0 – не работает, 1 – работает);
- бит 2 или 9 (0x4 или 0x200) – сигнал о наличии напряжения: 0 – норма, 1 – авария).

На вход **ICL** подаются следующие сигналы:

- бит 0 (0x1) – сигнал концевого выключателя закрытия (1 – закрыто полностью, 0 – открыто или закрыто не полностью);
- биты 3 и 8 (0x8 и 0x100) – аналоги битов 3 и 8 **IOP**;
- бит 2 или 9 (0x4 или 0x200) – аналоги битов 2 и 9 **IOP**.

Если на **ICL** ничего не подано, то $ICL = IOP \& 6$.

Аналогично **ZDV**, сигналы конечных выключателей имеют более высокий приоритет по сравнению с данными о положении клапана (как вычисленными, так и полученными от устройства).

Установленные в 1 биты входа **CSC** соответствуют следующим режимам работы блока:

Режим вычисляется как результат побитового сложения (OR) **CSC** и значения ключа $KLPSTSM=<число>$ в файле *CNF.

- бит 0 (0x1) – блокирование контроля концевого выключателя открытия;
- бит 1 (0x2) – блокирование контроля концевого выключателя закрытия;
- бит 2 (0x4) – блокирование контроля положения клапана;
- бит 3 (0x8) – принудительное присваивание выходу **Q%L** значения 0 («открыто») при выходе из аварийной ситуации (т.е. при установленном бите 5);
- бит 4 (0x10) – принудительное присваивание выходу **Q%L** значения 100 («закрыто») при выходе из аварийной ситуации (т.е. при установленном бите 5);
- бит 5 (0x20) – выход из аварийной ситуации. При возникновении аварийной ситуации при открытии/закрытии соответствующие выходы (**OPN** и **CLS**) принимают значение 0 (процедура прерывается), и блок **KLP** останавливается. Если в режиме остановки ошибку устранить (т.е. задать корректное сочетание значений входов), выполнение процедуры не возобновится, и признак аварии, сформированный на выходе **ALR**, сохранится. Для сброса признака аварии

и возобновления выполнения процедуры нужно установить данный бит в 1. При установке бита 5 возможна одновременная установка бита 3 (0x8) или бита 4 (0x10) для выполнения соответствующих функций (одновременная установка битов 3-5 равнозначна установке одного бита 5);

- бит 6 (0x40) – компенсировать инерционность. В этом режиме при реверсе к вычисленному положению добавляется/вычитается половина величины, соответствующей времени ожидания отключения концевого выключателя (т.е. 5% в отсутствие **SdV**);
- бит 7 (0x80) – перевод в режим отслеживания состояния клапана по конечным выключателям (аварийные ситуации не контролируются). Этот режим индицируется значением 15 байта 1 (0x0F00) выхода **ALR**;
- бит 10 (0x400) – блокирование команды открытия;
- бит 11 (0x800) – блокирование команды закрытия.

Если команда открытия (закрытия) блокируется битом 10 (11) **STS** в момент выполнения, клапан останавливается.

- бит 12 (0x1000) – если бит 12 равен 1, клапан переходит в положение, заданное неотрицательным значением **CMD** (режим отслеживания). Скорость перехода определяется **PT**;
- бит 16 (0x10000) – индикация процента открытия вместо процента закрытия на выходе **Q%L**;
- бит 21 (0x200000) – разрешение анализа сигнала электрической части (привода).

Байт 0 (0xFF) **ALR** (этот байт показывает состояние клапана) может принимать следующие значения:

- 1 – открывается;
- 2 – закрывается;
- 4 – открыт;
- 8 – закрыт;
- 17 – остановка при открытии (16#11);
- 18 – остановка при закрытии (16#12);

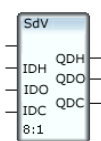
Байт 1 (0xFF00) **ALR** указывает на возникновение аварийных ситуаций (аварийные ситуации не контролируются, если они связаны с сигналами, контроль которых заблокирован):

- 0 – норма;
- 1 – одновременно установлены в 1 сигналы конечных выключателей открытия и закрытия;
- 2 – при открытии не отключился концевой выключатель закрытия по истечении времени, заданного с помощью блока **SdV** (*);

- 3 – при закрытии не отключился концевой выключатель открытия по истечении времени, заданного с помощью блока **SdV** (*);
- 4 – остановка при открытии по времени (истекли **PT** секунд, а сигнала концевого выключателя открытия нет);
- 5 – остановка при закрытии по времени (истекли **PT** секунд, а сигнала концевого выключателя закрытия нет);
- 6 – остановка при возникновении ошибки 2 (возможно, клапан закрыт внешними средствами);
- 7 – остановка при возникновении ошибки 3 (возможно, клапан открыт внешними средствами);
- 8 – наличие сигнала электрической части (привода) при неподвижном клапане;
- 9 – отклонение вычисленного положения клапана от реального (**Q%L** от **CDI**) превышает максимально допустимое значение, заданное с помощью блока **SdV**(*). Данная ошибка детектируется, если при движении клапана разрешается контроль положения (сбрасывается бит 2 **CSC**);
- 10 – клапан открыт, но сигнала концевого выключателя открытия нет;
- 11 – клапан закрыт, но сигнала концевого выключателя закрытия нет;
- 12 – нет напряжения;
- 14 – клапан движется, а сигнала электрической части (привода) нет;
- 15 – это значение указывает, что блок находится в режиме отслеживания.

(*) Таймаут (время ожидания отключения) концевого выключателя в отсутствие блока **SdV** равно $0.1 * PT$ секунд. Максимально допустимое отклонение **Q%L** от **CDI** в отсутствие блока **SdV** равно $1000/PT$.

Настройки параметров задвижек и клапанов (SdV)



По значениям выходов блока **SdV** устанавливаются критерии аварийных ситуаций для блоков **KLP** и **ZDV**:

QDH – по значению этого выхода устанавливается максимально допустимое отклонение вычисленного положения клапана от реального: при $|CDI - Q\%L| \geq 100 * QDH / PT$ формируется ошибка 9 и процедура открытия/закрытия клапана прерывается (см. описа-

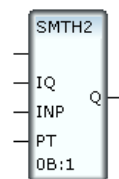
ние блока **KLP**). $QDH=IDH$, если $0 \leq IDH \leq 100$; $QDH=0$, если $IDH > 100$ или $IDH < 0$, а также если вход **IDH** не определен. При $QDH=0$ ошибка 9 формируется даже при $CDI=Q\%L$;

QDO – если при закрытии задвижки или клапана из положения «открыто» по истечении $QDO*PT$ секунд не отключился концевой выключатель открытия, формируется ошибка 3, и процедура закрытия прерывается (см. описание блоков **ZDV** и **KLP**). $QDO=IDO$, если $0.1 \leq IDO \leq 0.5$; $QDO=0.1$, если $IDO < 0.1$ или $IDO > 0.5$, а также если вход **IDO** не определен;

QDC – если при открытии задвижки или клапана из положения «закрыто» по истечении $(1-IDC)*PT$ секунд не отключились концевые выключатели закрытия и муфты, формируется ошибка 2, и процедура открытия прерывается (см. описание блоков **ZDV** и **KLP**). $QDC=IDC$, если $0.5 \leq IDC \leq 0.9$; $QDC=0.9$, если $IDC < 0.5$ или $IDC > 0.9$, а также если вход **IDC** не определен.

После обработки блока **SdV** для всех блоков **KLP** и **ZDV** на текущем узле будут использоваться установленные им настройки. Они будут изменены после обработки следующего блока **SdV** или следующего пересчета этого блока с измененными значениями его входов.

Звено второго порядка (SMTH2)



Данный блок с соединенными выходом **Q** и входом **IQ** выполняет сглаживание входного сигнала (**INP**) звеном второго порядка. Постоянная времени сглаживания определяется входом **PT**. Значение этого входа задает количество тактов, за которое выходное значение сравняется с входным при условии неизменности последнего после первого изменения.

Алгоритм сглаживания выглядит следующим образом (**i** обозначает текущий такт пересчета).

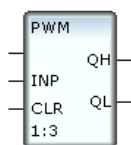
```

IF (INPi <> INPi-1) && (N == PT) THEN N := 1
IF N < PT THEN Qi := IQi + N * (INPi - IQi) / PT; N := N + 1
ELSE Qi := INPi

```


Звено ШИМ (PWM)

Широтно-импульсная модуляция (ШИМ) - вид импульсной модуляции, при которой под действием модулирующего сигнала изменяется длительность (ширина) импульсов при постоянной частоте их следования.



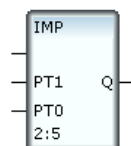
При подаче на вход **INP** модулирующего сигнала на выходе блока формируется широтно-модулированный импульсный сигнал единичной амплитуды с частотой следования импульсов $1/2t$, где t – период пересчета блока.

Длина импульса округляется до величины, кратной периоду пересчета блока. Для компенсации возникающей в связи с этим ошибки разности между округленной и истинной длиной импульсов суммируются. Когда абсолютное значение этой суммы превышает длительность одного такта, один такт добавляется или вычитается из длины очередного импульса. Подача на вход **CLR** отличной от нуля величины сбрасывает накопленную ошибку.

Значение модулирующего сигнала (**INP**) не должно выходить за границы диапазона $[-100, 100]$, максимальная частота должна быть много меньше частоты пересчета блока.

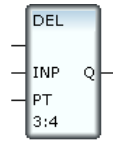
При положительном значении входной величины импульсы формируются на выходе **QH** (при этом **QL=0**), при отрицательном – на выходе **QL** (при этом **QH=0**).

Циклический импульс (IMP)



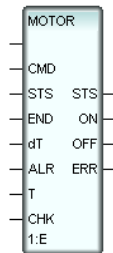
На выходе этого блока формируется последовательность импульсов с единичной амплитудой, длительностью **PT1** тактов пересчета и периодом следования (**PT1 + PT0**) тактов.

Задержка реакции (DEL)



Данный блок воспроизводит на выходе значение входа **INP**. Однако при равенстве входа нулю значение выхода перестает меняться на время задержки, задаваемой значением входа **PT** (в тактах пересчета).

Управление устройством типа 'двигатель' (MOTOR)



Данный блок предназначен для управления одно- или двухскоростным мотором, снабженным входом (входами) включения и выключения. Кроме того, мотор может быть оборудован следующими цифровыми датчиками: датчиком включения на скорости 1, датчиком включения на скорости 2, датчиком выключения и датчиком нагрузки. По сигналам этих датчиков блок **MOTOR** может анализировать состояние мотора.

Блок **MOTOR** может находиться в следующих режимах: WORK (работа), RESERVE (резерв), REPAIR (ремонт) и ERROR (ошибка). Режим RESERVE имеет разновидности: ACTIVE_RESERVE – резерв с возможностью включения, PASSIVE_RESERVE – резерв без возможности включения

Блок имеет следующие функциональные входы:

- **CMD** – команда управления:
 - 1 – включить на скорости 1;
 - 2 – выключить;
 - 4 – в ACTIVE_RESERVE: включить на скорости 1;
 - 8 – включить на скорости 2;
- **STS** – перевод в нужный режим:

Режим вычисляется как результат побитового сложения (OR) **STS** и значения ключа **PMPSTSM=<число>** в файле *CNF.

- 0 – блокировка команд **CMD**;
- 2 – разрешение команд **CMD**;
- 3 – перевод в **PASSIVE_RESERVE**;
- 4 – перевод в **REPAIR**;
- 5 – перевод в **ERROR**;
- 6 – сброс ошибок, предыдущий режим запоминается;
- 7 – сброс ошибок, предыдущий режим не запоминается;
- 8 – перевод в **ACTIVE_RESERVE**;
- 9 – в **PASSIVE_RESERVE**: перевод в **ACTIVE_RESERVE**.
- **END** – установленные в 1 биты этого входа интерпретируются следующим образом:
 - бит 0 – мотор включен на скорости 1;
 - бит 1 – мотор выключен;
 - бит 2 – наличие нагрузки;
 - бит 3 – мотор включен на скорости 2;
 - бит 4 – не анализировать сигнал датчика включения на скорости 1;
 - бит 5 – не анализировать сигнал датчика выключения;
 - бит 6 – не анализировать сигнал датчика нагрузки;
 - бит 7 – не анализировать сигнал датчика включения на скорости 2;
- **dT** – максимальная длительность сигналов, генерируемых на выходах **ON** и **OFF** (в секундах);
- **ALR** – ненулевое значение этого входа интерпретируется как команда аварийного выключения мотора, при этом блок переводится в режим **ERROR**. Эта команда имеет наивысший приоритет;
- **T** – в течение (**dT+T**) секунд с момента подачи команды блок ожидает прихода подтверждающих сигналов датчиков (см. описание выхода **ERR**);
- **CHK** – установленные в 1 биты интерпретируются следующим образом (все команды **CHK** имеют более высокий приоритет по сравнению с аналогичными командами других входов блока):
 - биты 0, 1, 2, 3 – ручная команда включения/выключения;
 - бит 5 (0x20) – аналог **STS**(вход)=7;
 - бит 6 (0x40) – разрешение анализа сигнала датчика нагрузки;
 - бит 7 (0x80) – отслеживание по сигналам датчиков включения и выключения;
 - бит 8 (0x100) – установка бита 7 (0x80) в случае ошибки;
 - бит 10 (0x400) – блокировка команды включения по **CHK**;

- бит 9 (0x200) – аналог **STS**(вход)=6;
- бит 11 (0x800) – блокировка команды выключения по **CHK**;
- бит 12 (0x1000) – установка бита 6 (0x40) выхода **STS** (в режиме WORK этот бит равен 0);
- бит 13 (0x2000) – разрешение команд **CMD**;
- бит 14 (0x4000) – разрешение отработки команды **CMD**=1 в резерве;
- бит 15 (0x8000) – запрет **STS**(выход)=5 (кроме прямой команды **ALR**).

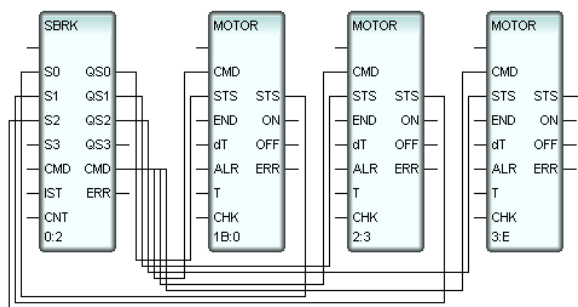
При использовании с блоком SBRK: при установке бита 15 необходимо установить бит 8.

Выходы блока имеют следующее назначение:

- **STS** – коды, индицируемые на этом выходе, соответствуют следующим состояниям (при ручном управлении по **CHK** дополнительно устанавливается бит 4 (0x10) **STS**):
 - 0 – инициализация;
 - 1 – включен на скорости 1;
 - 2 – выключен;
 - 3 – ACTIVE_RESERVE, выключен;
 - 4 – REPAIR;
 - 5 – ERROR;
 - 6 – выключен и запрещены команды (**STS**(вход)=0 и нет **CHK**=0x2000);
 - 7 – включен и запрещены команды;
 - 8 – включен на скорости 2;
 - 9 – процесс включения на скорости 1;
 - 10 – процесс выключения;
 - 11 – PASSIVE_RESERVE;
 - 12 – холостой ход;
 - 13 – процесс включения на скорости 2;
 - 14 – перевод в ACTIVE_RESERVE с выключением;
 - 15 – ACTIVE_RESERVE, включен.
- **ON** – в бите 0 этого выхода генерируются сигнал включения на скорости 1, в бите 2 – сигнал включения на скорости 2;
- **OFF** – на этом выходе генерируется сигнал выключения;
- **ERR** – характеристика аварийной ситуации:
 - 1 – одновременные сигналы любых двух или всех трех датчиков включения и выключения;

- 2 – мотор выключен внешними средствами (т.е. не с помощью блока **MOTOR**);
- 3 – мотор включен внешними средствами (на любой скорости);
- 4 – в течение (**dT+T**) секунд после подачи команды выключения не пришел подтверждающий сигнал от датчика выключения;
- 5 – в течение (**dT+T**) секунд после подачи команды включения не пришел подтверждающий сигнал от датчика включения;
- 6 – в течение **dT** секунд после подачи команды включения не сброшен сигнал датчика выключения;
- 7 – в течение **dT** секунд после подачи команды выключения не сброшен сигнал датчика включения;
- 8 – блок переведен в режим ERROR по **STS=5**. Значение 8 индицируется также при любой ошибке кроме тех, которые индицируются другими кодами **ERR**;
- 9 – отработана команда аварийного выключения двигателя **ALR<0**;
- 10 – при включенном моторе (на любой скорости) исчезла нагрузка;
- 11 – нет подтверждающего сигнала ни от одного из трех датчиков включения и выключения;
- 12 – мотор выключен или в резерве и появилась нагрузка;
- 14 – скорость изменена внешними средствами.

Управление группой устройств типа 'двигатель' (SBRK)



Данная FBD-программа управляет группой из 3 насосов (два рабочих и один резервный), поддерживающих давление P в трубопроводе в диапазоне [LOW, HIGH].

В программе входы **S0...S2** блока **SBRK** соединены с выходами **STS**, а выходы **QS0...QS2** – с входами **STS** блоков **MOTOR**. При этом вход **S_i** и

выход **QS_i** (т.е., имеющие один и тот же номер **i**) соединены соответственно с выходом **STS** и входом **STS** одного и того же блока **MOTOR**. По номеру $i=0\dots 2$ блок **SBRK** идентифицирует управляемое устройство.

Выход **CMD** блока **SBRK** соединен с входами **CMD** блоков **MOTOR**.

Установленные в 1 биты **CNT** интерпретируются следующим образом:

- бит 0 (0x1) – $P \geq \text{HIGH}$;
- бит 1 (0x2) – $P \leq \text{LOW}$;
- бит 2 (0x4) –
- бит 4 (0x10) – запрет анализа HIGH;
- бит 5 (0x20) – запрет анализа LOW;
- бит 6 (0x40) –

CMD – команда управления (может быть импульс): 1 – включить, 2 – выключить. **SBRK** запоминает последнюю команду.

Алгоритм работы:

- по команде «включить» включаются 2 мотора, начиная с нулевого. Насос, выбранный в качестве резервного (находится в режиме **PASSIVE_RESERVE**), не включается;
- при $P \geq \text{HIGH}$: насос, выбранный в качестве резервного, переводится в **ACTIVE_RESERVE**;
- при $P \leq \text{LOW}$: если хотя бы один насос включен, включается насос, находящийся в **ACTIVE_RESERVE**;
- по команде «выключить» включенные насосы выключаются, а насос в **ACTIVE_RESERVE** переводится в **PASSIVE_RESERVE**.

Вход **IST** используется для управления отдельными устройствами. Значения битов 0-3 задают команды 3, 8 и 9, аналогичные командам входа **STS** блока **MOTOR**. Номер устройства (блока **MOTOR**), которое должно выполнить такую команду, задается значением битов 4-7 **IST**.

Режим вычисляется как результат побитового сложения (OR) **IST** и значения ключа **OTHSTSM**=<число> в файле *CNF.

Значения байта 0 (0xFF) **ERR** индицируют следующие ситуации:

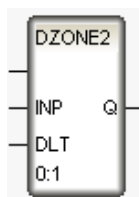
- 0 – последняя команда – выключить, не выбран резерв;
- 1 – команда – включить, не выбран резерв;
- 2 – команда – выключить, выбран резерв;
- 3 – команда – включить, выбран резерв;
- 4 – команда – выключить, резерв выбран и переведен (была ситуация $P \geq \text{HIGH}$);
- 5 – команда – включить, резерв выбран и переведен;
- 6 – ошибка;

- 7 – ошибка;
- 8 – ошибка;
- 9 – требуется включить резерв, а резерв был только выбран, но не переведен;
- 10 – ошибка;
- 11 – команда – включить, требуется включить резерв;
- 12 – все выключены, требуется включить резерв;
- 13 – все включены;
- 14 – все неисправны;
- 15 – нет подключенных блоков **MOTOR** (или они блокируют команды).

Биты байта 1 (0xFF00) **ERR**, установленные в 1, индицируют следующие ситуации:

- бит 8 (0x100) – невыполнимая команда включения (нет блоков для включения),
- бит 9 (0x200) – обнаружено устройство с ручным управлением;
- бит 10 (0x400) – некорректная команда **IST**.

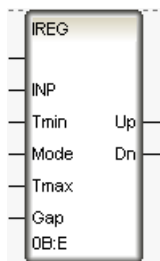
Зона нечувствительности 2 (DZONE2)



Данный блок работает по следующему алгоритму:

- $Q = INP - DLT$, если $INP > DLT$;
- $Q = INP + DLT$, если $INP < - DLT$;
- $Q = 0$, если $- DLT < INP < DLT$.

Импульсный регулятор (IREG)



Данный блок выполняет ШИМ-преобразование входного сигнала (**INP**) по следующему алгоритму:

- если $|\mathbf{INP}| > 100$, то $|\mathbf{INP}| = 100$;
- длительность импульсов (амплитуда равна 1) и длительность промежутков между импульсами определяются по формулам:

$$T_{\text{pulse}} = \frac{T_{\text{max}} * \mathbf{INP}}{100}, \text{ если } \frac{T_{\text{max}} * \mathbf{INP}}{100} \geq T_{\text{min}}$$

Если $\frac{T_{\text{max}} * \mathbf{INP}}{100} < T_{\text{min}}$, импульс не генерируется;

$$T_{\text{pause}} = T_{\text{max}} - T_{\text{pulse}}$$

T_{max} – время хода устройства; значения T_{min} и T_{max} задаются в секундах;

- для учета люфта устройства предусмотрен вход **Gap**. Значение нулевого байта **Gap** (**Gap₀**) задает уширение первого генерируемого импульса при изменении знака **INP** с минуса на плюс, значение первого байта **Gap** (**Gap₁**) – уширение первого импульса при изменении знака **INP** с плюса на минус. Целое значение байта интерпретируется как число десятков миллисекунд. Таким образом, длительность первого импульса при изменении знака **INP** с минуса на плюс определяется по формуле:

$$T_1 = T_{\text{pulse}} + \mathbf{Gap}_0$$

Длительность первого импульса при изменении знака **INP** с плюса на минус определяется по формуле:

$$T_1 = T_{\text{pulse}} + \mathbf{Gap}_1$$

Для разрешения использования **Gap₀** нужно установить бит 8 **Mode**. Для разрешения использования **Gap₁** нужно установить бит 9 **Mode**.

Если **INP** > 0, импульсы формируются на выходе **Up**, в противном случае – на выходе **Dn**;

- если **INP** меняет знак, соответствующий выход обнуляется, и генерируется импульс на другом выходе;
- если **INP** изменяется с сохранением знака, то возможны два случая:
 - $\Delta(\mathbf{INP}) > 0$
 - если изменение произошло при генерации импульса, длительность импульса увеличивается;
 - если изменение произошло в паузе, генерируется импульс (приоритет 1);
 - $\Delta(\mathbf{INP}) < 0$

- если изменение произошло при генерации импульса, длительность импульса уменьшается;

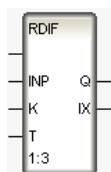
- если изменение произошло в паузе, длительность паузы увеличивается;

- если бит 0 **Mode** равен 1, блокируется выдача сигнала на выход **Up**;
- если бит 1 **Mode** равен 1, блокируется выдача сигнала на выход **Dn**;
- если бит 2 **Mode** равен 1, блокируется выдача сигналов на **Up** и **Dn**;
- если бит 3 **Mode** равен 0, то биты 0 и 1 **Mode**, кроме того, блокируют ручные команды, если 1 – не блокируют;
- если значение битов 4-7 (0xF0) **Mode** равно $a > 0$, то при изменении знака **INP** выходы **Up** и **Dn** принимают значение 0 на a тактов пересчета;
- если значение битов 10 и 11 (0xC00) **Mode** равно 3, на выходе вместо «длинного» импульса генерируется последовательность импульсов минимальной длительности. Если значение битов равно 1, в течение последних $x\%$ длительности «длинного» импульса на выходе генерируется последовательность импульсов минимальной длительности (по умолчанию $x=5$). Если значение битов равно 2, последовательность импульсов минимальной длительности генерируется на выходе в том случае, если $INP < k * T_{max}$ (по умолчанию $k=0.05$). С помощью файла *.cnf (ключи **IREGPR1** и **IREGPR2**) можно задать другие x и k для этого режима.

Для переопределения алгоритма используются следующие ключи в файле *.cnf:

- в режиме 1 – **IREGAG0**=2 или 3;
- в режиме 2 – **IREGAG1**=1 или 3;
- в режиме 3 – **IREGAG2**=1 или 2;
- присвоение 1 биту 12 (0x1000) **Mode** в ручном режиме является командой выработки импульса минимальной длительности на выходе **Up**;
- присвоение 1 биту 13 (0x2000) **Mode** в ручном режиме является командой выработки импульса минимальной длительности на выходе **Dn**;
- присвоение 1 биту 14 (0x4000) **Mode** переводит блок в ручной режим, алгоритм расчета импульса отключается, **Up=Dn=0**;
- присвоение 1 биту 15 (0x8000) **Mode** является командой рестарта алгоритма блока.

Реальное дифференцирование (RDIF)

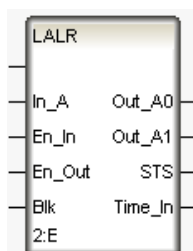


Данный блок работает по следующему алгоритму (n – номер такта пере-счета):

$$Q_n = \frac{Q_{n-1}}{1 + T} + K \frac{INP_n - INP_{n-1}}{1 + 1 / T}$$

$$IX_n = INP_n$$

Управление защитой (LALR)



Если бит 11 (0x800) **Blk** равен 0, биты 0-7 выхода **STS** индицируют со-стояния по выходу **Out_A0**, а биты 8-15 – состояния по выходу **Out_A1**.

Если бит 11 (0x800) **Blk** равен 1, биты 0-7 выхода **STS** индицируют со-стояния по выходу **Out_A0** (состояния по **Out_A1** не индицируются).

Установленные в 1 биты выхода **STS** индицируют следующие состояния:

- Бит 0, 0x1 (бит 8, 0x100) – выведено;
- Бит 1, 0x2 (бит 9, 0x200) – введено;
- Бит 2, 0x4 (бит 10, 0x400) – выведено и защита сработала;
- Бит 3, 0x8 (бит 11, 0x800) – введено и защита сработала;
- Бит 4, 0x10 (бит 12, 0x1000) – выведено и блокировка;
- Бит 5, 0x20 (бит 13, 0x2000) – введено и блокировка;
- Бит 6, 0x40 (бит 14, 0x4000) – выведено, блокировка и защита сра-ботала;
- Бит 7, 0x80 (бит 15, 0x 8000) – введено, блокировка и защита сра-ботала.

Состояние по умолчанию – **STS**=0.

Введение – **En_In**=1 (уровень или импульс).

Выведение – **En_Out**=1 (уровень или импульс).

Требование защиты – **In_A**=1.

Назначение битов **Blk**:

- бит 0 – 0 – защита не заблокирована, 1 – заблокирована;
- бит 3 (0x8) – **Out_A1=Out_A0** принудительно;
- бит 8 (0x100) – если защита сработала, 1 генерируется на **Out_A0** (если бит 8 равен 0) или на **Out_A1** (если бит 8 равен 1);
- бит 9 (0x200) – если 1, **Out_A1=Out_A0=0** при **In_A=0** (память отключена);
- бит 10 (0x400) – если 1, требование защиты игнорируется, **In_A** автоматически обнуляется;
- бит 11 (0x800) – режим **STS** (см. выше);
- биты 16-23 (0xFF0000) – значение этих битов задает задержку срабатывания (в секундах).

Выходы **Out_A1** и **Out_A0** принимают значение 1 в том случае, если **In_A=1**, **En_In=1**, защита не заблокирована и истекло время задержки.

В **Time_In** записывается время появления 1 на выходе **Out_A0/Out_A1**.

Out_A0/Out_A1 сохраняет значение 1 пока **En_Out=0**.

Блокировка не обнуляет **Out_A0/Out_A1**.

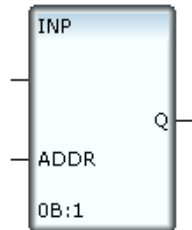
Блок VLV (VLV)

Зарезервирован.

Раздел 'Ввод/вывод. Переходы'

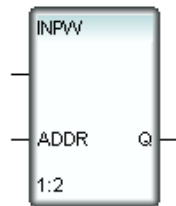
Функции чтения и записи в порт поддерживаются только в Микро МРВ для DOS.

Чтение из порта (INP)



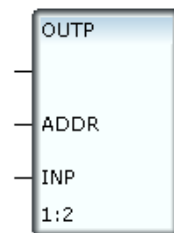
Этот блок считывает один байт данных из порта ввода/вывода, адрес которого задается входом **ADDR**. Считанное значение присваивается выходу **Q**.

Чтение слова из порта (INPW)



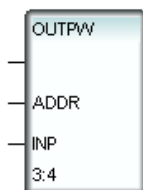
Этот блок считывает слово данных из порта ввода/вывода, адрес которого задается входом **ADDR**. Считанное значение присваивается выходу **Q**.

Запись в порт (OUTP)



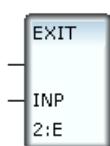
Этот блок посылает в порт ввода/вывода один байт данных. На вход **INP** подается посылаемое значение, на вход **ADDR** – адрес порта.

Запись слова в порт (OUTPW)



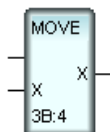
Этот блок посылает в порт ввода/вывода слово данных. На вход **INP** подается посылаемое значение, на вход **ADDR** – адрес порта.

Выход из программы (EXIT)



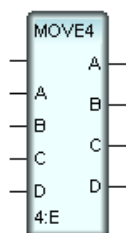
Данный блок позволяет выйти из программы до пересчета всех ее функциональных блоков. Выход производится при наличии ненулевого значения на входе **INP**.

Пересылка значения (MOVE)



$$X(\text{выход}) = X(\text{вход})$$

Пересылка четырех значений (MOVE4)

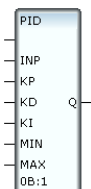


$$A(\text{выход}) = A(\text{вход}) ; B(\text{выход}) = B(\text{вход}) ;$$

$$C(\text{выход}) = C(\text{вход}) ; D(\text{выход}) = D(\text{вход}) .$$

Раздел 'Регулирование'

Звено PID (PID)



Этот блок формирует выходное значение по ПИД-закону от величины, поданной на вход **INP**:

$$Q_i = KP * INP_i + \frac{KD * (INP_i - INP_{i-1})}{\Delta t} + KI * \Delta t * \sum_{k=1}^i INP_k$$

где **i** – текущий такт пересчета, **KP**, **KD** и **KI** – соответственно коэффициенты при пропорциональной, дифференциальной и интегральной составляющих, **Δt** – период пересчета блока в секундах (длительность такта).

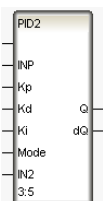
Модуль подаваемого на вход **KI** отрицательного значения передается на выход. Далее при подаче на вход **KI** неотрицательного значения регулирование начинается с установленной величины.

Для ограничения величины управляющего воздействия используются входы блока **MIN** и **MAX**. Если величина управления меньше **MIN**, то **Q = MIN**, если величина управления больше **MAX**, то **Q = MAX**, при этом в обоих случаях накопление интегральной составляющей закона регулирования прекращается.

Данный блок вычисляет величину управляющего воздействия по значению рассогласования регулируемой величины и задания, которое предварительно надо вычислять с помощью блока **X-Y**.

Введение в алгоритм параметра **Δt** исключает необходимость пересчета настроек регулятора при смене периода пересчета.

Звено PID2 (PID2)



Данный блок представляет собой PID-регулятор с возможностью модифицирования алгоритма и имеет, относительно блока **PID**, дополнительные входы **MODE** и **IN2** и дополнительный выход $dQ_k = Q_k^* - Q_{k-1}$ (Q_k^* – вычисленное значение на данном такте пересчета, Q_{k-1} – значение выхода **Q** на предыдущем такте пересчета).

Биты **MODE** имеют следующее назначение:

- бит 0 (0x1) и бит 1 (0x2) – установка бита 0 или бита 1 запрещает увеличение интегральной составляющей, при этом:
 - если установлен бит 0 и $dQ \leq 0$, то $Q = Q^*$
 - если установлен бит 0 и $dQ > 0$, то $Q = Q_{k-1}$
 - если установлен бит 1 и $dQ \geq 0$, то $Q = Q^*$
 - если установлен бит 1 и $dQ < 0$, то $Q = Q_{k-1}$
- бит 2 (0x4) – если установлен, **dQ** клипсируется по такому же механизму, что и **Q** при установленном бите 12 (см. описание бита 12). Сначала проверяется **Q**, потом – **dQ**;
- бит 3 (0x8) – см. описание бита 12;
- бит 4 (0x10) – если установлен, блок умножает рассогласование на входе **INP** на 0.5 в том случае, если **INP**>50 и на предыдущем такте пересчета **INP**<>0;
- бит 5 (0x20) – установка этого бита разрешает использование зоны нечувствительности (**zone**). Зона нечувствительности определяется по формуле $zone_{ni} = k_n * MAX_i$. Коэффициенты k_n (индекс $n = 0...15$ задается значением полубайта 5 (0xF0000) **MODE**) имеют предустановленные значения ($k_0=0.01$; $k_1=0.001$; $k_2=0.0001$; $k_3=0.002$; $k_4=0.003$; $k_5=0.005$; $k_6=0.0002$; $k_7=0.0003$; $k_8=0.0005$; $k_9=0.2$; $k_{10}=0.3$; $k_{11}=0.4$; $k_{12}=0.5$; $k_{13}=0.01$; $k_{14}=1$; $k_{15}=10$), которые могут быть изменены с помощью ключей **PIDDZN**<nn> в файле *.cnf (см. **Задание параметров работы мониторов**). По умолчанию используется k_0 (изменять значение этой переменной не рекомендуется);

В DOS **zone** = 1 всегда.

- бит 6 (0x40) – если установлен, значение выхода **dQ** сглаживается по формуле $dQ_k = 0.5 * (dQ_{k-1} + dQ_k^*)$;
- бит 7 (0x80) – если установлен, **Q** сглаживается по формуле $Q_k = 0.5 * (Q_{k-1} + Q_k^*)$;
- бит 8 (0x100) – установка этого бита переводит блок в ручной режим ($Q = IN2$); при последующем переходе в автоматический режим регулирования (бит 8 равен 0) скачок управляющего воздействия (**Q**) сглаживается.

Если установлены биты 8 (0x100) и 10 (0x400), то $Q = IN2$, но при последующем переходе в автоматический режим регулирования

скачок управляющего воздействия (Q) не сглаживается;

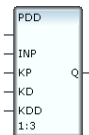
- бит 9 (0x200) – если установлен, $Q = Q^* + IN2$, где Q^* – значение, вычисленное по PID-закону;
- бит 10 – см. описание бита 8;
- бит 11 (0x800) – если установлен, интегральная составляющая PID-закона обнуляется;
- бит 12 (0x1000) – если установлен, интегральная составляющая фиксируется (запрещается ее изменение), а Q лежит в диапазоне $[MIN_i, MAX_i]$ (при выходе за границы Q клиппируется; индекс $i = 0 \dots 15$ задается значением полубайта 4 (0xF000) **MODE**). Константы MAX_i имеют предустановленные значения ($MAX_0=100$; $MAX_1=0.1$; $MAX_2=1$; $MAX_3=10$; $MAX_4=1000$; $MAX_5=10000$; $MAX_6=0.5$; $MAX_7=5$; $MAX_8=50$; $MAX_9=150$; $MAX_{10}=500$; $MAX_{11}=5000$; $MAX_{12}=50000$; $MAX_{13}=100$; $MAX_{14}=0.01$; $MAX_{15}=3660$), которые могут быть изменены с помощью ключей **PIDLIM**<пп> в файле *.cnf (см. **Задание параметров работы мониторов**). Константы MIN_i вычисляются по соответствующим MAX_i :
 - если бит 3 (0x8) не установлен, $MIN_i = -MAX_i$;
 - если бит 3 (0x8) установлен, $MIN_i = 0$.

По умолчанию используется MAX_0 (изменять значение этой переменной не рекомендуется);

В DOS $MAX = 100$ и $MIN = -100$ всегда.

- бит 14 (0x4000) – установка этого бита запрещает изменение интегральной составляющей;
- бит 15 (0x8000) – если установлен, дифференциальная составляющая считается, но не участвует в формировании выхода.

Звено PDD (PDD)



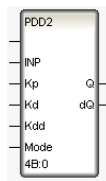
Этот блок формирует выходное значение по ПДД-закону от значения входа **INP**:

$$Q_i = KP * INP_i + \frac{KD * (INP_i - INP_{i-1})}{\Delta t} + \frac{KDD * (INP_i - 2 * INP_{i-1} + INP_{i-2})}{(\Delta t)^2}$$

где i – текущий такт пересчета, KP и KD – соответственно коэффициенты при пропорциональной и дифференциальной составляющих, KDD – коэффициент при второй производной, Δt – период пересчета блока в секундах (длительность такта).

Данный блок вычисляет величину управляющего воздействия по значению рассогласования регулируемой величины и задания, которое предварительно надо вычислять с помощью блока **X-Y**. Все дополнительные функции контура также программируются с помощью отдельных блоков.

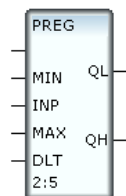
Звено PDD2 (PDD2)



Данный блок, относительно блока **PDD**, имеет дополнительный вход **MODE** и дополнительный выход $dQ_k = Q^*_k - Q_{k-1}$ (Q^*_k – вычисленное значение на данном такте пересчета, Q_{k-1} – значение выхода **Q** на предыдущем такте пересчета).

- если **MODE** = 1 и $dQ \leq 0$, то $Q = Q^*$;
- если **MODE** = 1 и $dQ > 0$, то $Q = Q_{k-1}$;
- если **MODE** = 2 и $dQ \geq 0$, то $Q = Q^*$;
- если **MODE** = 2 и $dQ < 0$, то $Q = Q_{k-1}$;
- если установлен бит 7 **MODE** (0x80), **Q** сглаживается по формуле $Q_k = 0.5 * (Q_{k-1} + Q^*_k)$;
- если установлен бит 8 **MODE** (0x100), блок переходит в ручной режим, при этом $Q=INP$;
- если установлен бит 12 **MODE** (0x1000), **Q** клипсируется при выходе из диапазона [-100,+100]. При этом интегральная составляющая также ограничивается, и адекватно корректируется dQ ;
- если установлен бит 15 **MODE** (0x8000), дифференциальная составляющая не учитывается.

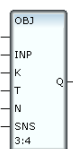
Трехпозиционный регулятор (PREG)



Этот блок сигнализирует о выходе контролируемой величины, подаваемой на вход **INP**, за границы диапазона [**MIN**, **MAX**]. Блок может быть использован в программе, с помощью которой значение некоторого сигнала удерживается в заданном диапазоне. Вход **DLT** блока предназначен для задания величины гистерезиса на отключение сигналов управления **QL** и **QH**.

Блок работает по следующему алгоритму. Пока **INP** не выходит за границы диапазона [**MIN**, **MAX**], **QL** = **QH** = 0. Если **INP** > **MAX**, выход **QL** принимает значение 1, которое сбрасывается в 0 только после того, как **INP** станет меньше (**MAX** - **DLT**). Аналогично, если **INP** < **MIN**, выход **QH** принимает значение 1, которое сбрасывается в 0 только после того, как **INP** станет больше (**MIN** + **DLT**).

Модель объекта (OBJ)



Данный блок моделирует объект управления для отладки алгоритмов регулирования или подготовки демонстрационных проектов. Он представляет собой комбинацию апериодического (инерционного) звена первого порядка и звена запаздывания, т.е. передаточная функция блока (при входной функции e^{st}) имеет вид:

$$\Phi_a(s) = \frac{k}{T * s + 1} e^{-ls}$$

где **k** и **T** – соответственно коэффициент усиления и постоянная времени инерционного звена первого порядка, $l > 0$ – время запаздывания.

Кроме того, на выходной сигнал блока можно наложить помеху в виде случайной составляющей, синусоидального сигнала или случайных бросков. Здесь же можно задать случайное колебание динамических характеристик объекта.

Входным по отношению к моделируемому объекту является вход **INP**. Входы **K**, **T** и **N** используются для задания соответственно коэффициента усиления, постоянной времени и времени запаздывания. Последние два параметра задаются в тактах пересчета, максимальное значение времени запаздывания – 4.

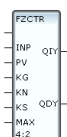
Вход **SNS** предназначен для управления случайными помехами, вносимыми в работу объекта. Значение 1 отдельных битов этого входа включает следующие помехи:

- 1 бит – добавление к выходному сигналу случайной величины в диапазоне от 0 до 1%;

- 2 бит – формирование пика величиной 25% от значения выхода с вероятностью 0,01;
- 3 бит – добавление к выходу синусоидального сигнала с амплитудой 2% от значения выхода;
- 5 бит – случайное увеличение коэффициента усиления в диапазоне от 0 до 2%;
- 6 бит – случайное увеличение постоянной времени в диапазоне от 0 до 2%;
- 7 бит – случайное изменение на 1 запаздывание.

Первые три помехи добавляются к выходу блока после формирования его нового значения. Динамические характеристики объекта (последние три помехи) корректируются до пересчета блока.

Нечеткий регулятор (FZCTR)



Данный функциональный блок реализует функцию нечеткого регулятора. На вход **INP** подается регулируемое значение. Вход **PV** предназначен для задания уставки.

На выходе **QIY** блока формируется величина управляющего воздействия. Выход **QDY** используется для вывода величины приращения управляющего воздействия на текущем пересчете блока.

Значения выходов формируются по следующему алгоритму:

$$QIY_i = QIY_{i-1} + QDY_i$$

$$QDY_i = kg * QDY_g + kn * QDY_n + ks * QDY_s$$

где

i – текущий такт пересчета,

kg , kn и ks – принадлежность рассогласования к категориям «большое», «среднее» и «малое» соответственно;

QDY_g , QDY_n , QDY_s – приращение управления по условию «большое», «среднее» и «малое» отклонение соответственно.

Приращения по каждой из категорий отклонения рассчитываются по следующей формуле:

$$QDY_j = K_j * SIGN(PV - INP)$$

где j – признак категории рассогласования:

g – сильное рассогласование;

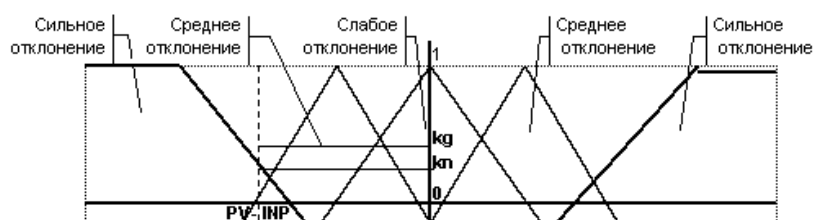
n – среднее рассогласование;

s – слабое рассогласование.

Коэффициенты **K_j** настраиваются входами **KG**, **KN** и **KS** данного блока. Если эти коэффициенты не заданы, то их значения принимаются по умолчанию равными 0.3, 0.2 и 0.05 соответственно.

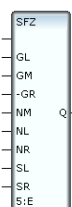
Вход **MAX** ограничивает величину приращения управляющего воздействия. Это ограничение вычисляется как произведение данного входа и рассогласования задания и регулируемого значения.

На рисунке демонстрируется подход к определению коэффициентов принадлежности к интервалам рассогласования.

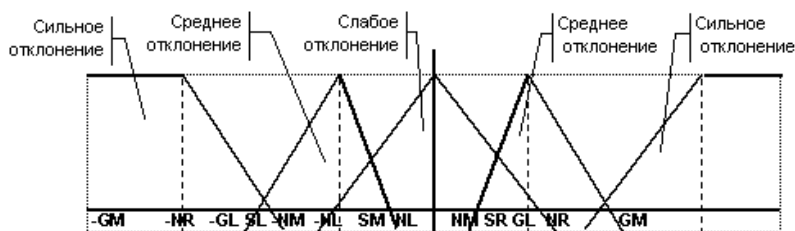


Границы диапазонов категорий рассогласования задаются с помощью блока **SFZ**. Он передает свои настройки всем нечетким регуляторам данного узла. Поэтому этот блок следует разместить перед каждым регулятором, имеющим индивидуальные настройки.

Настройка FZCTR (SFZ)



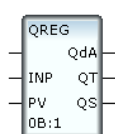
Данный блок предназначен для настройки диапазонов нечеткого регулятора, реализованного в функциональном блоке **FZCTR**.



Обозначенные на рисунке границы диапазонов настраиваются одноименными входами данного блока (на входе блока задается модуль значения). Если этот блок не использовался, то по умолчанию принимаются следующие значения настроек: **GM**=1, **-GR**=-0.5, **GL**=0.5, **NR**=0.75, **NM**=0.5, **NL**=0.02, **SR**=0.25, **SM**=0, **SL**=-0.25. Выход **Q** данного блока не используется.

Этот блок следует разместить перед каждым регулятором, имеющим индивидуальные настройки. Блок **SFZ** должен выполняться перед блоком нечеткого регулятора.

Показатели качества регулирования (*QREG*)

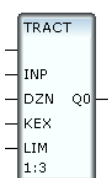


Этот блок предназначен для оценки качества регулирования. На вход **INP** подается значение регулируемого параметра, на вход **PV** – задание. На выходах формируются следующие значения:

- **QdA** – максимум абсолютного значения рассогласования (первый скачок при изменении задания исключается);
- **QT** – количество тактов с момента изменения задания, на которых рассогласование превышало 1% задания;
- **QS** – накапливающаяся сумма модуля рассогласования. При суммировании не учитываются значения меньше 0.5%.

При изменении задания (**PV**) выходы обнуляются.

Обработка сигнала (*TRACT*)



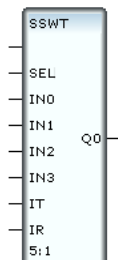
Этот блок позволяет реализовать ряд функций по обработке входного сигнала, подаваемого на вход **INP**. К ним относятся:

- зона нечувствительности;
- экспоненциальное сглаживание;
- ограничение значения.

Для настройки этих функций используются три входа. На вход **DZN** пода-

ется величина зоны нечувствительности, на вход **KEX** – величина коэффициента сглаживания, на вход **LIM** – ограничение на абсолютное значение.

Переключение с динамической балансировкой (SSWT)



Функцией этого блока является реализация безударной коммутации его входов **IN0**, ... **IN3** на выход **Q0**. Номер входа, коммутируемого на выход, задается входом **SEL**.

При каждом изменении значения входа **SEL** включается алгоритм динамической балансировки. При этом значение выхода вычисляется по следующей формуле:

$$Q_i = IN\langle N \rangle_{i-1} * (1 - IT) + Q_{i-1} * IT$$

где

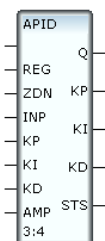
i – текущий такт пересчета;

N – номер коммутируемого входа;

IT – значение входа управления инерционностью. Это значение должно удовлетворять условию $0 \leq IT < 1$;

После того как в первый раз выполнится условие $Q - IN\langle N \rangle < IR$, алгоритм динамической балансировки отключается, и в дальнейшем значение выбранного входа просто копируется на выход **Q0**.

Адаптивное регулирование (APID)



Данный блок позволяет организовать регулирование по PID-закону с ав-

томатическим вычислением настроек регулятора.

В инструментальной системе работа блока адаптивного регулирования поддерживается только в отладчике программ. Для использования этого алгоритма в реальном времени требуется Adaptive Control MPB, Adaptive Control Double Force MPB или Adaptive Микро TRACE MODE с поддержкой данной функции.

Входы и выходы блока

Блок **APID** имеет семь функциональных входов и пять выходов. Входы имеют следующее назначение:

REG – вход управления, его значение определяет следующие режимы работы блока:

- 0 – регулирование с использованием настроек, поданных на входы **KP**, **KI** и **KD**;
- 1 – вычисление настроек регулятора;
- 2 – сброс всех результатов вычислений (значения настроек на выходах блока, критическая частота и амплитуда генератора);
- 3 – сброс критической частоты и амплитуды;
- 4 – установка режима нефорсированного управления, т.е. изменение задания не обрабатывается в пропорциональной и дифференциальной составляющих. В этом случае переходный процесс более затянут, но сокращается величина динамической погрешности. По умолчанию регулятор работает в этом режиме;
- 5 – установка режима форсированного управления, т.е. при смене задания произойдет скачок по управлению;
- 6 – непрерывное вычисление настроек регулятора с плавным переходом от старых значений к новым. В этом режиме определение текущих настроек и их введение в регулятор происходит автоматически. Значения настроек постоянно индицируются на выходах **KP**, **KI**, **KD**;
- 10 – регулирование с использованием вычисленных настроек, сформированных на выходах **KP**, **KI** и **KD**. Отрицательным значением (кратковременно) по входу **REG** может быть установлена добротность заграждающих фильтров в диапазоне 1-50 (по умолчанию 1). Чем выше уровень шума, тем большее значение добротности рекомендуется устанавливать. При этом увеличивается и время самонастройки регулятора;

ZDN – задание;

INP – регулируемый параметр;

KP – коэффициент при пропорциональной составляющей;

KD – коэффициент при дифференциальной составляющей;

KI – коэффициент при интегральной составляющей;

AMP – ограничения на амплитуды. Положительное значение – на амплитуду сигнала, добавляемого к выходу регулятора (по умолчанию 10, минимальная – 4); отрицательное – на колебания выхода объекта (по умолчанию 1, минимальная – 0.5). При старте МРВ значение этого входа должно быть положительным (т.е. вход должен задавать ограничение амплитуды на входе объекта).

На выходах данного блока формируются следующие величины:

Q – величина управляющего воздействия;

KP – коэффициент при пропорциональной составляющей;

KD – коэффициент при дифференциальной составляющей;

KI – коэффициент при интегральной составляющей;

STS – индикатор текущего состояния адаптации.

Величины, подаваемые на входы **ZDN**, **INP** и **AMP**, должны задаваться в процентах (в диапазоне от 0 до 100). Величина управляющего воздействия на выходе **Q** также формируется в диапазоне от 0 до 100.

Описание работы блока

Критерием вычисления настроек является минимизация среднеквадратичной ошибки регулирования. Для настройки регулятора на вход объекта подается пробный гармонический сигнал. При этом амплитуда колебаний регулируемого параметра удерживается в пределах от 0.3% до 1%. Получаемые настройки минимизируют колебания при переходных процессах.

Реализованный алгоритм является помехоустойчивым. Он работает даже в том случае, если дисперсия шума в несколько раз превышает амплитуду пробных колебаний выходного сигнала. При настройке он исключает появление неустойчивых режимов.

Быстродействие процесса самонастройки алгоритма управления зависит от уровня шумов и неслучайных возмущений, действующих на объект управления.

Если вход **REG** равен 0 или 10, блок реализует функцию PID-регулятора. В первом случае используются настройки на входах **KP**, **KD** и **KI**, а во втором – на выходах с теми же именами (значения, полученные при адап-

тации).

Для перехода в режим автонастройки (одноразовое определение оптимальных настроек регулятора) следует установить в 1 значение входа **REG**. Для перехода в режим непрерывной адаптации (постоянное определение оптимальных настроек регулятора) нужно на вход **REG** подать значение 6. В обоих случаях регулятор не отключается и процесс управления объектом осуществляется параллельно с процессом настройки регулятора.

Индикатором хода настройки регулятора является величина выхода **STS**. Он может принимать следующие значения:

- 0 – настройка завершена (или не запускалась);
- 1 – настройка регулятора завершена успешно. Получены значения оптимальных настроек;
- 2...100 – поиск оптимальных настроек. Значение выхода уменьшается от 100 до 1 и показывает близость к завершению работы алгоритма;
- 101 – невозможно провести адаптацию. Слишком много воздействий на систему или уровень шумов сравним с амплитудой гармоники на выходном сигнале объекта. Следует уменьшить внешние воздействия или попробовать увеличить уровень максимально возможной амплитуды колебаний на выходе объекта;
- 102 – невозможно провести адаптацию. Не удастся достигнуть необходимой амплитуды сигнала на выходе (от 0,3 до 1 максимальной). Следует увеличить амплитуду входа или уменьшить амплитуду выхода;
- 103 – невозможно провести адаптацию. Границы диапазона изменения управляющего сигнала не дают увеличить амплитуду сигнала на входе объекта. Следует либо изменить амплитуды, либо сместить уровень управляющего воздействия;
- 104 – невозможно провести адаптацию. Границы диапазона изменения регулируемого параметра не дают увеличить амплитуду сигнала на выходе объекта. Следует повысить ограничение на входную амплитуду или сместить входной сигнал с границ диапазона регулирования;
- 105 – невозможно провести адаптацию. Следует увеличить частоту пересчета программы.

В случае нормального завершения адаптации (значение 1 на выходе **STS**) на выходах **KP**, **KD** и **KI** формируются новые значения соответствующих настроек для регулятора. Для их использования надо присвоить входу **REG** значение 10.

При задании жестких ограничений на амплитуду пробного

сигнала алгоритм может сформировать сообщение о невозможности подобрать оптимальные настройки. Минимальное значение амплитуды следует устанавливать с учетом разрядности АЦП, коэффициента усиления и инерционности объекта.

Ограничения на применение

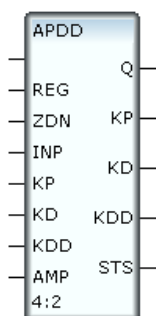
Данный алгоритм настройки PID-регулятора не может быть применён к структурно неустойчивым объектам управления.

Оптимальные настройки вычисляются при соотношении времени запаздывания к постоянной времени объекта не более 0,5.

Период опроса параметра и вызова алгоритма настройки должна быть не менее 0.05 постоянной времени объекта.

Амплитуда колебаний регулируемого параметра (отклик на пробный сигнал) должна быть не ниже 8 единиц кода АЦП. Например, 0.5% для 12-разрядного АЦП составляет 20 единиц.

Блок адаптивного регулирования (APDD)



Данный блок позволяет организовать регулирование по PDD-закону с автоматическим вычислением настроек регулятора.

В инструментальной системе работа блока адаптивного регулирования поддерживается только в отладчике программ. Для использования этого алгоритма в реальном времени требуется Adaptive Control MPB, Adaptive Control Double Force MPB или Adaptive Микро TRACE MODE с поддержкой данной функции.

Входы и выходы блока

Блок **APDD** имеет 7 функциональных входов и 5 выходов. Его входы имеют следующее назначение:

REG – вход управления. Значение этого входа определяет следующие режимы работы блока:

- 0 – регулирование с использованием настроек, поданных на входы **KP**, **KD** и **KDD**;
- 1 – вычисление настроек регулятора;
- 2 – сброс всех результатов вычислений (значения настроек на выходах блока, критическая частота и амплитуда генератора);
- 3 – сброс критической частоты и амплитуды;
- 6 – непрерывное вычисление настроек регулятора с плавным переходом от старых значений к новым. В этом режиме определение текущих настроек и их введение в регулятор происходит автоматически. Значения настроек постоянно индицируются на выходах **KP**, **KD**, **KDD**;
- 10 – регулирование с использованием вычисленных настроек, сформированных на выходах **KP**, **KD** и **KDD**;

ZDN – задание регулятору;

INP – регулируемый параметр;

KP – коэффициент при пропорциональной составляющей;

KD – коэффициент при первой производной PDD-закона;

KDD – коэффициент при второй производной PDD-закона;

AMP – ограничения на амплитуды. Положительное значение – на амплитуду сигнала, добавляемого к выходу регулятора (по умолчанию 10, минимальная – 4), а отрицательное – на колебания выхода объекта (по умолчанию 1, минимальная – 0.5). При старте МРВ значение этого входа должно быть положительным (т.е. вход должен задавать ограничение амплитуды на входе объекта).

На выходах данного блока формируются следующие величины:

Q – величина управляющего воздействия;

KP – коэффициент при пропорциональной составляющей;

KD – коэффициент при первой производной;

KDD – коэффициент при второй производной;

STS – индикатор текущего состояния адаптации.

Величины, подаваемые на входы **ZDN**, **INP** и **AMP**, должны задаваться в процентах (в диапазоне от 0 до 100). Величина управляющего воздействия на выходе **Q** формируется в диапазоне от -100 до 100.

Описание работы блока

Если вход **REG** равен 0 или 10, блок реализует функцию обычного PDD-регулятора. В первом случае используются настройки, заданные входами блока **KP**, **KD** и **KDD**, а во втором – полученные на соответствующих выходах.

Для перехода в режим автонастройки (одноразовое определение оптимальных настроек регулятора) следует установить в 1 значение входа **REG**. Для перехода в режим непрерывной адаптации (постоянное определение оптимальных настроек регулятора) нужно на вход **REG** подать значение 6. В обоих случаях процесс управления объектом осуществляется параллельно с процессом настройки регулятора.

Работа алгоритма настройки основана на подаче на вход объекта пробного гармонического сигнала. Амплитуда колебаний выходного сигнала объекта по умолчанию удерживается в пределах от 0.3% до 1%. Верхний предел амплитуды колебаний может быть изменен подачей отрицательной величины на вход **AMP**. Абсолютное значение этой величины задает ограничение на амплитуду колебаний выхода объекта.

Критерием выбора настроек является минимизация среднеквадратичной ошибки регулирования. Получаемые настройки минимизируют колебания при переходных процессах.

Реализованный алгоритм является помехоустойчивым. Он работает даже в том случае, если дисперсия шума в несколько раз превышает амплитуду пробных колебаний выходного сигнала. При настройке алгоритм исключает появление неустойчивых режимов работы контура.

Быстродействие процесса самонастройки зависит от уровня шумов, случайных возмущений, действующих на объект управления.

Индикатором хода процесса настройки регулятора является величина выхода **STS**. Он может принимать следующие значения.

- 0 – настройка завершена (или не запускалась);
- 1 – настройка регулятора завершена успешно. Получены значения оптимальных настроек;
- 2...100 – поиск оптимальных настроек. Значение выхода уменьшается от 100 до 1 и показывает близость к завершению работы алгоритма;
- 101 – невозможно провести адаптацию. Слишком много воздействий на систему или уровень шумов сравним с амплитудой гармоники на выходном сигнале объекта. Следует уменьшить внешние воздействия или попробовать увеличить уровень максимально возможной амплитуды колебаний на выходе объекта;
- 102 – невозможно провести адаптацию. Не удастся достигнуть необходимой амплитуды сигнала на выходе (от 0,3 до 1 максимальной).

ной). Следует увеличить амплитуду входа или уменьшить амплитуду выхода;

- 103 – невозможно провести адаптацию. Границы диапазона изменения управляющего сигнала не дают увеличить амплитуду сигнала на входе объекта. Следует либо изменить амплитуды, либо сместить уровень управляющего воздействия;
- 104 – невозможно провести адаптацию. Границы диапазона изменения регулируемого параметра не дают увеличить амплитуду сигнала на выходе объекта. Следует повысить ограничение на входную амплитуду или сместить входной сигнал с границ диапазона регулирования;
- 105 – невозможно провести адаптацию. Следует увеличить частоту пересчета программы.

В случае нормального завершения адаптации (значение 1 на выходе **STS**) на выходах **KP**, **KD** и **KDD** формируются новые значения настроек для регулятора. Для их использования надо присвоить новые значения соответствующим входам и задать входу **REG** значение 0. Для регулирования по полученным настройкам без присвоения их входам следует подать значение 10 на вход **REG**.

При задании жестких ограничений на амплитуду пробного сигнала алгоритм может сформировать сообщение о невозможности подобрать оптимальные настройки. Минимальное значение амплитуды следует устанавливать с учетом разрядности АЦП, коэффициента усиления и инерционности объекта.

В процессе адаптации значения настроек **KDD** и **KD** вычисляются по относительному времени – тактам вызова программы. Для использования настроек во внешних регуляторах необходимо привести их к реальной временной шкале. Значение периода вызова программы можно получить с помощью блока **TSTEP**.

Ограничения на применение

Данный алгоритм настройки **PDD**-регулятора не может быть применён к структурно неустойчивым объектам управления.

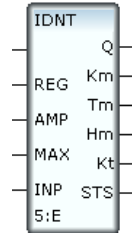
Оптимальные настройки вычисляются при соотношении времени запаздывания к постоянной времени объекта не более 0,5.

Период опроса параметра и вызова алгоритма настройки должна быть не менее 0.05 постоянной времени объекта.

Амплитуда колебаний регулируемого параметра (отклик на пробный сиг-

нал) должна быть не ниже 8 единиц кода АЦП. Например, 0.5% для 12-разрядного АЦП составляет 20 единиц.

Идентификация объекта (IDNT)



Этот блок по кривой отклика на прямоугольный импульс рассчитывает параметры модели инерционного звена первого порядка с запаздыванием.

Этот блок работает в отладчике программ инструментальной системы, а также под управлением Adaptive Control MPB, Adaptive Control Double Force MPB и Adaptive Микро TRACE MODE в реальном времени.

Входы данного блока используются для следующих целей:

REG – управление идентификацией:

1 – начать идентификацию;

0 – остановить идентификацию;

AMP – задание амплитуды пробного импульса;

MAX – задание условия снятия пробного импульса. Когда абсолютная величина разности текущего и начального значений выхода объекта превышает значение **MAX**, пробный импульс снимается;

INP – на этот вход подается сигнал с выхода анализируемого объекта.

Выходы используются следующим образом:

Q – с этого выхода пробный импульс подается на вход объекта;

Km – вычисленный коэффициент усиления модели объекта;

Tm – вычисленная постоянная времени модели объекта;

Hm – вычисленное время запаздывания модели объекта;

Kt – вычисленный коэффициент прореживания. Указывает модальному регулятору, насколько реже следует формировать управляющее воздействие по сравнению с циклом опроса;

STS – текущее состояние:

- 101 – недостаточно информации для идентификации;
- 102 – слишком большое несоответствие структуры модели и объекта управления;
- 103 – настройка не завершена. Сброс импульса произведен по длительности. Следует увеличить период канала и повторить процесс идентификации;
- 2-100 – идентификация;
- 1 – идентификация завершена успешно;
- 0 – нет задания на идентификацию.

Коэффициенты модели объекта **Tm** и **Hm** вычисляются в тактах пересчета. Для приведения их к реальной временной шкале надо использовать период вызова программы. Это значение можно получить с помощью блока **TSTEP**.

Описание работы блока

Для идентификации объекта управления необходимо задать амплитуду пробного импульса и условие его снятия.

Амплитуда пробного импульса может быть как положительной, так и отрицательной величиной. Ее следует задавать таким образом, чтобы модуль максимального значения выхода объекта был больше 4 (рекомендуемое значение - 100)

Условие снятия пробного импульса выбирается в соответствии с инерционностью объекта. Если задать его малым, то за время нарастания значения выхода объекта до указанной величины будет получено недостаточно точек для идентификации. Таких точек должно быть не менее 20.

Процесс идентификации нужно проводить на объекте в установившемся режиме, отключив регулятор. В противном случае вычисленная модель не будет адекватна объекту управления.

Для запуска идентификации надо присвоить входу **REG** значение 1. При этом на выход **Q** подается значение, заданное входом **AMP**. Когда абсолютная величина разности текущего и начального значений выхода объекта (выходное значение объекта поступает на вход **INP**) превышает значение **MAX**, пробный импульс снимается (**Q=0**). Пробный импульс снимается также в том случае, если в течение 20 тактов пересчета величина выхода объекта остается неизменной. Далее алгоритм анализирует отклик объекта после снятия пробного импульса. Когда значение выхода объекта становится меньше 3, формируются новые значения выходов **Km**, **Tm**, **Hm** и **Kt**.

За ходом процесса идентификации можно следить по значению выхода **STS**. Величина этого выхода устанавливается равной 100 после подачи на командный вход значения 1. Далее в ходе идентификации объекта его

значение уменьшается до 1. Значение выхода, равное 1, является индикатором нормального завершения определения параметров объекта.

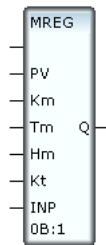
Для наиболее качественной идентификации объекта управления желательно проведение нескольких опытов при небольших отклонениях от рабочего режима, причем как на положительное приращение, так и на отрицательное с последующим усреднением коэффициентов модели.

Ограничения на применение

Данный алгоритм идентификации не может быть применен к объектам с сильно выраженными накопительными свойствами.

Чем менее инерционен объект, тем выше погрешность его идентификации.

Модальный регулятор (MREG)



Этот блок представляет собой модальный регулятор с функцией вычисления настроек по параметрам объекта управления, что предполагает его совместное использование с блоком **IDNT**.

Этот блок работает в отладчике программ инструментальной системы, а также под управлением Adaptive Control MPB, Adaptive Control Double Force MPB и Adaptive Микро TRACE MODE в реальном времени.

Входы этого блока имеют следующее назначение:

PV – задание регулятору;

Km – коэффициент передачи модели объекта;

Tm – постоянная времени модели объекта (в тактах пересчета);

Hm – время запаздывания модели объекта (в тактах пересчета);

Kt – коэффициент прореживания;

INP – регулируемая величина с максимальным значением от 4 до 100.

На выходе **Q** формируется управляющее воздействие.

Описание работы блока

По параметрам объекта, заданным на входах **Km**, **Tm** и **Hm**, вычисляются настройки модального регулятора с наблюдателем полного порядка. Наблюдатель осуществляет динамическую компенсацию запаздывания в объекте, что резко повышает качество управления. Чем меньше точность описания объекта и больше период квантования, тем менее качественно работает блок.

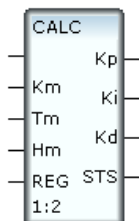
Регулятор переходит в ручной режим в случае равенства 0 входов **Km**, **Tm** и **Hm** одновременно. При этом на выход **Q** копируется значение задания регулятору (вход **PV**).

При значительном уровне шума схема модального регулятора обеспечивает фильтрацию шумовых составляющих в выходном сигнале при сохранении хорошей динамики отработки детерминированных возмущений.

Ограничения на применение

Регулятор используется только при известной математической модели объекта управления. Такая критичность вызвана тем, что выбор структуры регулятора полностью определяется имеющимся математическим описанием объекта.

Настройка PID-закона по параметрам объекта (CALC)



Этот блок рассчитывает коэффициенты PID-регулятора на основе параметров математической модели объекта первого порядка с запаздыванием.

Этот блок работает в отладчике программ инструментальной системы, а также под управлением Adaptive Control MPB, Adaptive Control Double Force MPB и Adaptive Микро TRACE MODE в реальном времени.

Входы блока имеют следующее назначение:

- Km** – коэффициент усиления модели объекта;
- Tm** – постоянная времени модели объекта;
- Hm** – время запаздывания модели объекта;

Reg – выбор типа регулятора:

0 – PI-регулятор;

1 – PID-регулятор.

На входные параметры налагаются следующие требования:

- значения входов должны быть неотрицательны;
- коэффициент передачи и постоянная времени объекта управления должны быть больше нуля;
- отношение запаздывания к постоянной времени должно лежать в пределах от 0 до 2.

Выходы данного блока используются следующим образом:

Kp – коэффициент при пропорциональной составляющей;

Ki – коэффициент при интегральной составляющей;

Kd – коэффициент при дифференциальной составляющей;

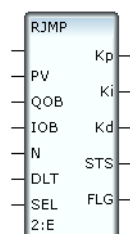
STS – флаг ошибки входных параметров:

0 – входные данные удовлетворяют требованиям;

-1 – входные данные не удовлетворяют требованиям.

Постоянная времени и запаздывание объекта управления задаются в тактах пересчета. Пусть, например, постоянная времени объекта равна 5 с, а запаздывание – 2 с. Период пересчета примем равным 0.2 с. При этом значение входа **Tm** должно быть равным $5/0.2 = 25$, а значение входа **Hm** – $2/0.2 = 10$.

Настройка PID-закона по скачку сигнала задания (RJMP)



Этот блок вычисляет настройки ПИ/ПИД регулятора. Для этого он обрабатывает два массива данных: массив значений входа объекта и его выхода. Их накопление начинается при смене задания регулятора на величину, превышающую 6% (от этой величины зависит точность вычисления настроек регулятора). Накопление прекращается, когда объект перейдет в установившееся состояние. Это означает, что выход объекта заданное число раз (значение входа **N**) не отличается от задания на величину, заданную входом **DLT**.

Этот блок работает в отладчике программ инструментальной системы, а также под управлением Adaptive Control MPB, Adaptive Control Double Force MPB и Adaptive Микро TRACE MODE в реальном времени.

Входы блока имеют следующее назначение:

- PV** – задание. При его изменении более чем на 6% и **FLG** = 100 (установившееся состояние) начинается накопление массивов;
- QOB** – вход, контролирующий выход объекта управления;
- IOB** – на этот вход подается вход объекта управления (управляющее воздействие);
- N** – вход, задающий число тактов пересчета для определения установившегося состояния. По умолчанию (при равенстве входа 0) это число равно 60 (минимальное значение – 40);
- DLT** – вход, задающий максимальное отклонение выхода объекта от задания для определения установившегося состояния. По умолчанию (при равенстве входа 0) это значение равно 0.8 (максимальное значение – 5, минимальное – 0.1);
- SEL** – выбор типа закона: 1 - ПИД, 0 - ПИ;

Выходы данного блока используются следующим образом:

- Kp** – коэффициент при пропорциональной составляющей;
- Ki** – коэффициент при интегральной составляющей;
- Kd** – коэффициент при дифференциальной составляющей;
- STS** – флаг результата последней обработки массивов:
- 0 – обработка массивов остановлена пользователем, т.е. в процессе обработки (значения от 99 до 1 на выходе **FLG**) пользователь изменил задание;
 - 100 – успешное завершение обработки массивов;
 - 101 – большое запаздывание объекта, но можно попытаться еще раз, минимизировав действие внешних возмущений в момент накопления массивов и увеличив скачок задания;
 - 102 – много данных. Следует повторить настройку, увеличив период опроса;
 - 103 – мало данных. Следует повторить настройку, уменьшив период опроса;
 - 104 – отношение запаздывания к постоянной времени объекта не входит в допустимый диапазон (от 0 до 2), но можно попытаться еще раз;
- FLG** – флаг установившегося состояния объекта:

- 0 – объект не в установившемся состоянии, требуется либо дождаться этого состояния или скорректировать значения входов **N** и **DLT**;
- 100 – объект находится в установившемся состоянии, скачок по заданию приведет к накоплению массивов и вычислению новых настроек регулятора;
- 99 ... 0 – идет обработка накопленных массивов. Значение 100 на этом выходе в момент переходного процесса свидетельствует о накоплении массивов, и когда объект войдет в установившееся состояние, значение входа будет уменьшаться до нуля.

Значения входов **PV**, **QOB** и **IOB** следует задавать в процентах.

Для получения более точных настроек следует проделать серию опытов по смене задания.

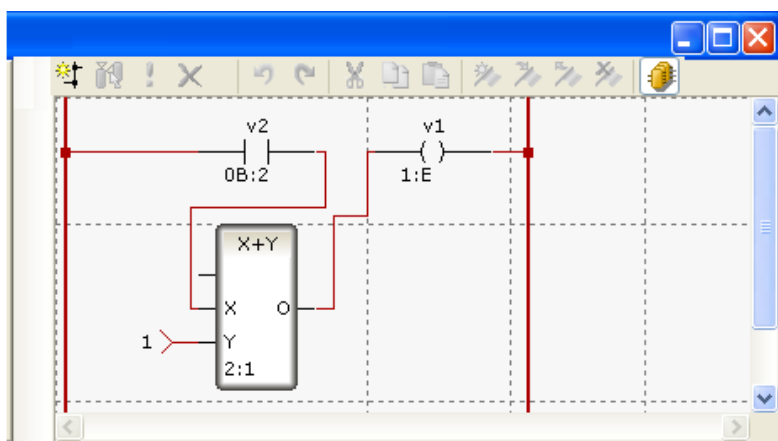
При смене задания в процессе накопления массивов данные продолжают накапливаться. Однако в этом случае коэффициенты регулятора не вычисляются.

Оптимальная обработка блоков в программе будет следующей: контроль выхода объекта, вычисление управляющего воздействия, пересчет.

Блок может использоваться и при ручном управлении объектом. В данном случае надо привести объект в установившееся состояние. Далее следует на входе **PV** установить значение входа **QOB**. Когда на выходе **FLG** появится 100, следует изменить задание и вручную подогнать объект к этому значению. После этого начнется обработка массивов и вычисление настроек регулятора.

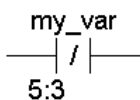
Редактирование LD-программ

LD-программа представляет собой диаграмму последовательно выполняемых **функциональных блоков**. На рисунке показан вид программы в LD-редакторе.



Функциональный блок – это графическое изображение вызова встроенной функции **Техно LD** (LD-блока), функции (функции-блока), определенной пользователем, или FBD-блока.

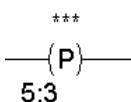
Вид LD-блока показан на следующем рисунке.



Над блоком выводится имя **связанной переменной** (**my_var** на рисунке).

Связанной переменной называется переменная, от значения которой зависит выполняемое блоком действие или значение которой устанавливается в процессе выполняемого блоком действия. Связанная переменная задается пользователем.

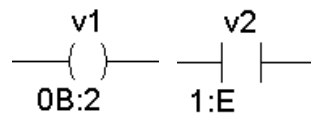
Если связанная переменная не задана, над блоком отображаются три звездочки:



В качестве изображения блока используется обозначение выполняемой

этим блоком функции (**I/I** и **(P)** на рисунках). Отрезок слева обозначает вход блока, отрезок справа – выход. Все LD-блоки имеют один вход (**in**) и один выход (**out**).

Под блоком выводится его номер и, после двоеточия, номер следующего выполняемого блока (**5:3** на рисунках выше). Номера блоков задаются последовательно при их размещении в рабочем поле редактора; номера следующих выполняемых блоков определяются автоматически при размещении других блоков и соединении входов и выходов блоков (образовании диаграммы). На блоке, который выполняется первым в программе, после его номера отображается символ **B**; на блоке, который выполняется последним, – символ **E**:



Используемые в программе FBD-блоки, а также функции и функции-блоки отображаются на LD-диаграмме в виде, аналогичном виду функциональных блоков в FBD-редакторе.

Шины изображаются на диаграмме в виде вертикальных линий. В **Техно LD** используются две **основные** шины (левая и правая) и **вспомогательные** шины. Между основными шинами размещаются все функциональные блоки LD-программы; на вспомогательные шины могут замыкаться выходы блоков, расположенных один над другим.

Шины имеют следующее назначение:

- значение левой основной шины всегда равно 1 (аналог положительной шины питания);
- значение правой основной шины и вспомогательной шины формируется как логическая сумма (**OR**) значений выходов блоков, связанных с этой шиной

В процессе выполнения программы блоки пересчитываются последовательно в соответствии с их номерами. Значение правой основной шины и вспомогательной шины равно логической сумме значений выходов блоков, пересчитанных на текущий момент времени выполнения программы.


LD-программа может выступать в роли основной программы, функции и функции-блока.

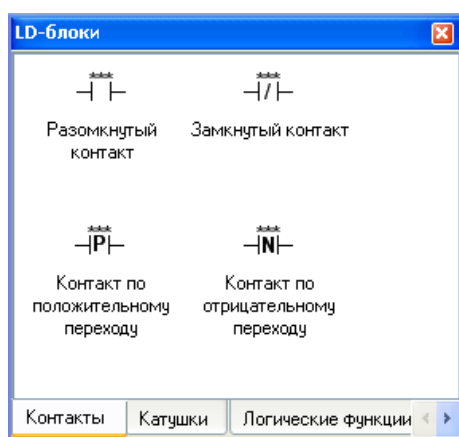
Для создания LD-программы и подключения ее к проекту нужно выполнить следующие операции:

- разместить необходимые функциональные блоки в рабочем поле LD-редактора;
- задать необходимые связи (образовать LD-диаграмму);

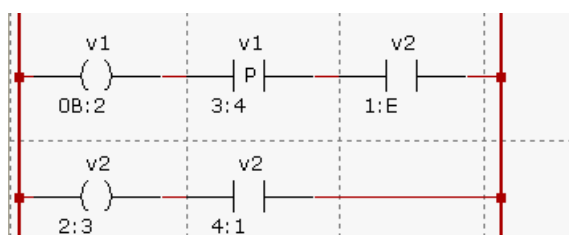
- задать аргументы, переменные и константы программы;
- привязать входы/выходы LD-диаграммы к аргументам, переменным и константам программы и задать связанные переменные;
- скомпилировать программу.

Размещение блоков в рабочем поле LD-редактора

Размещение LD-, FBD- и пользовательских функциональных блоков в рабочем поле LD-редактора производится с помощью навигатора аналогично размещению блоков в FBD-редакторе (см. **Размещение FBD-блоков в рабочем поле редактора**). Открыть/закрыть окно LD-навигатора блоков можно с помощью кнопки  панели инструментов LD-редактора. Вид LD-навигатора показан на следующем рисунке.



При размещении блоков и задании связей номера следующих выполняемых блоков автоматически устанавливаются таким образом, чтобы при запуске программы первыми выполнялись блоки (в соответствии с их номерами), расположенные в первом (самом левом) столбце диаграммы, затем – во втором и т.д. (см. порядок выполнения программы, показанной на следующем рисунке).




Первоначально заданный порядок выполнения в дальнейшем (после задания всех связей) не зависит от расположения блоков на диаграмме. Например, порядок выполнения программы, показанной на рисунке выше, не изменится, если блок номер 4 переместить в свободную ячейку справа.

Редактирование LD-диаграммы

Редактирование LD-диаграмм производится аналогично редактированию FBD-диаграмм (см. **Редактирование диаграммы FBD-блоков**), за исключением работы с шинами, которые являются особенностью языка **Техно LD**.

Основные шины автоматически отображаются в рабочем поле LD-редактора при размещении первого LD-блока.

Для размещения вспомогательной шины нужно нажать ЛК на кнопке  панели инструментов LD-редактора или выполнить команду **Вертикальная шина** из контекстного меню, установить курсор в нужное место диаграммы и нажать ЛК. Информация об основной/дополнительной шине отображается во всплывающей подсказке:

| | | |
|--|--|--|
| <p>Левая шина</p> <p>Выходы</p> <ul style="list-style-type: none"> • LD блок 2:0 (I P I) | <p>Правая шина</p> <p>Входы</p> <ul style="list-style-type: none"> • LD блок 2:0 (I P I) | <p>Вертикальная шина 6</p> <p>Входы</p> <ul style="list-style-type: none"> • LD блок 1:0 (I P I) |
|--|--|--|

Для задания связей с шинами используется метод drag-and-drop, удаление таких связей производится аналогично удалению межблочных связей.

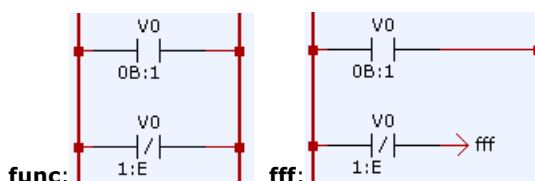
Привязка входов и выходов LD-диаграммы и задание связанных переменных

Привязка входов/выходов LD-диаграммы выполняется аналогично привязке входов/выходов FBD-диаграммы (см. **Привязка входов и выходов FBD-диаграммы**).

Функция (SFC-условие) на языке **Техно LD** возвращает значение правой основной шины, если ни один из выходов LD-диаграммы не привязан к имени функции (SFC-условия).

Пример

Пусть заданы следующие LD-функции **func** и **fff** (**VO=1**):



Значение правой основной шины для обеих функций равно 1, однако результат вызова функций различен:


```
PROGRAM
    VAR VAR_001 : INT; END_VAR
```



```

VAR_001 = func (); //VAR_001=1
VAR_001 = fff (); //VAR_001=0
END_PROGRAM

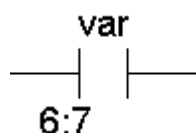
```

Для перехода в режим выбора связанной переменной нужно выделить блок и далее или нажать ЛК или нажать кнопку  панели инструментов или нажать ПК и выполнить команду **Привязать** из контекстного меню.

Описание LD-блоков

Раздел 'Контакты'

Разомкнутый контакт (| |)

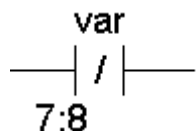


Если $\text{var} < 0$ и $\text{in} < 0$, то $\text{out} = 1$.

Если $\text{var} < 0$, а $\text{in} = 0$, то $\text{out} = 0$.

Если $\text{var} = 0$, то $\text{out} = 0$.

Замкнутый контакт (/|)

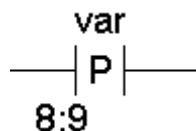


Если $\text{var} = 0$, а $\text{in} < 0$, то $\text{out} = 1$.

Если $\text{var} = 0$ и $\text{in} = 0$, то $\text{out} = 0$.

Если $\text{var} < 0$, то $\text{out} = 0$.

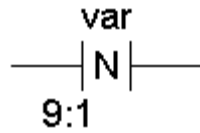
Контакт по положительному переходу (P|)



Если $\text{in} < 0$, а var меняет свое значение с 0 на любое ненулевое, то на

один (следующий) такт пересчета **out**=1. Во всех остальных случаях **out**=0.

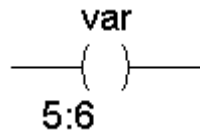
Контакт по отрицательному переходу ($\downarrow N$)



Если $in \triangleleft 0$, а **var** меняет свое значение с любого ненулевого на 0, то на один (следующий) такт пересчета **out**=1. Во всех остальных случаях **out**=0.

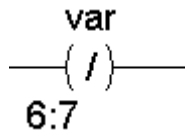
Раздел 'Катушки'

Катушка (())



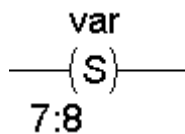
Если $in \triangleleft 0$, **var**=**out**=1; если $in=0$, **var**=**out**=0.

Инверсная катушка (/)

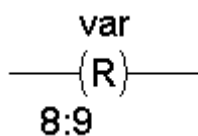


Если $in \triangleleft 0$, **var**=**out**=0; если $in=0$, **var**=**out**=1.

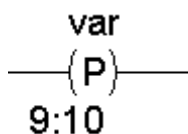
Катушка установки ((S))



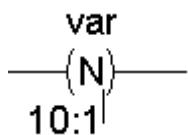
Связанная переменная **var** принимает значение 1 при подаче на вход любого ненулевого значения. В дальнейшем **var** не зависит от значения входа. При этом во всех случаях **out** = 1.

Катушка сброса ((R))

Связанная переменная **var** принимает значение 0 при подаче на вход любого ненулевого значения. В дальнейшем **var** не зависит от значения входа. При этом во всех случаях **out** = 1.

Катушка положительного перехода ((P))

Если значение входа изменяется с 0 на любое ненулевое, то на один (следующий) такт пересчета **var**=1. При этом если **in**>0, **out**=1; если **in**=0, **out**=0.

Катушка отрицательного перехода ((N))

Если значение входа изменяется с любого ненулевого на 0, то на один (следующий) такт пересчета **var**=1. При этом если **in**>0, **out**=1; если **in**=0, **out**=0.

Создание пользовательских функциональных блоков

Для создания в TRACE MODE 6 пользовательского функционального блока достаточно создать в программе функцию или функцию-блок на любом из встроенных языков. Такая функция отображается в виде блока в разделе **Пользовательские блоки** навигаторов FBD- и LD-редакторов.

Пользовательские блоки могут быть использованы только в той основной программе (и ее компонентах), в рамках которой они созданы. Использовать запрограммированный блок в функции, из которой этот блок создан, нельзя.

Ограничения по использованию пользовательских блоков в основной программе описаны в разделе **Пользовательские функции Техно ST / Вызов функции и функции-блока..**




Аргумент типа **ВХОД** порождает вход пользовательского блока, аргумент типа **ВЫХОД** – выход. Аргумент типа **ВХОД/ВЫХОД** порождает одновременно вход и выход пользовательского блока.

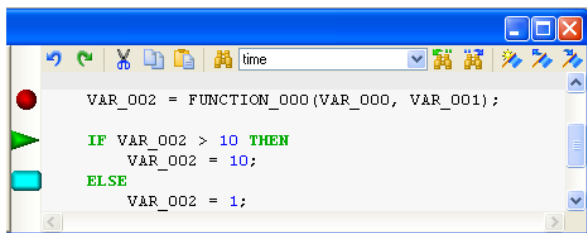
Отладка программ

Средства отладки (в том числе удаленной) включают в себя несколько режимов непрерывного и пошагового выполнения программы с возможностью установки точек останова. Для запуска требуемого режима отладки можно использовать команды меню **Программа** главного меню или аналогичные команды панели инструментов отладчика. Для вывода служебных сообщений отладчика и компилятора предусмотрены специальные окна.

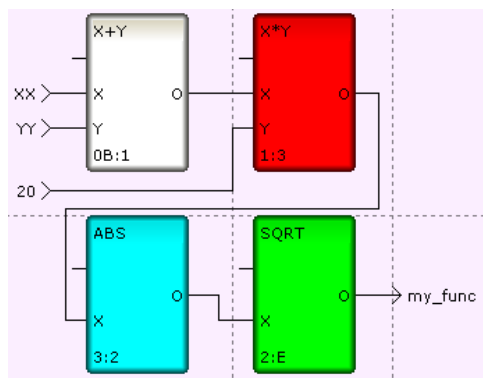
Отладка программы возможна только после ее успешной компиляции.

Параметры отладчика программ настраиваются в соответствующем диалоге (см. **Настройка редакторов и отладчика**).

В листинге текстовых программ точка останова обозначается значком , закладка – значком . При пошаговой отладке текущий шаг обозначается значком .

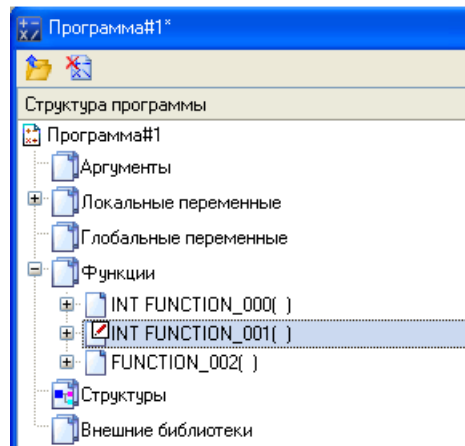


В программах, заданных в графическом виде, закладки, текущий шаг и точки останова обозначаются соответствующим цветом:



В пошаговом режиме отладки в окне структуры программы автоматиче-

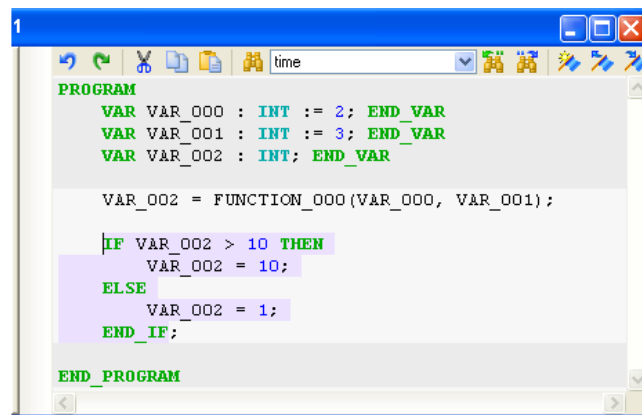
ски выделяется текущий компонент программы:



Автоматическое выделение конструкций языка












В текстовых редакторах (**IL** и **ST**) при установке курсора в позицию ключевого слова, задающего начало или конец конструкции языка, вся конструкция автоматически выделяется. Чтобы снять выделение, нужно нажать ЛК в строке, не содержащей таких ключевых слов.

На следующем рисунке показано, как выглядит выделение конструкции **IF...END_IF** в редакторе **ST**:



Меню 'Программа' и панель инструментов отладчика

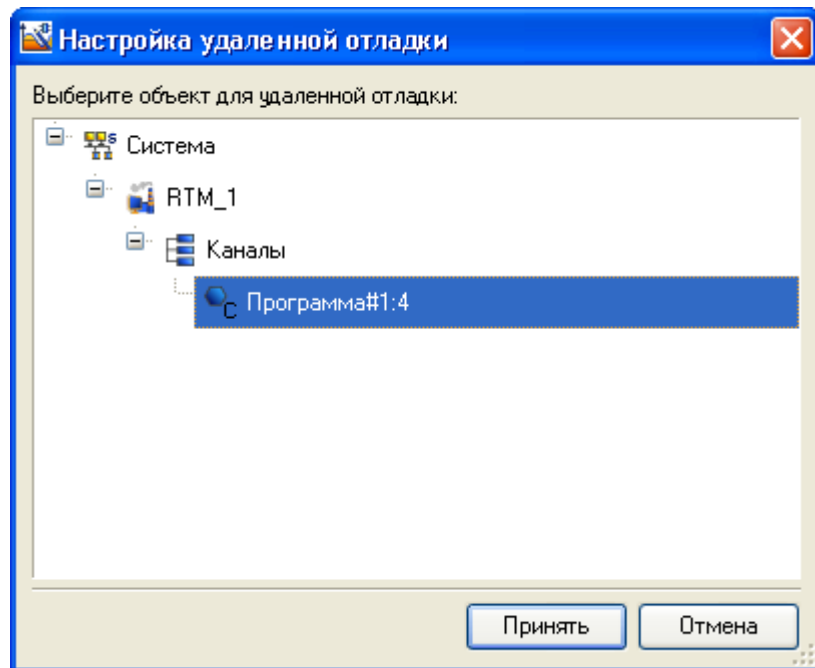
Меню **Программа** главного меню и панель инструментов отладчика содержат следующие команды для запуска требуемого режима отладки и настройки параметров редакторов программ:

-  **Компиляция (F7)** – запустить компиляцию программы;
-  **Установить/удалить точку останова (F9)** – установить/удалить точку останова программы;
-  **Удалить все точки останова (Ctrl+Shift+F9)** – удалить все точки останова программы;
-  **Удаленная отладка** – переключатель вида отладки: **локальная** (отжатое состояние) или **удаленная** (нажатое состояние) (см. **Удаленная отладка**);
-  **Старт (F5)** – запустить выполнение программы в непрерывном режиме;
-  **Выполнять до курсора (Ctrl+F10)** – запустить выполнение программы в непрерывном режиме до текущей позиции курсора;
-  **Трассировка (F11)** – запустить пошаговое выполнение программы с пошаговым выполнением вызываемых функций;
-  **Шаг (F10)** – запустить пошаговое выполнение программы с выполнением вызываемых функций в непрерывном режиме;
-  **Выйти из функции (Shift+F11)** – выполнить текущую функцию/программу в непрерывном режиме (доступно в режиме отладки);
-  **Стоп (Shift+F5)** – выйти из режима отладки;
-  **Посмотреть значение переменной (Shift+F9)** – открыть диалог **Быстрый просмотр** (доступно в режиме отладки);
-  **Переменные** – открыть/закрыть окно переменных;
-  **Стек** – открыть/закрыть окно стека вызовов функций;
-  **Сообщения** – открыть/закрыть окно сообщений компилятора и отладчика.

Удаленная отладка

При удаленной отладке (см. **Меню 'Программа' и панель инструментов отладчика**) ИС запрашивает значения аргументов и глобальных переменных программы у узла, выполняемого под управлением монитора на удаленном компьютере или в контроллере. Это позволяет отлаживать программу на реальных значениях обрабатываемых сигналов.

При переходе к данному виду отладки нужно выбрать программу в следующем диалоге:



Окно просмотра переменных

Данное окно включает в себя в виде вкладок 5 окон просмотра текущих значений переменных.



В окне **Локальные** отображаются переменные текущего программного компонента (в том числе переменные объектов, определенных в текущем компоненте, – на рисунке таким объектом является **var_002**):

| Переменные | | | | |
|------------|-------|------------|-----------------|--------------------|
| Переменная | Тип | Десятичный | Шестнадцатичный | Двоичный |
| var_000 | INT | 10 | 16#000A | 2#0000000000001010 |
| var_001 | INT | 20 | 16#0014 | 2#0000000000010100 |
| var_002 | m_str | | | |
| m_type | INT | 10 | 16#000A | 2#0000000000001010 |
| var_003 | INT | -24 | 16#FFFFFFE8 | 2#1111111111101000 |

Аргументы **Локальные** Глобальные Тек. структура Просмотр

В окнах **Аргументы**, **Глобальные** и **Текущая структура** отображаются соответственно аргументы, глобальные переменные и переменные объекта.

При пошаговой отладке программы в окнах **Локальные** и **Глобальные**

можно вручную задать значения переменных (для перехода к заданию значения нужно дважды нажать ЛК в поле значения или выполнить команду **Изменить значение** из контекстного меню). Для восстановления значений аргументов по умолчанию контекстное меню окна **Аргументы** содержит команды  **Восстановить значение по умолчанию** и  **Восстановить все значения по умолчанию**.

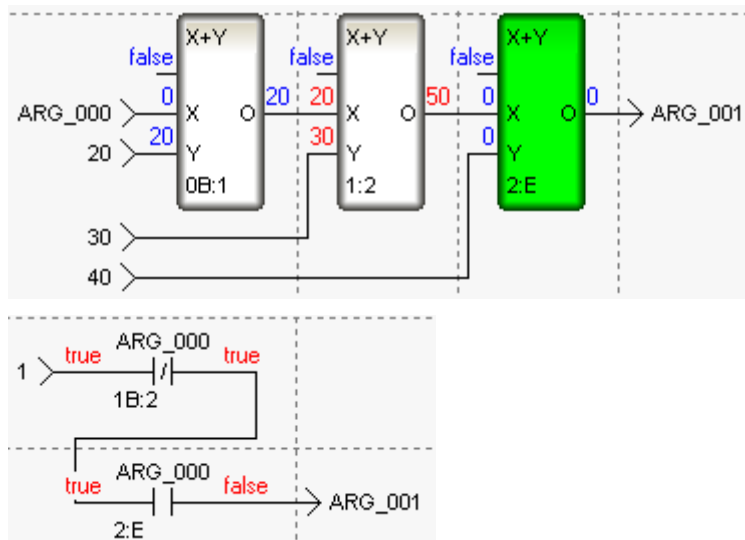
В окне **Просмотр** отображаются переменные и выражения, заданные пользователем. По умолчанию оно пустое. Список переменных и выражений для просмотра задается с помощью диалога **Быстрый просмотр**. Корректировать список можно в самом окне с помощью команд контекстного меню, которое выводится на экран при нажатии ПК в области окна. Меню содержит следующие команды:

Вставить – добавить строку в список;

Удалить – удалить выделенную строку из списка (аналог нажатия клавиши **Del** на клавиатуре);


Переименовать – редактировать выделенную строку списка.

Текущие значения переменных отображаются на диаграммах FBD и LD:

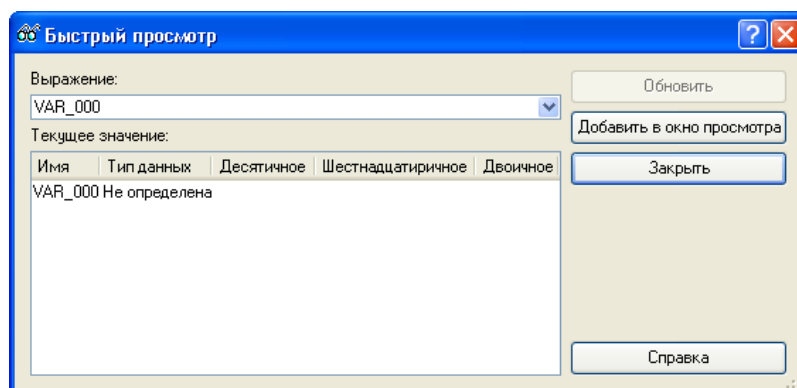


Диалог 'Быстрый просмотр'

Данный диалог используется для оценки значения переменных и выражений в процессе отладки, а также для задания списка переменных и выражений для отображения в окне-вкладке **Просмотр** окна просмотра переменных.

Для входа в диалог нужно в режиме отладки нажать кнопку  инстру-

ментальной панели или выполнить команду **Посмотреть значение переменной** меню **Программа** (при отладке в цикле диалог недоступен).



В поле **Выражение** задается переменная или выражение, для оценки переменной или выражения нужно нажать кнопку **Обновить**, для добавления переменной или выражения в окно **Просмотр** – кнопку **Добавить**.

Окно стека вызовов функций

В данном окне выводится информация о пути, по которому вызывается текущая функция. Например, в окне, показанном на рисунке ниже, выведена информация о том, что основная программа (**\$main**) обратилась к функции **fff**, которая, в свою очередь, вызвала функцию **fff_1**. Функция **fff_1** является текущей.

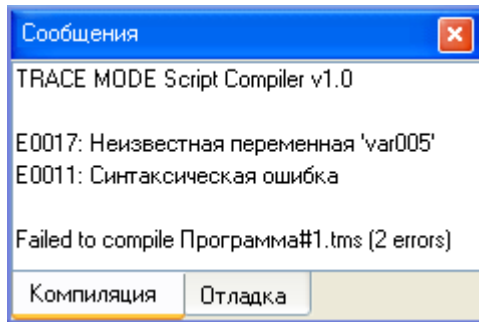
| Имя | Тип | Десятичный | Шестнадцатичный | Двоичный |
|--------|-----|------------|-----------------|--------------------|
| fff_1 | INT | 10 | 16#000A | 2#0000000000001010 |
| fff | INT | | | |
| \$main | | | | |

Окно 'Сообщения'

Это окно содержит в виде вкладок 2 окна для вывода служебных сообщений отладчика и компилятора.

Окно сообщений компилятора

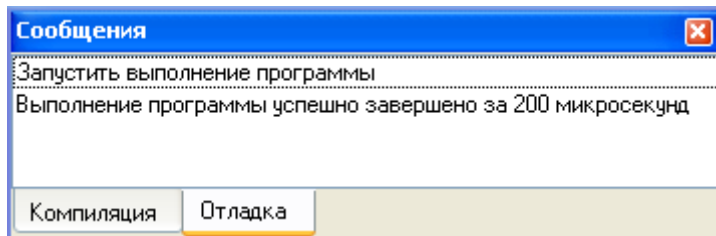
В это окно выводятся сообщения об ошибках, обнаруживаемых в процессе компиляции программы.



Если нажать ЛК на строке описания ошибки, в программе выделяется строка, содержащая эту ошибку.

Окно сообщений отладчика

В это окно выводятся сообщения о запуске/останове и времени выполнения программы, а также об ошибках, обнаруживаемых в процессе отладки.



Глава 8

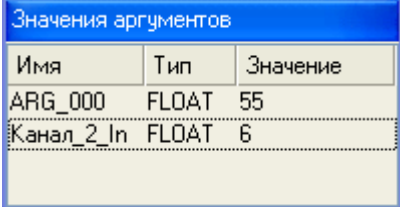
Разработка графического интерфейса

Редактор представления данных

Режимы работы РПД

РПД может находиться в одном из следующих режимов (режимы переключаются с помощью инструментов панели **Графические элементы** – см. **Главное меню и панели инструментов РПД**):

- режим **размещения** предназначен для заполнения графических слоев экранов графическими элементами;
- режим **редактирования** предназначен для внесения изменений в созданные ранее графические экраны;
- в режиме **эмуляции** проверяется зависимость работы графического экрана от значений его аргументов. В этом режиме открывается диалог просмотра/задания значений аргументов:



| Имя | Тип | Значение |
|------------|-------|----------|
| ARG_000 | FLOAT | 55 |
| Канал_2_In | FLOAT | 6 |

Кроме того, в РПД предусмотрено два режима отображения графических экранов – обычный (в окне) и полноэкранный.

В режимах размещения и редактирования текущие координаты курсора отображаются в строке статуса (внизу справа). Там же отображается состояние флага **Располагать по сетке** (“snap ON” / “snap OFF” – см. **Задание параметров РПД**). При выделении ГЭ (см. **Выделение ГЭ**) вместо координат курсора в строке статуса отображаются координаты точки привязки (см. **Размещение ГЭ**), размеры и угол поворота ГЭ.

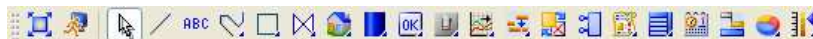
Работа с несколькими дисплеями

TRACE MODE поддерживает вывод экранов на несколько дисплеев, при этом координаты на дополнительном дисплее будут отрицательными, если при конфигурировании он расположен слева (выше) основного.




Чтобы на полиэкранном рабочем столе поддерживался полноэкранный режим, надо при сохранении проекта для МРВ в свойствах панели задач Windows установить флаг **Автоматически скрывать панель задач**.


Главное меню и панели инструментов РПД

Панель инструментов 'Графические элементы'



С помощью инструментов этой панели выбираются графические элементы для размещения их в графических слоях экранов (см. **Размещение ГЭ**). При выборе ГЭ редактор переходит в режим размещения (см. **Режимы работы РПД**).


С помощью кнопки  данной панели можно перейти в режим редактирования, с помощью кнопки  – в режим эмуляции (для выхода из режима эмуляции надо нажать кнопку  повторно).

Кнопка  предназначена для переключения режима отображения графических экранов (обычный/полноэкранный).

Меню и панель инструментов 'Правка'



Меню и панель инструментов **Правка** содержат ряд типовых инструментов для редактирования графических экранов. Данные инструменты доступны также из контекстного меню ГЭ (см. **Контекстные меню РПД и Типовые средства редактирования**).

В списке (**Масштаб**) можно выбрать предустановленный масштаб или вручную задать произвольный. Для выбора предустановленного масштаба можно также использовать кнопки  или сочетания клавиш **CTRL+ПЛЮС/МИНУС** на цифровой клавиатуре. При выделении некоторой области **AxВ** экрана с помощью мыши с удержанием клавиши **Z** экран масштабируется в $\text{MIN}\{C/A, D/B\}$ раз, где **CxD** – размеры видимой области. Во всех случаях масштабирование производится относительно центра видимой области.

Меню 'Сервис' и панель инструментов 'Топология экрана'



Данная панель инструментов и меню содержат команды для позиционирования и тиражирования ГЭ (см. **Позиционирование ГЭ, Тиражирование ГЭ**).

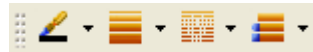
Панель инструментов 'Параметры текста'







В режиме редактирования с помощью типовых инструментов данной панели (см. **Типовые средства редактирования**) задаются параметры текста в выделенном графическом элементе (выделенной группе ГЭ). Данные команды применимы только к такому тексту, который может быть введен/отредактирован с помощью клавиатуры.

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры линии'



В режиме редактирования с помощью инструментов этой панели задаются параметры линии (линии контура) выделенного графического элемента (выделенной группы ГЭ):



-  – цвет линии;
-  – толщина линии;
-  – стиль линии;
-  – край линии (**плоский, квадратный, круглый**).

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Параметры заливки'



В режиме редактирования с помощью инструментов этой панели задаются параметры заливки выделенного графического элемента (выделенной группы ГЭ):

-  – цвет заливки;
-  – стиль заливки.

Вид ГЭ при его размещении в графическом слое зависит от параметров, установленных с помощью инструментов этой панели.

Панель инструментов 'Ресурсные библиотеки'



Инструменты данной панели предназначены для операций с библиотеками строк, рисунков и других ресурсов, которые могут быть использованы при разработке графических экранов (см. **Операции с ресурсными библиотеками**).

Меню 'Вид'

Команды этого меню управляют видимостью редактора аргументов экрана, окна **Слои**, таблицы графических элементов и окна **Избранное** (см. **Создание и удаление графических слоев**, **Таблица 'Графические элементы'** и **Библиотека избранных ГЭ и библиотека избранных eГЭ**), а также панелей инструментов **Топология экрана** и **Параметры текста**.

Меню **Вид** содержит также команду **Параметры экрана** (см. **Задание параметров графического экрана**).

Задание параметров РПД

Окно **Настройки**, вызываемое из меню **Файл**, в разделе **РПД** содержит диалог задания параметров РПД.

Этот диалог содержит следующие инструменты:

- **Подсвечивать при наведении мыши** – если этот флаг установлен, при наведении курсора мыши на ГЭ его вершины (узловые точки) выделяются цветом, заданным в поле справа. Не следует путать эту функцию с функцией выделения ГЭ (см. **Выделение ГЭ**);
- **Открывать свойства автоматически** – от этого флага зависит режим РПД после размещения графического элемента на экране (см. **Размещение ГЭ**);
- **Располагать по сетке** – если этот флаг установлен, при размещении, перемещении и масштабировании вершины прямоугольника, ограничивающего ГЭ, располагаются в узлах сетки. При размещении в узлах сетки располагаются также узловые точки ГЭ.

Установка этого флага не является командой автоматического упорядочивания ГЭ на графических экранах.

- **Показать сетку** – если этот флаг установлен, сетка отображается на графических экранах. Для установки/сброса флага при редактировании можно использовать сочетание клавиш CTRL+G;
- **Шаг сетки** – задание шага сетки в пикселях (3-100);

- **Цвет сетки** – выбор цвета сетки;
- **Цвет контура выделения** – выбор цвета прямоугольника, ограничивающего ГЭ при выделении в режиме редактирования;
- **Выделение при редактировании** – если этот флаг установлен, при масштабировании/повороте ГЭ выделение сохраняется, в противном случае – скрывается;
- **Использовать устаревшие функции** – при установке этого флага доступны некоторые опции предыдущих версий TRACE MODE (см. **Группа ГЭ ‘Свободные формы’**);
- **Выделение в реальном времени** – если этот флаг установлен, для ГЭ может быть сконфигурировано выделение в реальном времени по нажатию ЛК (см. **Статические атрибуты ГЭ**);
- **Цвет рамки** – цвет контура выделения ГЭ в реальном времени;
- **Толщина рамки** – толщина контура выделения ГЭ в реальном времени (в пикселях);
- **Стиль рамки** – стиль контура выделения ГЭ в реальном времени;
- **Отступ** – отступ контура выделения ГЭ в реальном времени.

Значения параметров выделения ГЭ в реальном времени сохраняются в файле **gr_settings.xml**.

Контекстные меню РПД

При нажатии ПК в произвольной точке графического экрана появляется меню, содержащее стандартные команды для работы с буфером обмена (см. **Типовые средства редактирования**).

При нажатии ПК на выделенном ГЭ (группе ГЭ) появляется меню, которое, помимо стандартных команд для работы с буфером обмена и команд управления расположением подслоев (см. **Позиционирование ГЭ**), содержит следующие команды:

- **Свойства объекта** – открыть окно **Свойства объекта** (см. **Задание типовых свойств ГЭ**);
- **Переместить в слой** – переместить ГЭ/группу ГЭ в слой, выбранный в подменю данной команды;
- **Копировать в избранное** – см. **Библиотека избранных ГЭ и библиотека избранных eГЭ**;
- **Переместить/масштабировать** – перейти в режим перемещения и масштабирования ГЭ/группы ГЭ (см. **Перемещение и масштабирование ГЭ**);
- **Повернуть** – перейти в режим поворота ГЭ/группы ГЭ (см. **Поворот ГЭ**);
- **Редактировать узловые точки** – перейти в режим позиционирования узловых точек (эта команда доступна в контекстных меню

ГЭ групп **Ломаные** и **Кривые**);

- **Восстановить размер** – восстановить оригинальный размер ГЭ (эта команда доступна в контекстных меню ГЭ **Растровое изображение** и **Видеоклип**).

Некоторые окна РПД имеют свои контекстные меню, которые появляются на экране при нажатии ПК в области окна.

Таблица ‘Графические элементы’

Чтобы открыть/закрыть таблицу **Графические элементы**, нужно выполнить команду **Таблица графических элементов** из меню **Вид**.

Данная таблица представляет собой список всех графических элементов, расположенных в редактируемом экране:

| Тип | Выделен | Подсказка | Позиция | Размер | Управление | Заливка | Трансфор | Динами | Слой |
|-----|---------|-----------|----------|---------|------------|---------|----------|--------|-------|
| АВС | | | 118, 167 | 60 x 30 | 77 | | | | Слой1 |
| OK | | | 240, 85 | 60 x 30 | | | | | Слой1 |
| △ | | | 111, 71 | 60 x 30 | | | | | Слой1 |
| △ | ✓ | | 345, 207 | 60 x 30 | | | | | Слой3 |

В таблицу может быть также выведена информация о следующих параметрах ГЭ:

- **Тип** – иконка ГЭ;
- **Выделение** – выделенный графический элемент отмечается в этом столбце знаком ✓. Чтобы выделить (снять выделение) ГЭ, нужно дважды нажать ЛК в этом поле. С помощью этого механизма можно выделить на экране группу ГЭ (см. также **Выделение ГЭ**);
- **Подсказка** – всплывающая подсказка (см. **Статические атрибуты ГЭ**);
- **Позиция** – координаты точки привязки ГЭ (**X, Y** в пикселях относительно верхнего левого угла экрана). При двойном нажатии ЛК в этом поле открывается диалог **Геометрия**:

Геометрия

X: 132 Y: 55

Масштаб по X: 1 Масштаб по Y: 1

Угол (градусы): 0

Ширина: 60 Высота: 30

Готово Отмена

В этом диалоге могут быть заданы следующие параметры ГЭ:

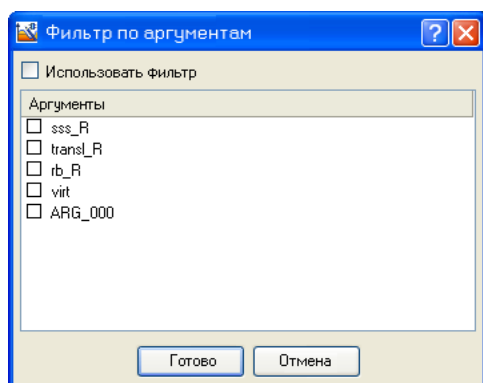
- **X** – абсцисса точки привязки ГЭ;
- **Y** – ордината точки привязки ГЭ;
- **Масштаб по X** – масштаб ГЭ по оси абсцисс;
- **Масштаб по Y** – масштаб ГЭ по оси ординат;
- **Угол (градусы)** – угол поворота ГЭ в градусах (см. **Поворот ГЭ**);
- **Ширина** – произведение (**Ширина * Масштаб по X**) равно ширине ГЭ (в пикселях);
- **Высота** – произведение (**Высота * Масштаб по Y**) равно высоте ГЭ (в пикселях);

Параметры **Угол (градусы)**, **Ширина** и **Высота** доступны не для всех ГЭ.

- **Размер** – размеры ГЭ (**X** и **Y** в пикселях);
- **Управление** – графический элемент, для которого задана хотя бы одна функция управления, отмечается в этом столбце. В этом же столбце отображается код доступа к использованию функции управления (см. **Функции управления ГЭ**);
- **Заливка** – графический элемент, для которого задана динамическая заливка (см. **Динамическая заливка ГЭ**), отмечается в этом столбце;
- **Трансформация** – графический элемент, для которого задан хотя бы один вид динамической трансформации (см. **Динамические свойства ГЭ**), отмечается в этом столбце;
- **Динамизация** – графический элемент, у которого динамизирован хотя бы один атрибут (см. **Динамизация атрибута ГЭ**), отмечается в этом столбце;
- **Слой** – имя слоя, в котором размещен ГЭ. Чтобы переместить ГЭ в другой слой, нужно дважды нажать ЛК в этом поле и выбрать слой в списке. При таком перемещении ГЭ располагается в самом нижнем подслое выбранного слоя (см. **Размещение ГЭ**).

Набором выводимых параметров можно управлять с помощью контекстного меню таблицы (подменю **Отображать параметры**). По умолчанию таблица содержит информацию обо всех вышеперечисленных параметрах.

Контекстное меню содержит стандартные команды выделения и работы с буфером обмена, а также команду **Фильтр по аргументам**, при выполнении которой открывается следующий диалог:



При выделении аргументов в списке, установке флага **Использовать фильтр** и нажатии кнопки **Готово** таблица отображает только те графические элементы, которые связаны с выбранными аргументами. При переключении между экранами флаг **Использовать фильтр** и сам фильтр сбрасываются.

Операции с графическими экранами

Задание параметров графического экрана

Чтобы открыть окно параметров редактируемого графического экрана, нужно выполнить команду **Параметры экрана** меню **Вид** (см. **Главное меню и панели инструментов РПД**) или дважды нажать ЛК в области экрана, не содержащей ГЭ. Окно параметров экрана автоматически заменяется окном свойств выделяемого ГЭ (см. **Выделение ГЭ**, а также **Задание типовых свойств ГЭ**).

Окно параметров экрана содержит следующие инструменты:

- **Размеры** – задание размера экрана в пикселях. Размер можно выбрать из нескольких стандартных или задать свой с помощью опции **Произвольно** и полей **Ширина** и **Высота**;
- **Масштабировать содержимое** – если TRUE, размещенные на экране ГЭ масштабируются пропорционально изменению размеров экрана;
- **Фон** – выбор типа фона:
 - **Цвет** – при выборе этой опции в разделе **Фон** доступен атрибут **Цвет**; при нажатии ЛК в поле **Значение** этого атрибута на экране появляется стандартный диалог выбора цвета;
 - **Изображение** – при выборе этой опции в разделе **Фон** доступен атрибут **Изображение**; при нажатии ЛК в поле **Значение** этого атрибута на экране появляется диалог выбора рисунка из ресурсной библиотеки;
- **Положение источника света (%) X** и **Положение источника света (%) Y** – положение источника света относительно экрана (угол с осями X и Y в процентах). Значение (50, 50) соответствует расположению источника света на нормали к экрану;
- **Код доступа** – код доступа к экрану. Права на доступ к экранам задаются для пользователя в виде маски в разделе **Доступ / Экраны** канала **Пользователь** (см. **Канал класса ПОЛЬЗОВАТЕЛЬ**). При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к экрану разрешен, в противном случае – запрещен;

Если пользователи в системе не заданы, значение кода доступа не учитывается.

Если ни один бит маски канала **Пользователь** не выделен, доступ к экрану разрешен только при значении кода доступа 0.

Если для пользователя задана некоторая маска, то для его доступа к экрану с кодом 0 нужно установить соответствующий флаг (см. **Канал класса ПОЛЬЗОВАТЕЛЬ**).

- **Горячая клавиша** – меню выбора функциональной горячей клавиши (**F2–F12**). При запуске проекта в реальном времени нажатие заданной клавиши будет сопровождаться переходом на данный экран;
- **Всплывающее окно** – если TRUE, экран определяется как всплывающий и отображается в списке всплывающих экранов MPB; характеристики всплывающего экрана задаются с помощью следующих атрибутов:
 - **Показывать всплыв. окно при запуске** – если TRUE, всплывающий экран отображается при загрузке узла в MPB;
 - **Нач. позиция всплыв. окна, X** и **Нач. позиция всплыв. окна, Y** – положение всплывающего экрана по осям X и Y при загрузке узла в MPB;
 - **Сохранить размер** – если TRUE, размер экрана не может быть изменен;
- **Загружать** – см. **Экономичная загрузка** в разделе **Особенности вызова графического экрана**.
- **Показать в меню экранов** – если TRUE, команда перехода на экран добавляется в меню и панель инструментов **Экраны MPB** (см. **Профайлер с поддержкой графических экранов**);
- **Верхний объект, Нижний объект, Левый объект, Правый объект** – разделы конфигурирования соответственно верхнего, нижнего, левого и правого колонтитулов экрана. В качестве колонтитулов используются графические объекты. Раздел содержит следующие атрибуты:
 - **Выравнивание** – выравнивание объекта;
 - **Используемый объект**:
 - **Авто** – использовать объект, значение атрибута **Использовать как колонтитул** которого соответствует конфигурируемому колонтитулу;
 - **Вручную** – при выборе этого значения доступен атрибут **Объект**. При нажатии ЛК в поле **Значение** этого атрибута открывается дерево проекта, в котором нужно выбрать объект.

В верхней части окна параметров экрана отображается имя текущего слоя (выбранного в окне **Слой**).

Часть параметров для создаваемых графических экранов можно задать в редакторе группы шаблонов экранов (см. **Редактор группы шаблонов экранов**).

Окно параметров ГО содержит также следующие инструменты:

- **Скрываемый слой** – если TRUE, все слои данного ГО (кроме самого нижнего слоя) – скрываемые (см. **ГЭ 'Объект'**);
- **Сохранить как картинку** – если этот флаг установлен, графический объект сохраняется в исполняемый файл ***.res** (см. **Файлы узла, создаваемые при экспорте**) как картинка (с потерей динамических свойств), в противном случае – как экран;
- **Использовать как колонтитул** – см. выше конфигурирование колонтитулов экрана.

Особенности вызова графического экрана

Для вызова шаблона экрана в узле создается канал CALL с типом вызова (4) **Screen**, настроенный на вызов соответствующего шаблона (см. **Канал класса CALL и Атрибуты канала класса CALL**).

В профайлере с периодом 1 (5) минут вычисляется значение атрибута 87, **СС** такого канала. Величина **СС*10** мс равна промежутку между временем подачи команды отработки канала и временем доступности соответствующего шаблона для передачи ему параметров (т.е. временем фактической реализации шаблона).

Количество аргументов канала вызова экрана ограничено; в случае 4-байтовых аргументов их число не должно превышать 6000.

Атрибут (0, **R**) канала вызова экрана задает следующие действия:

- 1 – сделать экран невидимым;
- 2 – сделать экран видимым;
- 8 – сделать экран видимым с принудительным обновлением.
- 4 – переместить всплывающий экран.

Для всплывающих экранов (см. **Задание параметров графического экрана**): атрибуты (242, **X_pos**) и (243, **Y_pos**) индицируют и задают положение экрана соответственно по осям X и Y. Для изменения положения всплывающего экрана надо задать (изменить) (242, **X_pos**) и (243, **Y_pos**) и установить (0, **R**)=4 в канале вызова экрана.

- 6, 7 – принудительная перерисовка экрана;
- 254 – сохранение экрана в файл PNG.

Имя текущего экрана, время перехода на текущий экран и имя текущего пользователя отображаются в строке статуса MPB, а в полноэкранном режиме – в правой части окна меню (см. **Профайлер с поддержкой**

графических экранов). Если экран всплывающий, в его заголовке отображаются информация, зависящая от атрибута **Параметр**:

- **Параметр** = 0 – имя экрана;
- **Параметр** = 2 – имя экрана и время;
- **Параметр** = 4 – имя экрана и дата;
- **Параметр** = 6 – имя экрана, дата и время;
- **Параметр** = 8 – имя экрана/канала, на который перепривязаны аргументы экрана;
- **Параметр**=12 – последние сообщения в tm6_log.txt и в отладочный файл.

Для ограничения числа открытых всплывающих экранов в файле *.cnf может использоваться ключ **SCR_POPUP**=<число>.

Установленные биты атрибута (245, **A_OPT**) канала вызова экрана отображают следующую информацию

- бит 0 – экран всплывающий;
- бит 1 – в экране есть тренд;
- бит 2 – резерв;
- бит 3 – экран видим.

В консолях: при установке в CALL.Screen флага **Запрос времени значения** автоматически считывается атрибут (7, **P**) привязанных каналов.

Если при старте на экран выводится информация через **SubNum** (см. **Номер SubNum**), в канале вызова этого экрана автоматически устанавливается **ForceUpdate** (см. описание атрибута (52, **FS**) в разделе **Атрибуты каналов, отображаемые профайлером**).

Экономичная загрузка

Режим экономии ОЗУ за счет графики активируется с помощью следующих ключей команды запуска (см. также **Дополнительные ключи команды запуска** в разделе **Профайлер с поддержкой графических экранов**):

- **/scr_swap** – экономичная загрузка обычных экранов, у которых не установлен флаг **Загружать** (см. **Задание параметров графического экрана**);
- **/scr_swap_all** – экономичная загрузка всех обычных экранов (вне зависимости от флага **Загружать**).

Экономичная загрузка всплывающих экранов не поддержи-

вается.

Совместно с ключом активации могут быть заданы следующие ключи:

- **/scr_swap_mem** – если этот ключ задан, при запуске узла «экономичные» экраны помещаются в ОЗУ в сжатом виде; если ключ не задан – в единый временный файл **../<имя файла prj>/screens.tmp**.

При вызове «экономичный» экран загружается в ОЗУ из своего хранилища; при последующем вызове другого экрана MPB освобождает память, занятую «экономичным» экраном (в том числе очищает его буферы).

Специальные операции с графическими экранами

В ИС определены следующие виды перетаскивания шаблона/канала вызова экрана из навигатора проекта в шаблон экрана, открытый в РПД:

- обычное перетаскивание (drag-n-drop) – см. ГЭ 'Кнопка';
- перетаскивание с удержанием **CTRL** – см. ГЭ 'Ссылка на экран';
- перетаскивание с удержанием **CTRL** и **SHIFT**:
 - из перетаскиваемого экрана создается графический объект с именем **<имя шаблона>**, который помещается в группу **objects_from_screens** слоя **Ресурсы**. Если в объекте несколько слоев, для него **Скрываемый слой** = TRUE автоматически;
 - открывается контекстное меню, содержащее опции **PO**, **Все** и **Создать и привязать аргументы**, от которых зависит результат вставки созданного графического объекта в экран. При выборе первых двух опций выполняются те же действия, что и при вставке обычного графического объекта способом 3 (см. ГЭ 'Объект'). При выборе опции **Создать и привязать аргументы** в экран копируется таблица аргументов перетаскиваемого экрана, включая привязки (привязки копируются только в канал вызова экрана);
- перетаскивание с удержанием **SHIFT**:
 - из перетаскиваемого экрана создается графический объект с именем **<имя шаблона>**, который помещается в группу **objects_from_screens** слоя **Ресурсы**. Если в объекте несколько слоев, для него **Скрываемый слой** = TRUE автоматически;
 - созданный объект вставляется в экран, при этом аргументы объекта привязываются к аргументам экрана с анализом флага **PO** (для таких аргументов ищутся совпадающие по имени аргументы экрана, если их нет – создаются);

- оригинал (шаблон/канал вызова) удаляется из дерева проекта.

Кроме того, определены те же 4 вида перетаскивания объекта библиотеки компонентов (см. **Группы слоя 'Библиотеки компонентов'**) на экран в узле. Во всех случаях вначале производится копирование и вставка объекта в узел (см. **Копирование и вставка объекта библиотеки в узел** в разделе **Копирование и вставка объекта структуры**), а затем – одна из операций, описанных выше (перетаскивается канал вызова экрана, созданного из экрана с младшим ID в объекте).

Сохранение экрана в файл

В ИС: для сохранения экрана в файл .../<директория ИС>/<имя шаблона>.bmp нужно нажать ЛК в произвольной точке шаблона и нажать **CTRL+SHIFT+P**. В режиме эмуляции данная функция не работает.

В МРВ: для сохранения видимого экрана в файл .../<папка узла>/<ID шаблона>_<ID канала>.png нужно послать 254 в атрибут 0, **R** соответствующего канала **CALL.Screen**.

Перезагрузка шаблона экрана в реальном времени

Перезагрузка шаблона экрана в реальном времени производится при следующих значениях атрибута (0, **R**) канала вызова экрана:

- 128, 138 – из файла *.res (см. **Файлы узла, создаваемые при экспорте**);
- 129, 139 – из файла **def_<имя канала>.rld**;
- 130, 140 – из файла **new_<имя канала>.rld**.

Файлы *.rld имеют такой же формат, что и файлы *.res.

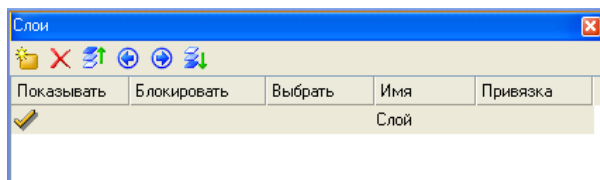
Операции с графическими слоями

Графический экран может содержать один или более графических **слоев**, каждый из которых, в свою очередь, может содержать один или более **подслоев**.

Начиная с релиза 6.06, в экране нельзя создать больше 32 слоев. Проекты, сделанные в предыдущих релизах и содержащие экраны с большим количеством слоев, сохраняются и экспортируются корректно (сохраняются все слои).

Создание и удаление графических слоев

Вновь созданный графический экран содержит один слой с именем по умолчанию:



Окно **Слой** содержит типовые инструменты и создания/удаления слоев. Имя, автоматически присваиваемое слою при его создании, может быть изменено (см. **Типовые средства редактирования**).

Вновь созданный слой по умолчанию располагается поверх слоев, созданных ранее.

При удалении слоя, содержащего графические элементы, на экран выводится диалог-предупреждение, в котором следует подтвердить/отменить удаление.

Управление видимостью графических слоев


Для управления видимостью графических слоев предназначен первый столбец (**Показывать**) списка слоев окна **Слой**. Отображаемые слои помечаются в этом столбце знаком (для нового слоя признак отображения устанавливается по умолчанию). Чтобы установить/удалить этот знак, нужно нажать ЛК в соответствующем поле указанного столбца.

В реальном времени видимостью графических слоев можно управлять с помощью привязки слоя к аргументу экрана. Аргумент выбирается в табличном редакторе аргументов, для открытия которого нужно дважды

нажать ЛК в поле **Привязка**. Имя аргумента после привязки отображается в окне **Слой**.

В реальном времени при равенстве аргумента нулю слой будет виден на экране, при любом другом значении аргумента слой невидим (см. также **Задание параметров графического экрана**).

Блокировка редактирования графического слоя





Для блокировки редактирования графических слоев используется второй столбец (**Блокировать**) списка слоев окна **Слой**. Чтобы запретить редактирование слоя, нужно установить для него знак  в этом столбце (при создании нового слоя его редактирование по умолчанию разрешено).

Чтобы установить/удалить знак блокировки, нужно нажать ЛК в соответствующем поле указанного столбца.

Z-позиционирование графических слоев

В списке окна **Слой** слои экрана отображаются в соответствии с их расположением вдоль оси **Z**, перпендикулярной плоскости экрана (самый верхний слой занимает самую верхнюю строку списка).


Существуют следующие способы изменения расположения слоев по оси **Z**:

- метод **drag-and-drop** (см. **Типовые средства редактирования**);
- инструменты окна **Слой**:
 -  – переместить наверх;
 -  – переместить на один слой вверх;
 -  – переместить на один слой вниз;
 -  – переместить вниз;
- атрибут **Слой** ГЭ (см. **Статические атрибуты ГЭ**).

При изменении расположения слоев поддерживаются операции **Отменить/Вернуть**.

Выделение элементов слоя

Третий столбец (**Выбрать**) списка слоев окна **Слой** используется следующим образом:

- **выделение всех ГЭ слоя** – при двойном нажатии ЛК в поле **Выбрать** устанавливается знак  и все ГЭ соответствующего слоя выделяются. Для снятия выделения нужно дважды нажать ЛК

в поле **Выбрать** или нажать ЛК в произвольном месте экрана, не содержащем ГЭ;

- **индикация выделения элемента (элементов) слоя** – при выделении хотя бы одного ГЭ слоя (см. **Выделение ГЭ**) в соответствующем поле **Выбрать** устанавливается знак ✓.

Операции с графическими элементами

Размещение ГЭ

Встроенные графические элементы разбиты на группы. Каждой группе соответствует кнопка на панели инструментов **Графические элементы** (см. **Главное меню и панели инструментов РПД**).

Чтобы выбрать ГЭ для размещения, нужно выполнить следующие действия:

- нажать ЛК на кнопке панели инструментов **Графические элементы**. При этом выбирается тот элемент, чья иконка выведена на кнопку (элемент, заданный по умолчанию для соответствующей группы, или элемент, выбранный ранее);
- дважды нажать ЛК на кнопке и затем нажать ЛК на иконке требуемого ГЭ в появившемся меню (меню не открывается, если группа содержит только один графический элемент).

При нажатии ЛК на левом крае меню оно становится самостоятельной панелью, которая содержит, в том числе, кнопку закрытия.

После выбора элемента его иконка выводится на кнопку группы.

При выборе графического элемента редактор представления данных переходит в режим размещения (см. **Режимы работы РПД**), при этом курсор на графическом экране приобретает вид **+**.

Далее в окне **Слой** необходимо нажатием ЛК указать слой, в котором должен быть размещен выбранный графический элемент.

При размещении в слое для каждого ГЭ создается отдельный **подслой**, т.е. графический элемент размещается в собственном подслое выбранного слоя графического экрана. Вновь размещаемый ГЭ располагается поверх всех ГЭ, размещенных ранее в данном слое. Управление расположением слоев описано в разделе **Z-позиционирование графических слоев**, управление расположением подслоев – в разделе **Позиционирование ГЭ**.

Подслои оконных ГЭ размещаются поверх всех подслоев вне

зависимости от слоя. К таким ГЭ относятся **Кнопка, Группа кнопок, Картинка-кнопка**, тренды, таблицы, **Текст из файла, ОТ узла, Диаграмма Ганта, Объект в окне, Компонент ActiveX**.

Оконные ГЭ в РПД в Windows 7: для корректного расчета толщины рамок в случае интерфейса Windows 7 Aero следует задать стиль отображения ИС **TRACE MODE 6** или **WindowsXP** (см. **Раздел 'Вид'** в разделе **Вкладка 'Интегрированная среда разработки'**), а в случае Windows 7 Classic – стиль **Windows**. В MPB согласование стилей выполняется автоматически (см. также **Дополнительные ключи команды запуска** в разделе **Профайлер с поддержкой графических экранов**).

После размещения ГЭ его можно переместить в другой слой (см. **Контекстные меню РПД** и **Таблица 'Графические элементы'**).

Далее продолжить процедуру размещения ГЭ можно двумя способами:

- перетащить ГЭ с панели инструментов на экран (метод **drag-and-drop** – см. **Типовые средства редактирования**); после размещения ГЭ имеет размеры, заданные по умолчанию, РПД переходит в режим редактирования, окно свойств ГЭ открывается автоматически. При перетаскивании с панели инструментов ГЭ групп **Ресурсы** и **Объекты** на экране размещается первый ресурс/объект первой библиотеки;
- переместить курсор в нужную точку экрана и нажатием ЛК установить **точку привязки** ГЭ. Далее действия по размещению ГЭ могут отличаться, однако для большинства графических элементов они стандартны – перемещение мыши после установки точки привязки выводит на экран образ ГЭ, при этом отрезок от точки привязки до текущего положения курсора является диагональю прямоугольника, ограничивающего ГЭ. (Если при перемещении мыши удерживать нажатой клавишу **CTRL**, ряд ГЭ окажется вписанным в квадрат). Повторное нажатие ЛК приводит к размещению графического элемента в выбранном графическом слое. (В ряде случаев для размещения ГЭ на экране достаточно установить точку привязки).

Для графических элементов группы **Ломаные и кривые** каждое нажатие ЛК после установки точки привязки задает **узловую точку** (промежуточную вершину). Для установки последней вершины и выхода из режима размещения этих ГЭ нужно нажать ПК. Положение узловых точек, заданное при размещении, можно в дальнейшем изменить (см. **Позиционирование узловых точек ГЭ**). Если при рисовании нажата клавиша **CTRL**, угол изгиба ГЭ

может быть только кратным 45°.

Режим РПД после размещения ГЭ данным способом зависит от флага **Открывать свойства автоматически** (см. **Задание параметров РПД**):


- если флаг установлен, то после размещения ГЭ автоматически открывается окно его свойств, а РПД переходит в режим редактирования;
- если флаг не установлен, то после размещения ГЭ РПД остается в режиме размещения. Этот способ удобен для многократного размещения на экране одного и того же графического элемента.

Для отмены размещения в его процессе надо нажать **ESC**.

Особенности размещения указаны в описании ГЭ.

Выделение ГЭ

Для выделения ГЭ (группы ГЭ) используются стандартные операции редактирования (см. **Типовые средства редактирования**), а также таблица графических элементов (см. **Таблица ‘Графические элементы’**) и окно **Слои** (см. **Выделение элементов слоя**).

Выделить можно, в том числе, группу графических элементов, лежащих в разных слоях экрана. При наведении на выделенный ГЭ или выделенную группу ГЭ курсор принимает вид .

Чтобы снять выделение, нужно нажать ЛК в области экрана, не содержащей ГЭ, или **ESC**, а также использовать таблицу ГЭ или окно **Слои**.

Перемещение и масштабирование ГЭ

Для перемещения или изменения размеров выделенного ГЭ (группы ГЭ) нужно выбрать в контекстном меню графического элемента режим **Перемещать/масштабировать** и далее использовать стандартные операции редактирования (см. **Типовые средства редактирования**).

Для дискретного перемещения выделенного ГЭ (группы ГЭ) можно использовать комбинации клавиш **SHIFT+→/←/↓/↑**, для дискретного изменения размеров – **CTRL+→/←/↓/↑**. Если флаг **Располагать по сетке** не установлен (см. **Задание параметров РПД**), дискретное перемещение/масштабирование производится с шагом 1px, если флаг установлен – с шагом сетки.

Перемещение и масштабирование ГЭ может быть также выполнено с помощью таблицы графических элементов (см. **Таблица ‘Графические элементы’**) или в окне свойств в разделе **Геометрия** (см. **Статиче-**

ские атрибуты ГЭ).

При изменении размеров отдельного ГЭ с помощью мыши или с помощью клавиш **CTRL+→/←/↓/↑** начальные размеры ГЭ не запоминаются (в диалоге **Геометрия** таблицы графических элементов изменяются размеры). При изменении теми же способами размеров группы ГЭ начальные размеры каждого элемента группы запоминаются (в диалоге **Геометрия** изменяется масштаб).

Удаление ГЭ

Для удаления выделенного ГЭ (группы ГЭ) нужно нажать клавишу **Del** на клавиатуре или выполнить команду **Удалить** с помощью меню или панели инструментов **Правка** (см. **Главное меню и панели инструментов РПД**), а также с помощью контекстного меню ГЭ или контекстного меню таблицы графических элементов (см. **Таблица 'Графические элементы'**).

В полноэкранный режиме редактирования для удаления выделенного ГЭ (группы ГЭ) с помощью клавиатуры используется комбинация клавиш **Ctrl+Del**.

Копирование и вставка ГЭ

Для копирования/перемещения выделенного ГЭ (группы ГЭ) в буфер обмена нужно выполнить команду **Копировать/Вырезать**.

Для вставки содержимого буфера обмена в слой нужно выполнить команду **Вставить**.


Скопировать в буфер обмена можно, в том числе, группу графических элементов, лежащих в разных слоях экрана, однако при вставке такой группы все ее ГЭ будут размещены в одном и том же слое.

Выполнить команды работы с буфером обмена можно с помощью меню или панели инструментов **Правка** (см. **Главное меню и панели инструментов РПД**), а также с помощью контекстного меню ГЭ или таблицы графических элементов (см. **Таблица 'Графические элементы'**).

Поворот ГЭ

Существует 3 способа поворота ГЭ на экране в режиме редактирования.

Режим ‘Повернуть’

Для перехода в этот режим нужно выбрать в контекстном меню выделенного ГЭ (группы ГЭ) опцию **Повернуть**. Далее надо установить курсор в одну из вершин прямоугольника, ограничивающего ГЭ (группу ГЭ) (курсор при этом принимает вид ).

Затем нужно нажать ЛК и, удерживая кнопку нажатой, перемещением мыши задать нужный угол поворота ГЭ. Для выхода из режима надо отпустить ЛК.

При использовании данного метода ГЭ (группа ГЭ) вращается относительно центра ограничивающего прямоугольника (центром прямоугольника является точка пересечения его диагоналей).

Поворот из таблицы графических элементов

Угол поворота ГЭ можно задать в поле **Угол (градусы)** диалога **Геометрия** (см. **Таблица ‘Графические элементы’**). При использовании этого метода ГЭ поворачивается относительно точки привязки (при задании положительного значения – по часовой стрелке).

Поворот группы ГЭ из таблицы графических элементов выполнить нельзя.


Поворот из окна свойств


Угол поворота ГЭ можно задать в разделе **Геометрия** окна свойств (см. **Статические атрибуты ГЭ**).


Позиционирование ГЭ


Взаимное позиционирование ГЭ

Для взаимного позиционирования выделенных ГЭ нужно выполнить одну из следующих команд меню **Сервис** или панели инструментов **Топология экрана** (см. **Главное меню и панели инструментов РПД**):


 **Выровнять влево** – выровнять влево по ГЭ, левый край которого имеет наименьшую абсциссу;

 **Выровнять вправо** – выровнять вправо по ГЭ, правый край которого имеет наибольшую абсциссу;


 **Выровнять вверх** – выровнять вверх по ГЭ, верхний край которого имеет наименьшую ординату;


 **Выровнять вниз** – выровнять вниз по ГЭ, нижний край которого имеет наибольшую ординату;


Точка с координатами (0, 0) размещается в верхнем левом углу экрана, ось абсцисс направлена вправо, ось ординат – вниз.


 **Центрировать в группе горизонтально** – по этой команде графические элементы перемещаются по горизонтали так, чтобы абсциссы их центров стали равны абсциссе центра изначально выделенной группы;


Под центром ГЭ (группы ГЭ) понимается центр ограничивающего прямоугольника.


 **Центрировать в группе вертикально** – по этой команде графические элементы перемещаются по вертикали так, чтобы ординаты их центров стали равны ординате центра изначально выделенной группы;

 **Упорядочить горизонтально** – данная команда доступна при выделении группы из $N > 2$ графических элементов. При выполнении команды графические элементы, занимающие крайние положения по оси Z , не перемещаются, а все остальные ГЭ размещаются в соответствии с положением по оси Z с равными промежутками между левыми границами;

 **Упорядочить вертикально** – упорядочивание ГЭ по вертикали. Алгоритм выполнения этой команды аналогичен алгоритму выполнения команды **Упорядочить горизонтально**;

 **Выровнять по ширине** – по этой команде ширины всех ГЭ становятся равными ширине самого верхнего по оси Z элемента;

 **Выровнять по высоте** – по этой команде высоты всех ГЭ становятся равными высоте самого верхнего по оси Z элемента;

 **Выровнять по размеру** – по этой команде ширины и высоты всех ГЭ становятся равными соответственно ширине и высоте самого верхнего по оси Z элемента.


Центрирование ГЭ на экране

Для центрирования на экране выделенного ГЭ (группы ГЭ) нужно выполнить одну из следующих команд меню **Сервис** или панели инструментов **Топология экрана** (см. **Главное меню и панели инструментов РПД**):

 **Центрировать на экране горизонтально**


 **Центрировать на экране вертикально**


Если выделена группа ГЭ, то при выполнении этих команд элементы группы центрируются взаимно, после чего центрируется группа.

По команде  **Центрировать на экране во весь экран** выделенный ГЭ занимает весь экран.

Управление расположением подслоев

Для изменения расположения подслоев используются следующие команды меню **Сервис** или панели инструментов **Топология экрана** (см. **Главное меню и панели инструментов РПД**):

 **Переместить вниз** – переместить подслою, содержащий выделенный ГЭ, на самый нижний уровень. По этой команде все остальные подслои перемещаются на один уровень вверх;


 **Переместить вверх** – переместить подслою, содержащий выделенный ГЭ, на самый верхний уровень. По этой команде все остальные подслои перемещаются на один уровень вниз.

Таким образом, в отличие от графических слоев (см. **Z-позиционирование графических слоев экрана**), пошаговое перемещение подслоев не предусмотрено.

Если данные команды выполняются при выделении в слое группы ГЭ, то подслои, содержащие выделенные ГЭ, перемещаются на самые верхние/нижние уровни слоя с сохранением взаимного расположения по оси Z.

Позиционирование узловых точек ГЭ

Для перехода в режим позиционирования узловых точек выделенного ГЭ группы **Ломаные и Кривые** или ГЭ **‘Труба’** нужно выполнить команду **Редактировать узловые точки** из контекстного меню ГЭ. По этой команде узловые точки выделяются.


Для перемещения узловой точки нужно навести на нее курсор (курсор при этом принимает вид ) и использовать метод **drag-and-drop** (см. **Типовые средства редактирования**).

Для добавления узловой точки нужно нажать ЛК на имеющейся точке при удержании клавиши CTRL.

Для удаления узловой точки нужно нажать на ней ЛК при удержании клавиши SHIFT.

Тиражирование ГЭ

Тиражирование – это копирование выделенного графического элемента и его множественная вставка с табличным упорядочением. Копии вставляются вправо и вверх/вниз относительно выделенного ГЭ. Тиражирование выделенной группы ГЭ не поддерживается.

Операция тиражирования конфигурируется в диалоге, который открывается при выполнении команды **Тиражировать** из меню **Сервис** или нажатии кнопки  на панели инструментов **Топология экрана** (см. **Главное меню и панели инструментов РПД**).

В этом диалоге задаются следующие параметры:

- **Число столбцов** – число элементов в строке (включая первоначально выделенный ГЭ);
- **Число строк** – число строк;
- **Промежуток по горизонтали** – промежуток между копиями по горизонтали в пикселях;
- **Промежуток по вертикали** – промежуток между копиями по вертикали в пикселях;
- **Вверх** – размножение по строкам вверх относительно выделенного ГЭ;
- **Вниз** – размножение по строкам вниз относительно выделенного ГЭ.

Операции с аргументами в РПД

В РПД автоматически выполняются следующие операции при перетаскивании (drag-n-drop) выделенного аргумента (выделенных аргументов) из редактора аргументов на ГЭ, размещенный на экране:

- на ГЭ 'Текст' – если атрибут **Текст** не динамизирован (см. **Динамизация атрибута ГЭ**) – динамизация атрибута (вид индикации – **Значение**, формат – **%g**). Если атрибут динамизирован – замена привязки (вид индикации и формат не изменяются);
- на плоские фигуры с динамизируемой заливкой (кроме ГЭ 'Многоугольник' и ГЭ 'Замкнутая кривая') – если цвет заливки не динамизирован – динамизация **Arg >= Конст.** (**Константа = 0**). Если цвет заливки динамизирован – замена привязки (вид индикации и значение константы не изменяются);
- на объемные фигуры – если атрибут **Базовый цвет** не динамизирован – динамизация **Arg >= Конст.** (**Константа = 0**). Если атрибут динамизирован – замена привязки (вид индикации и значение константы не изменяются);
- на ГЭ 'Кнопка' и ГЭ 'Картинка-кнопка' – удаление всех функций управления (см. **Функции управления ГЭ**) и задание функции управления **Передать значение** (событие – **mousePressed**, тип передачи – **Ввести и передать**);
- на тренды – добавление кривых в соответствии с возрастанием порядковых номеров аргументов в выделенной группе. Если выделенная группа содержит 2k аргументов, для ГЭ 'Тренд ХУ' задается k кривых (аргумент с младшим порядковым номером в паре привязывается к атрибуту кривой **Привязка X**).

При перетаскивании выделенного аргумента в таблицу слоев производится привязка к этому аргументу выбранного слоя (см. **Управление видимостью графических слоев**).

Операции, выполняемые при перетаскивании аргументов в eРПД, описаны в разделе **Операции с аргументами в eРПД**.

Задание типовых свойств ГЭ

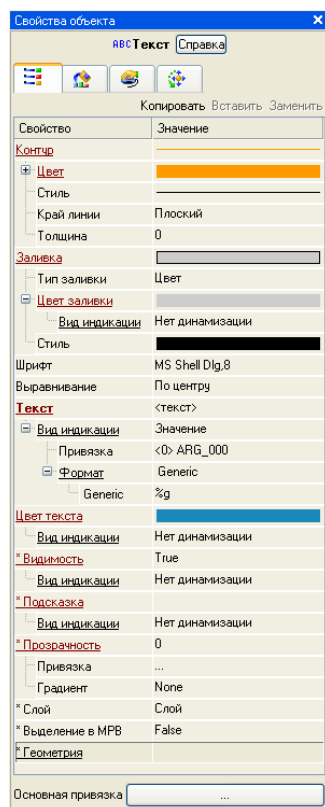
В этом разделе описаны типовые свойства графических элементов и инструменты их задания. Специфические свойства отдельных ГЭ рассматриваются при описании этих ГЭ.

Графические элементы имеют следующие настраиваемые свойства:

- **Атрибуты**
- **Динамические свойства**
- **Функции управления**

Эти параметры определяют вид графических элементов и выполняемые ими функции отображения и управления при работе в реальном времени:

Для задания свойств ГЭ (группы ГЭ) используется окно **Свойства объекта**, содержащее различное число вкладок для разных элементов:



Чтобы открыть это окно, нужно выполнить команду **Свой-**

ства из контекстного меню выделенного ГЭ (выделенной группы ГЭ) или дважды нажать ЛК на ГЭ. При снятии выделения ГЭ окно его свойств автоматически заменяется окном свойств экрана. Окно **Свойства объекта** недоступно, если графический элемент расположен в слое, редактирование которого запрещено (см. **Блокировка редактирования графического слоя**).

Атрибуты – это простейшие свойства графического элемента. Они задаются на вкладке  (**Основные свойства**) окна **Свойства объекта**.

В окне свойств атрибуты могут быть сгруппированы – наименования таких групп выделены подчеркиванием, при двойном нажатии на них ЛК раскрывается список свойств.

Существуют 2 вида атрибутов ГЭ:

- **статический** – атрибут, который не изменяется при работе в реальном времени;
- **динамизируемый** – два одноименных атрибута (статический и динамический). Разделы конфигурирования динамизируемых атрибутов выделены в окне свойств красным цветом и содержат атрибуты **Вид индикации** и **Привязка** (см. **Динамизация атрибута ГЭ**). Динамизируемый атрибут может также рассматриваться как один атрибут с двумя значениями (статическим и динамическим).

Статические атрибуты ГЭ

В данном разделе описано задание типовых статических атрибутов ГЭ с помощью вкладок окна **Свойства объекта**.

Некоторые типовые статические атрибуты ГЭ могут быть также заданы с помощью панелей инструментов РПД (см. **Главное меню и панели инструментов РПД**).

Видимость

Если не сконфигурировано никакое другое управление видимостью ГЭ (см. **Динамизация атрибута ГЭ**), при переходе в режим реального времени (т.е. при запуске проекта в МРВ или при переходе в режим эмуляции) ГЭ видим, если **Видимость**=TRUE, и невидим, если **Видимость**=FALSE.

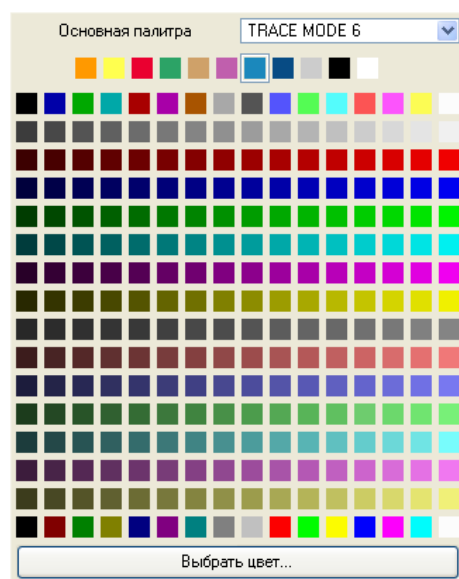
В режиме редактирования видимы все ГЭ вне зависимости от атрибута **Видимость**.

Всплывающая подсказка

В разделе **Подсказка** для ГЭ задается всплывающая подсказка, отображаемая на экране при наведении курсора на ГЭ в режиме реального времени. Чтобы отменить изменение текста, нужно нажать **ESC**.

Цветовые атрибуты

Разделы конфигурирования цветовых атрибутов ГЭ (**Цвет**, **Цвет линии**, **Цвет заливки**, **Цвет текста** и т.п.) в окне **Свойства объекта** содержат кнопку, при нажатии которой на экран выводится диалог выбора предустановленных цветов:



Раскрывающийся список **Основная палитра** позволяет выбрать различные наборы предустановленных базовых цветов, а также набор из 16 цветов, определяемых пользователем (для переопределения цвета нужно нажать ПК на его пиктограмме и выбрать цвет в стандартном диалоге). Изменение в пользовательском наборе сохраняется в `gtext.xml` при закрытии (любым способом) диалога ИС выбора цвета.

При нажатии кнопки **Выбрать цвет** на экран выводится стандартный диалог выбора цвета.

Выбранный цвет выводится на кнопку соответствующего раздела в окне свойств.

Любой цветовой атрибут ГЭ и экрана хранит как RGB, так и индекс в пользовательском наборе, если этот набор был ис-

пользован. RGB проверяется по индексу после редактирования цвета и при загрузке экрана.

Системный цвет

Подобный атрибут может быть корневым в окне свойств ГЭ или располагаться в разделе конфигурирования некоторого цветового атрибута.

Если значение атрибута – TRUE, то в первом случае он задает использование соответствующих системных цветов Windows для всех цветовых атрибутов ГЭ, во втором – только для атрибута, в разделе которого находится.

Если значение атрибута – FALSE, используются цвета, заданные в окне свойств ГЭ.

Толщина

В разделе конфигурирования атрибута **Толщина** толщина задается в пикселях.

Стиль линии

Раздел конфигурирования атрибута **Стиль линии** содержит кнопку, при нажатии которой на экране появляется меню выбора стиля линии.

Выбранный вид линии выводится на кнопку раздела.

Стиль заливки

Раздел конфигурирования атрибута **Стиль заливки** содержит кнопку, при нажатии которой на экране появляется меню выбора стиля заливки.

Выбранный стиль выводится на кнопку раздела.

Прозрачность заливки

При значении TRUE (значение по умолчанию) этот атрибут задает прозрачность фона ГЭ при одном из линейных стилей заливки. Если атрибут имеет значение FALSE, цвет фона ГЭ – белый.

При точечных стилях заливки этот атрибут не работает (фон ГЭ – белый).

Тип заливки

Раздел конфигурирования заливки ГЭ содержит атрибут **Тип заливки**, для которого может быть установлено значение **Цвет** (значение по умол-

чанию) или **Изображение**.

При значении **Цвет** параметры заливки определяются ее цветовыми атрибутами.

При значении **Изображение** в окне свойств отображается атрибут **Изображение**. При нажатии кнопки в разделе его конфигурирования на экране появляется диалог выбора картинки из ресурсной библиотеки изображений.

Выбранная картинка отображается в разделе конфигурирования атрибута **Изображение** и устанавливается в качестве фона ГЭ.

Текстовые атрибуты

Раздел конфигурирования текстовых атрибутов (**Текст**, **Надпись** и т.п.) содержит окно обычного текстового редактора для задания значения атрибута (для некоторых ГЭ подобному атрибуту по умолчанию присваивается некоторое значение). Чтобы отменить изменение текста, нужно нажать **ESC**.

Многострочный текст может быть задан для атрибута **Текст** ГЭ **Текст** и ГЭ **Кнопка** (для вставки символа перевода каретки надо нажать **CTRL+ENTER**).

Шрифт

Раздел конфигурирования атрибута **Шрифт** содержит кнопку, при нажатии которой на экран выводится меню выбора параметров шрифта.

По команде **Размеры** этого меню на экране появляется меню выбора размера шрифта, по команде **Шрифт** – стандартный диалог задания параметров шрифта.

Выбранные параметры шрифта отображаются на кнопке раздела.

По умолчанию задан шрифт **MS Shell Dlg** с размером 8.

Выравнивание

Раздел конфигурирования атрибута **Выравнивание** содержит стандартные инструменты выравнивания текста: **Влево**, **По центру** и **Вправо**.

Прозрачность

Атрибут **Прозрачность** определяет степень прозрачности ГЭ. Этот параметр задается в процентах (0-100), 0 соответствует абсолютной непрозрачности. С помощью дополнительного атрибута **Градиент** можно задать несколько видов градиента прозрачности ГЭ. Особенности прозрач-

ности объектов описаны в разделе **ГЭ 'Объект'**.

Слой

Атрибут **Слой** задает слой экрана (из числа созданных слоев), в который необходимо переместить ГЭ.

Выделение в МРВ

Атрибут **Выделение в МРВ** разрешает/запрещает выделение графического элемента в реальном времени по нажатию ЛК.

Выделение работает только в том случае, если для ГЭ задана функция управления (см. **Функции управления ГЭ**).

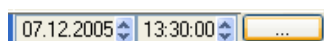
Если ГЭ входит в состав графического объекта (для объекта должно быть разрешено выделение), то по нажатию ЛК на ГЭ выделяется объект.


Для оконных ГЭ (см. **Размещение ГЭ**) эта функция отключена.

Глобальное разрешение выделения графических элементов в реальном времени и параметры выделения (цвет, отступы, толщина и стиль контура) задаются в файле **gr_settings.xml** (тег **<last_clicked_attrs>**), расположенном в директории МРВ. Параметры выделения могут быть также заданы при конфигурировании РПД (см. **Задание параметров РПД**).

Временные атрибуты

Дата и время должны задаваться в форматах, заданных в региональных настройках ОС. Для перехода к заданию времени нужно нажать ЛК в соответствующем поле, которое при этом принимает следующий вид:



При нажатии кнопки  открывается меню, содержащее команды **Сброс** (задать нулевое время, т.е. 01.01.1970) и **Текущее** (задать текущее время).

Масштабируемый

Если значение атрибута **Масштабируемый** – TRUE, размеры оконного ГЭ (в т.ч. окна отображения скрываемого слоя) можно изменять в реальном времени. Кроме того, этот атрибут влияет на толщину рамки окна:

FALSE – тонкая, TRUE – толстая.

Ориентация

Этот атрибут задает ориентацию ГЭ (как правило, направление оси симметрии).

Заголовок

Для оконных ГЭ: если значение атрибута **Заголовок** не является пустой строкой, окно имеет заголовок, в котором отображается заданная строка. Для отображения текста в заголовке окна используется шрифт ОС.

Геометрия

В этом разделе окна свойств задаются следующие атрибуты: координаты точки привязки ГЭ (**X, Y**), размеры ГЭ (**Ширина, Высота**) и, для некоторых ГЭ, угол поворота. Эти атрибуты доступны при **Геометрия = Редактировать**.

Динамизация атрибута ГЭ

Динамизацией атрибута называется задание условий изменения его динамического значения в зависимости от значения привязанного аргумента. При динамизации атрибута графический элемент становится индикатором выполнения заданных условий.

При размещении ГЭ на экране все его динамизируемые атрибуты (см. **Задание типовых свойств ГЭ**) по умолчанию статические, и разделы их конфигурирования на вкладке **Основные свойства** окна свойств содержат инструмент задания соответствующего статического параметра (см. **Статические атрибуты ГЭ**).

Для динамизации атрибута нужно выполнить следующие действия (см. также **Операции с аргументами в РПД** и **Операции с аргументами в еРПД**):

- раскрыть в окне свойств ГЭ раздел конфигурирования атрибута как динамического параметра (дважды нажать ЛК на имени атрибута);
- в списке **Вид индикации** выбрать вид условия (и, соответственно, вид индикатора, создаваемого из ГЭ);

Набор доступных видов индикации зависит от атрибута.

- привязать конфигулируемое динамическое свойство к аргументу экрана:
 - нажать ЛК в поле **Значение** атрибута **Привязка** – по этой команде на экране появляется диалог, содержащий редактор аргументов шаблона экрана (см. **Табличный редактор аргументов**), флаг **Использовать привязанный атрибут**, а также список атрибутов каналов (**attribute.txt**);
 - выполнить в диалоге одно из следующих действий:
 - выбрать аргумент и установить флаг **Использовать привязанный атрибут**, если конфигулируемое динамическое свойство должно быть привязано к тому атрибуту канала, который привязан (или будет привязан) к выбранному аргументу. После привязки в поле **Значение** атрибута **Привязка** отображается порядковый номер и имя привязанного аргумента (например, «<1> ARG_000»);
 - выбрать аргумент, сбросить флаг **Использовать привязанный атрибут** и выбрать атрибут (0,R по умолчанию) или задать номер атрибута в поле **Атрибут**, если конфигулируемое динамическое свойство должно быть привязано к атрибуту канала, отличному от того, который привязан (будет привязан) к выбранному аргументу. После привязки в поле **Значение** атрибута **Привязка** отображается порядковый номер и имя аргумента, а также короткое имя атрибута (например, «<1> ARG_000 (In)»). Если короткое имя атрибута не существует, отображается номер атрибута («<1> ARG_000 (452)»). Исключение: подобная привязка атрибута типа БИТ, БАЙТ, ПОЛУ-БАЙТ не обрабатывается (не считывается и не записывается).

Если выбранный аргумент привязан к некоторому атрибуту А, а динамическое свойство привязано к другому атрибуту В, то индикация изменения В отрабатывается только при изменении А.

Вне зависимости от флага **Использовать привязанный атрибут**, конфигулируемое динамическое свойство будет зависеть от значения привязанного аргумента, если аргумент не имеет привязки.

Если к привязанному аргументу привязан некоторый аргумент **argL** (но не атрибут-аналог аргумента (140-186)), то конфигулируемое динамическое свойство будет зависеть от значения **argL** вне зависимости от флага **Использовать привязанный атрибут**.

Окно свойств некоторых ГЭ содержит раздел **Основная привязка** для привязки к аргументу. Если основная привязка задана, динамизируемые атрибуты, как правило, автома-

тически привязываются к ней.

- задать остальные атрибуты в разделе конфигурирования динамического свойства (набор атрибутов зависит от вида индикации).

Ниже приведен полный список видов индикации с указанием доступности для динамизируемых атрибутов:

- **Значение** – индикация значения аргумента (для текстовых атрибутов и атрибута **Подсказка**);
- **Arg = Конст.** – индикация равенства аргумента заданной константе (для цветowych и текстовых атрибутов, а также для атрибутов **Видимость** и **Подсказка**);
- **Arg >= Конст.** – индикация превышения аргументом заданного порога (для цветowych и текстовых атрибутов, а также для атрибутов **Видимость** и **Подсказка**);
- **Arg & Конст.** – индикация состояния битов значения аргумента, заданных маской **Константа**. Если хотя бы один такой бит установлен, индицируется **ИСТИНА**, иначе – **ЛОЖЬ** (для цветowych и текстовых атрибутов, а также для атрибутов **Видимость** и **Подсказка**);
- **Arg в диапазоне** – индикация нахождения значения аргумента в заданных диапазонах (для цветowych и текстовых атрибутов, а также для атрибута **Подсказка**);
- **Arg в интервале** – индикация нахождения значения канала **FLOAT** или **DOUBLE FLOAT**, привязанного к аргументу, в интервалах канала (для цветowych атрибутов, см. также **Границы и интервалы канала FLOAT** и **Канал класса DOUBLE FLOAT**);
- **Атр.46 в диапазоне** – индикация нахождения атрибута 46 канала, привязанного к аргументу, в заданных диапазонах (для цветowych и текстовых атрибутов, а также для атрибута **Подсказка**). Данный вид индикации работает при привязке аргумента к любому атрибуту канала (чтение атрибута 46 производится при изменении привязанного атрибута);
- **Набор {Arg = Конст}** – индикация набора фиксированных значений аргумента (для цветowych и текстовых атрибутов, а также для атрибута **Подсказка**);
- **Набор {Arg & Конст = Конст}** – индикация набора выполнения условий **Arg & Конст = Конст**. В отличие от вида индикации **Arg & Конст.**, условие **Arg & Конст = Конст** означает проверку установки в значении аргумента всех битов, заданных маской (для цветowych и текстовых атрибутов, а также для атрибута **Подсказка**);
- **Набор {Arg & Конст1 = Конст2}** – индикация набора выполнения условий **Arg & Конст1 = Конст2** (для цветowych и текстовых атрибутов, а также для атрибута **Подсказка**);

- **Имя** – отображение имени канала, привязанного к аргументу, во всплывающей подсказке (только для атрибута **Подсказка**). Данный вид индикации работает при привязке аргумента к любому атрибуту канала;
- **Кодировка** – отображение кодировки канала, привязанного к аргументу, во всплывающей подсказке (только для атрибута **Подсказка**). Данный вид индикации работает при привязке аргумента к любому атрибуту канала.

Конфигурирование индикации значения

Данный вид индикации имеет специфический атрибут **Формат**, который определяет формат отображения значения (**Generic**, **Integer**, **Float**, **Exponential**, **По умолчанию**; **Binary**, выбирается в списке).

Для каждого формата можно задать более точное описание в нотации языка Си (см. **Формат Си вывода чисел**):

| | |
|--------|-------|
| Формат | Float |
| Float | %.3f |

При выборе формата **По умолчанию** ГЭ будет отображать значение канала, привязанного к аргументу, в формате, определяемом МРВ. Например, если к аргументу с типом данных **REAL** привязан атрибут **R** канала, связанного с переменной **@Status** с атрибутом **Параметр=0**, то:

- если **Формат** \diamond **По умолчанию**, значение переменной отображается как число;
- если **Формат** = **По умолчанию**, значение переменной отображается как строка ("WORK", "TRACE" и т.п.).

Конфигурирование индикации интервала

Раздел конфигурирования данного вида индикации имеет следующий вид:

| | |
|----------------|-----------------|
| Цвет заливки | |
| Вид индикации | Arg в интервале |
| Привязка | saw_R |
| Предупреждение | |
| Авария | |
| Вне границ | |

Ниже приведено соответствие цвета интервалу:

- интервал 0 (**LW**<arg<**HW**) – цвет определяется статическим значением атрибута;
- интервалы 1 и 2 (**LA**<arg<**LW** или **HW**<arg<**HA**) – цвет определяется атрибутом **Предупреждение**;
- интервалы 3 и 4 (**LL**<arg<**LA** или **HA**<arg<**HL**) – цвет определя-

ется атрибутом **Авария**;

- интервалы 5 и 6 (**arg**<LL или **HL**<arg) – цвет определяется атрибутом **Вне границ**.

Конфигурирование других видов индикации

Наборы инструментов конфигурирования других видов индикации зависят от динамизируемого атрибута. Если параметр индикации при динамизации атрибута может быть задан как вручную, так и выбран из ресурсной библиотеки (см. **Операции с ресурсными библиотеками**), набор инструментов содержит переключатель **Использовать ресурсы** (FALSE – вручную, TRUE – из библиотеки).

На рисунках ниже показаны инструменты конфигурирования различных видов индикации при динамизации атрибута **Текст**.

| Текст (ARG_000) | | Текст (ARG_000) | |
|---------------------------|-----------------|---------------------------|------------------|
| <текст> | | <текст> | |
| Вид индикации | Arg = Констант. | Вид индикации | Arg >= Констант. |
| Привязка | ARG_000 | Привязка | ARG_000 |
| Использовать ресурсы | False | Использовать ресурсы | False |
| Если ИСТИННО | Правильно | Если ИСТИННО | Правильно |
| Доп. значение для ИСТИННО | =55 | Доп. значение для ИСТИННО | >=55 |
| Если ЛОЖНО | Неправильно | Если ЛОЖНО | Неправильно |
| Доп. значение для ЛОЖНО | < 55 | Доп. значение для ЛОЖНО | <55 |
| Константа | 55 | Константа | 55 |

| Текст (sine_R) | | Текст (sine_R) | |
|---------------------------|---------------------|----------------------|-----------------|
| <text> | | <text> | |
| Вид индикации | Arg & Констант | Вид индикации | Arg в диапазоне |
| Привязка | sine_R | Привязка | sine_R |
| Использовать ресурсы | False | Использовать ресурсы | False |
| Если ИСТИННО | Правильно | Диапазоны | |
| Доп. значение для ИСТИННО | бит 2 установлен | Диапазон | 0 1 Диап.0 |
| Если ЛОЖНО | Неправильно | Мин. | 0 |
| Доп. значение для ЛОЖНО | бит 2 не установлен | Макс. | 1 |
| Константа | 4 | Значение | Диап.0 |
| | | Доп. значение | Правильно |
| | | Диапазон | 1 2 Диап.1 |

| Текст (saw_P) | | Текст (ARG_000) | |
|------------------------|------------------------|----------------------|-----------------------------|
| Набор (Arg = Констант) | | FALSE | |
| Вид индикации | Набор (Arg = Констант) | Вид индикации | Набор (Arg & Const = Const) |
| Привязка | saw_P | Привязка | ARG_000 |
| Использовать ресурсы | False | Использовать ресурсы | False |
| Пары | | Пары | |
| Пара | 0 range 0 | Пара | 0x7 TRUE |
| Arg = | 0 | Arg = | 0x7 |
| Значение | range 0 | Значение | TRUE |
| Доп. значение | main | Доп. значение | OK |
| Пара | 1 range 1 | Пара | 0x3 TRUE |
| Пара | 2 range 2 | | |

При конфигурировании видов индикации **Arg = Констант.**, **Arg >= Констант.** и **Arg & Констант.** в полях **Если ИСТИННО** и **Если ЛОЖНО** задаются значения, которые должен принимать динамизируемый атрибут при выполнении заданного условия (**ИСТИННО**) и в противном случае (**ЛОЖНО**). Если задано **Доп. значение для ИСТИННО (ЛОЖНО)**, то значение ат-

рибута, заданное в поле **Если ИСТИННО (ЛОЖНО)**, периодически (1 раз в 0.5 с) сменяется значением, заданным в соответствующем поле **Доп. значение...** (т.е. происходит мигание). Период мигания для цветового атрибута может быть задан с помощью атрибута **Мигание (быстрое – 0.5с, среднее – 1с, медленное – 2с)**. Чтобы отобразить дополнительное значение, надо выполнить соответствующую команду из контекстного меню поля **Если ИСТИННО (ЛОЖНО)**. Значение константы, с которой сравнивается аргумент, задается в поле **Константа**.

При конфигурировании индикации диапазонов разделы описания диапазонов создаются/удаляются с помощью контекстного меню атрибута **Диапазоны** (для создания нового диапазона) или созданного атрибута **Диапазон** (для его удаления). В полях **Мин.** и **Макс.** задаются границы диапазонов (**Макс** должно быть больше **Мин**). Индикатор этого вида работает по следующему алгоритму: ищется первый по списку диапазон, которому удовлетворяет значение аргумента (**Мин <= arg < Макс**) и атрибуту присваивается значение, заданное в соответствующем поле **Значение**. Если значение аргумента находится вне всех диапазонов, атрибуту присваивается его статическое значение. Создание и назначение поля **Доп. значение** – такое же, как при конфигурировании видов индикации **Arg = Констант.**, **Arg >= Констант.** и **Arg & Констант.** Контекстное меню атрибута **Диапазоны** содержит также команду **Соединить диапазоны** – по этой команде **Мин.** каждого диапазона принимает значение **Макс.** предыдущего диапазона.

При конфигурировании индикации **Набор {Arg=Конст.}**, **Набор {Arg & Констант = Констант}** и **Набор {Arg & Констант1 = Констант2}** разделы **Пара** задания констант и значения атрибута создаются/удаляются аналогично созданию/удалению разделов описания диапазонов. Если ни одно условие не выполняется, атрибуту присваивается его статическое значение. Создание и назначение поля **Доп. значение** – такое же, как при конфигурировании видов индикации **Arg = Констант.**, **Arg >= Констант.** и **Arg & Констант.**

Если в поле **Использовать ресурсы** установлено значение **True**, поля **Если ИСТИННО (ЛОЖНО)** и **Доп. значение...** содержат кнопки, при нажатии которых открываются навигаторы соответствующих библиотек для выбора ресурса.

Для типовых атрибутов поля **Если ИСТИННО (ЛОЖНО)** и **Доп. значение...** содержат кнопки, при нажатии которых открываются стандартные диалоги задания параметра (например, цвета).

Управление видимостью ГЭ

Видимость ГЭ динамизируется следующим образом:

- **Видимость** = TRUE;

- Вид индикации = Arg=Конст., Arg>=Конст. или Arg&Конст.;
- Константа = <число>.

При выполнении заданного условия ГЭ видим, в противном случае – невидим.

В реальном времени управлять видимостью ГЭ можно с помощью функции управления ГЭ (см. **Функция управления видимостью ГЭ**) и с помощью ГЭ **Свободные формы** (см. **Группа ГЭ ‘Свободные формы’**). В общем случае, ГЭ видим только тогда, когда это разрешают одновременно все функции управления видимостью, сконфигурированные для этого ГЭ.

Буфер обмена окна ‘Свойства объекта’

Окно **Свойства объекта** снабжено собственным буфером обмена, который позволяет копировать выделенный атрибут и заменять им любой аналогичный, а также вставлять как новый там, где допускается добавление аналогичных атрибутов.

Замена разрешается, если

- целевой атрибут полностью аналогичен;
- частный случай – целевой атрибут имеет тот же тип значения и возможность динамизации стандартным образом.

Содержимое буфера статическое и сохраняется в течение сеанса редактирования проекта.

Сочетания клавиш в окне ‘Свойства объекта’

- **ENTER** – перейти к редактированию выделенного атрибута; в редакторе – применить;
- **ESC** – выйти из режима редактирования атрибута с отменой не сохраненных изменений;
- **→** – раскрыть выделенную группу атрибутов;
- **←** – свернуть выделенную группу атрибутов;
- **↓/↑/Home/End/Pg Up/PgDn** – типовые команды навигации в окне свойств.

В окне выбора цвета для навигации могут использоваться клавиши **→/←/↓/↑**.

Динамические свойства ГЭ

К динамическим свойствам графических элементов относятся **динамическая заливка**, 3 вида **динамической трансформации** (**перемещение, масштабирование и вращение**) и **динамический контур**.

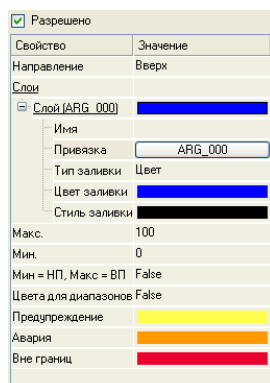
Динамические трансформации автоматически привязываются к аргументу экрана с порядковым номером 0.

Динамические трансформации могут быть заданы, в том числе, для **ГЭ 'Объект'**. Допускается также суперпозиция динамических свойств ГЭ, заданных в объекте, и динамических трансформаций объекта, размещенного на экране.

Динамические свойства настраиваются соответственно на вкладках **Динамическая заливка** (🎨), **Динамическая трансформация** (🔄) и **Динамический контур** (🔗) окна **Свойства объекта**.

Динамическая заливка ГЭ

При использовании данного свойства ГЭ отображает значение привязанного аргумента числового формата в виде закрашенной области (такая область далее называется **слоем**). Поддерживаются два вида динамической заливки – **однослойная** (отображает значение одного аргумента) и **многослойная** (отображает значения нескольких аргументов). Оба вида настраиваются на вкладке **Динамическая заливка** (🎨) окна **Свойства объекта**. Для использования динамической заливки нужно на этой вкладке установить флаг **Разрешено**:



Для добавления/удаления слоя используется контекстное меню, вызываемое нажатием ПК мыши на названиях пунктов **Слой/Слой** соответственно. Настройки для всех создаваемых слоев имеют одинаковое назначение.

Вкладка содержит следующие инструменты конфигурирования заливки.

- **Направление** – направление заливки (**вверх, вниз, вправо, влево**).
- **Имя** – имя слоя. Для перехода к редактированию нужно дважды нажать ЛК в этом поле.
- **Привязка** – выбор аргумента, к которому привязывается слой. При нажатии ЛК в данном поле на экране появляется диалог выбора аргумента (см. **Табличный редактор аргументов**). Слой, для которого привязка к аргументу не задана, считается привязанным к 0. Если заливка многослойная, значения привязываемых аргументов должны быть неотрицательными.
- **Тип заливки** – выбор типа заливки (см. **Статические атрибуты ГЭ**). На вкладке доступны типовые атрибуты конфигурирования заливки выбранного типа.
- **Мин, Макс** – числа, которые ставятся в соответствие границам ГЭ, используемым в качестве пределов шкалы. Например, если выбрано направление заливки справа налево, **Мин** соответствует правой, а **Макс** – левой границе графического элемента. Если флаг **Мин = LL, Макс = HL** не установлен, значения **Мин** и **Макс** могут быть заданы вручную.
- **Мин = LL, Макс = HL** – если выбранный аргумент привязан к значению канала, то при установке этому атрибуту значения TRUE пределы шкалы устанавливаются равными соответственно нижнему и верхнему пределам канала. Если заливка многослойная, то пределы шкалы установятся равными соответственно **LL** и **HL** первой по порядку привязки.
- **Цвета для диапазонов** – если выбранный аргумент привязан к значению канала, то при установке этому атрибуту значения TRUE с помощью цветовых атрибутов **Предупреждение, Авария, Вне границ** можно задать дополнительные цвета заливки, соответствующие нахождению значения канала в определенном диапазоне (см. **Границы и интервалы канала FLOAT** и **Канал класса DOUBLE FLOAT**): При многослойной заливке значение атрибута **Цвета для диапазонов** должно быть FALSE.

В зависимости от значений **Мин** и **Макс** возможны 2 варианта индикатора, создаваемого из ГЭ.

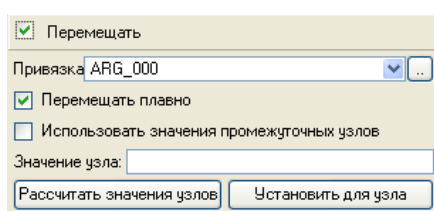
- **Вариант 1: Мин=0, Макс** больше или равно максимально возможной сумме аргументов. В этом случае ГЭ отображает абсолютный вклад аргументов в их общую сумму.
- **Вариант 2: МАХ > МИН > 0**. Этот вариант предназначен для решения специальных задач отображения. Примером такой задачи может служить отображение в заданном диапазоне уровней несмешивающихся жидкостей в емкости, если в аргументы передаются

толщины слоев жидкостей. Диапазон отображаемых уровней задается параметрами **МИН** и **МАКС**.

ГЭ 'Овал' и **ГЭ 'Емкость'** могут отображать слои в виде наложенной гистограммы прямоугольной формы. Для этих ГЭ вкладка содержит атрибуты **Гистограмма** (задает расположение гистограммы в пределах ГЭ) и **Ширина гистограммы**. Если **Гистограмма** \diamond **Вся фигура**, высота гистограммы равна высоте цилиндрической части ГЭ.

Динамическое перемещение ГЭ

Это свойство настраивается в разделе **Перемещать** вкладки **Динамическая трансформация** (🎨) окна **Свойства объекта**:



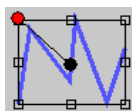
Чтобы использовать данное динамическое свойство, надо установить флаг **Перемещать**.

При работе в реальном времени графический элемент перемещается вдоль траектории, которая задается как ломаная линия (количество узлов ломаной не ограничено). Текущее положение ГЭ зависит от значения привязанного аргумента (числовой аргумент для привязки выбирается в списке **Привязка**), от значений, заданных для узлов траектории, и флага **Перемещать плавно**.


Задание траектории перемещения

Под заданием траектории понимается задание положения ее узлов и задание значений для этих узлов.

По умолчанию траектория динамического перемещения представляет собой отрезок от точки привязки ГЭ до центра ограничивающего прямоугольника, т.е. имеет 2 узла. Значения для этих узлов устанавливаются равными 0 и 100 соответственно (значение, заданное для узла, отображается в окне **Значение узла** при наведении курсора на узел):



С помощью метода **drag-and-drop** (см. **Типовые средства редакти-**

рования) положение узлов траектории на экране можно изменять (при наведении на узел курсор принимает вид , для выделения узла нужно нажать на нем ЛК).

Чтобы добавить новый узел, нужно выделить один из имеющихся узлов и далее использовать метод **drag-and-drop** при нажатой клавише **CTRL** (при этом для узла-потомка устанавливается значение, которое задано для узла-родителя).

Узел, первоначально размещенный в точке привязки ГЭ (этот узел обозначается красной точкой), остается крайним узлом при любых манипуляциях с траекторией и в дальнейшем называется первым узлом.

Значения для крайних узлов траектории задаются вручную. Для этого нужно выделить крайний узел, ввести число в окне **Значение узла** и нажать кнопку **Установить для узла**.

Значения для промежуточных узлов траектории могут быть заданы вручную или рассчитаны автоматически.

Чтобы задать значения для промежуточных узлов вручную, надо установить флаг **Использовать значения промежуточных узлов** и далее задавать значение для каждого промежуточного узла аналогично заданию значения для крайнего узла.

Значения для узлов должны монотонно возрастать (убывать) от одного крайнего узла до другого.

Чтобы задать значения для промежуточных узлов автоматически, надо сбросить флаг **Использовать значения промежуточных узлов** или при установленном флаге **Использовать значения промежуточных узлов** нажать кнопку **Рассчитать значения узлов** – в обоих случаях значения для промежуточных узлов рассчитываются исходя из значений, заданных для крайних узлов, и общей длины траектории.

Задание режима перемещения

Если флаг **Перемещать плавно** не установлен, при работе в реальном времени графический элемент скачкообразно перемещается от узла к узлу, располагаясь в каждый момент времени на том узле, для которого задано значение, ближайшее к текущему значению привязанного аргумента.

Если флаг **Перемещать плавно** установлен, автоматически рассчитывается значение для каждого пикселя траектории, при этом расчет зависит от флага **Использовать значения промежуточных узлов**:

- Если флаг **Использовать значения промежуточных узлов**

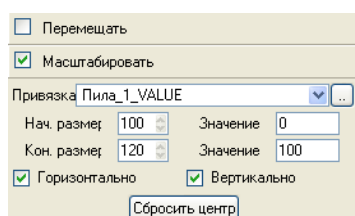
не установлен, значение для пикселей рассчитывается исходя из значений, заданных для крайних узлов и общей длины траектории.

- Если флаг **Использовать значения промежуточных узлов** установлен, значение для пикселей рассчитывается на каждом отрезке траектории исходя из его длины и значений, заданных для его узлов.

При установленном флаге **Перемещать плавно** положение графического элемента привязано к пикселю, значение которого имеет наименьшее отклонение от текущего значения привязанного аргумента.

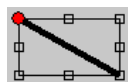
Динамическое масштабирование ГЭ

Это свойство настраивается в разделе **Масштабировать** вкладки **Динамическая трансформация** (🌈) окна **Свойства объекта**:



Для перехода в этот раздел надо нажать ЛК на заголовке раздела. Чтобы данное динамическое свойство было использовано при работе в реальном времени, надо установить флаг **Масштабировать**.

Применение свойства **Масштабировать** изменяет координаты точек ГЭ относительно **центра масштабирования**, который по умолчанию располагается в точке привязки ГЭ (в режиме редактирования центр масштабирования отображается на экране в виде красной точки):



С помощью метода **drag-and-drop** (см. **Типовые средства редактирования**) центр масштабирования может быть перемещен в произвольную точку экрана.

При нажатии кнопки **Сбросить центр** центр масштабирования устанавливается в точку пересечения диагоналей прямоугольника, ограничивающего ГЭ.

Координата точки ГЭ относительно центра масштабирования зависит от значения привязанного аргумента числового формата следующим образом (аргумент для привязки выбирается в списке **Привязка**):

$$\arg_n < \arg_{\min}, r_n - r_c = \frac{\text{MIN}}{100} (r_0 - r_c)$$

$$\text{arg}_{\min} \leq \text{arg}_n \leq \text{arg}_{\max}, \quad r_n - r_c = \frac{r_0 - r_c}{100} \left[\text{MIN} + \frac{\text{MAX} - \text{MIN}}{(\text{arg}_{\max} - \text{arg}_{\min})} (\text{arg}_n - \text{arg}_{\min}) \right]$$

$$\text{arg}_n > \text{arg}_{\max}, \quad r_n - r_c = \frac{\text{MAX}}{100} (r_0 - r_c)$$

где

r_c – координата центра масштабирования в пикселях;

r_0 – координата точки при размещении ГЭ на экране в пикселях;

r_n – текущая координата точки ГЭ (на n -ом такте обновления графического экрана) в пикселях;

MIN и MAX – пределы диапазона масштабирования (задаются в процентах в полях **Нач.размер** и **Кон.размер**);

arg_{\min} и arg_{\max} (arg_{\max} должно быть больше arg_{\min}) – значения аргумента, которым соответствуют MIN и MAX; задаются в соответствующих полях **Значение**.

Для изменения по указанному закону абсциссы нужно установить флаг **Горизонтально**, для изменения ординаты – флаг **Вертикально**.

Частный случай динамического масштабирования – изменение размеров ГЭ – имеет место при установке центра масштабирования в центр прямоугольника, ограничивающего ГЭ.

Динамическое вращение ГЭ

Это свойство настраивается в разделе **Вращать** вкладки **Динамическая трансформация** (🌈) окна **Свойства объекта**:

Для перехода в этот раздел надо нажать ЛК на заголовке раздела. Чтобы данное динамическое свойство было использовано при работе в реальном времени, надо установить флаг **Вращать**.

Угол поворота ГЭ отсчитывается от оси **X** (положительное направление – по часовой стрелке) и зависит от значения привязанного аргумента числового формата следующим образом (аргумент для привязки выбирается в списке **Привязка**):

$$\arg_n < \arg_{\min}, \varphi_n = \varphi_1$$

$$\arg_{\min} \leq \arg_n \leq \arg_{\max}, \varphi_n = \varphi_1 + \frac{\varphi_2 - \varphi_1}{\arg_{\max} - \arg_{\min}} (\arg_n - \arg_{\min})$$

$$\arg_n > \arg_{\max}, \varphi_n = \varphi_2$$

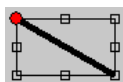
где

φ_n и \arg_n – текущие значения угла и аргумента (на n-ом такте обновления графического экрана);

φ_1 и φ_2 – границы диапазона вращения (задаются в градусах соответственно в полях **Нач.угол** и **Кон.угол**);

\arg_{\min} и \arg_{\max} (\arg_{\max} должно быть больше \arg_{\min}) – значения аргумента, которым соответствуют φ_1 и φ_2 ; задаются в соответствующих полях **Значение**.

Точка, относительно которой вращается ГЭ (**центр вращения**), по умолчанию располагается в точке привязки ГЭ (в режиме редактирования данного динамического свойства центр вращения отображается на экране в виде красной точки):



С помощью метода **drag-and-drop** (см. **Типовые средства редактирования**) центр вращения может быть перемещен в произвольную точку экрана (в том числе за пределами ГЭ).

При нажатии кнопки **Сбросить центр** центр вращения устанавливается в точку пересечения диагоналей прямоугольника, ограничивающего ГЭ.

Динамический контур ГЭ

Динамический контур представляет собой прокручиваемый по часовой стрелке пунктир (под прокруткой здесь подразумевается дискретное перемещение с шагом, равным длине штриха). Это свойство настраивается на вкладке **Динамический контур** (🔗) окна **Свойства объекта**:

| Свойство | Значение |
|------------------|---|
| Привязка | ARG_000 |
| Цвет штриха |  |
| Цвет промежутка |  |
| Длина штриха | 5 |
| Промежуток/штрих | 1 |

На вкладке размещены следующие инструменты настройки динамическо-

го контура:

- **Привязка** – задание привязки к аргументу экрана (аргумент должен иметь числовой формат). От значения привязанного аргумента зависит скорость прокрутки контура. Если аргумент равен 1, контур перемещается на 1 шаг на каждом такте обновления экрана; если аргумент равен 2, контур перемещается на 1 шаг 1 раз за 2 такта, и т.д. Если аргумент равен 0, контур не прокручивается.
- **Цвет штриха** – выбор цвета штриха.
- **Цвет промежутка** – выбор цвета промежутка.
- **Длина штриха** – задание длины штриха (и, соответственно, шага перемещения контура) в пикселях (2-100).
- **Промежуток/штрих** – задание отношения длины промежутка к длине штриха (1-10).

Функции управления ГЭ

Функции управления ГЭ – это действия, заданные для графических элементов на этапе редактирования проекта АСУ; выполнение этих действий при работе в реальном времени инициализируется оператором с помощью мыши. Задание функций управления для графических элементов придает графическим экранам свойство интерактивности и обеспечивает одно из важнейших качеств АСУ – управление техпроцессом с помощью графических средств.

Функции управления задаются на вкладке **Действия** (🖱️) окна **Свойства объекта** (см. также **Операции с аргументами в РПД**):

| Свойство | Значение |
|----------------|----------|
| Код доступа | 0 |
| События | |
| mousePressed | |
| Подтверждение | False |
| Сигнал | False |
| mouseReleased | |

Определены следующие **события**, по которым инициализируется выполнение действий в реальном времени:

- **mousePressed** (нажатие ЛК на ГЭ);
- **mouseReleased** (отжатие ЛК на ГЭ);

Для каждого из событий может быть независимо задано несколько функций управления, выбираемых из контекстного меню (меню открывается при нажатии ПК мыши на названии события):

- **передать значение**
- **показать/скрыть элементы**
- **перейти на экран**
- **послать комментарий**
- **послать подсказку**
- **послать строку**
- **выполнить**
- **передать в атрибут 46** (см. **Атрибуты каналов, отображаемые профайлером**), работает при привязке к любому атрибуту канала.

Функции управления отображаются в виде новых разделов списка свойств объекта (для каждой функции создается отдельный раздел). Для удаления функции управления или изменения ее позиции в списке используется контекстное меню, вызываемое нажатием ПК мыши на названии функции. Если для события задано несколько функций, в реальном времени они обрабатываются по порядку в соответствии с позицией в списке

(функция перехода на экран всегда выполняется последней).

Для каждого события можно задать подтверждение и звуковой сигнал (атрибуты **Подтверждение** и **Сигнал**). Если подтверждение задано и событие произошло, MPB отображает диалог, в котором можно подтвердить или отменить выполнение функций управления, заданных для данного события. Заголовок и сообщение диалога конфигурируются (соответственно атрибуты **Заголовок** и **Текст** раздела **Подтверждение**).


Если **Сигнал**=TRUE, то при выполнении заданных функций MPB воспроизводит **Стандартный звук**, заданный в Windows.

Код доступа – код доступа к использованию функций управления. Права на доступ к функциям управления задаются для пользователя в виде маски в разделе **Доступ / Формы** канала **Пользователь** (см. **Канал класса ПОЛЬЗОВАТЕЛЬ**). При корреляции маски с кодом доступа (результат побитового логического умножения отличен от нуля) доступ к функциям управления разрешен, в противном случае – запрещен. Код доступа к использованию функций управления отображается в таблице графических элементов (см. **Таблица 'Графические элементы'**).

Если пользователи в системе не заданы, значение кода доступа не учитывается.

Если ни один бит маски канала **Пользователь** не выделен, доступ к функциям управления разрешен только при значении кода доступа 0.

Если для пользователя задана некоторая маска, то для его доступа к функциям управления с кодом 0 нужно установить соответствующий флаг (см. **Канал класса ПОЛЬЗОВАТЕЛЬ**).

При наведении на ГЭ с функцией управления курсор принимает вид . Для ГЭ с функцией управления может быть также сконфигурировано выделение в реальном времени (см. **Статические атрибуты ГЭ**).

Выполнение функций управления в точке перекрытия графических элементов

Функции управления в точке перекрытия группы ГЭ выполняются в соответствии со следующими правилами:

- если группа не содержит видимых оконных ГЭ (см. **Размещение ГЭ**), режим выполнения функций управления зависит от значения ключа **GTHSPRE** в файле *.cnf (см. **Задание параметров работы мониторов**);

- если группа содержит видимые оконные ГЭ, выполняются функции управления верхнего видимого оконного элемента.

Функция передачи значения

Функция передачи значения используется для изменения значения аргументов экрана.

Для одного ГЭ можно задать несколько функций передачи значения, применительно к различным аргументам.

При добавлении этой функции управления (см. **Функции управления ГЭ**) в списке свойств объекта появляется следующий раздел:

| Свойство | Значение |
|--------------------------|----------|
| Код доступа | 0 |
| События | |
| pressed | |
| Подтверждение | False |
| Сигнал | False |
| Передать значение | |
| Тип передачи | Прямая |
| Значение | 0 |
| Результат | ... |
| Источник | ... |
| Восстанавливать значение | False |
| released | |

Поле **Тип передачи** содержит следующие варианты для передачи значений:

- Прямая
- Ввести и передать
- XOR
- OR
- AND
- Добавить
- Добавить процент шкалы
- Умножить
- Разделить

Атрибут **Источник** задает исходный аргумент, с которым проводится выбранная операция. Результат операции записывается в аргумент, задаваемый атрибутом **Результат**. Атрибуты **Источник** и **Результат** могут иметь привязку к одному и тому же аргументу.

Атрибут **Восстанавливать значение** используется только для ГЭ, запускающих выполнение действий по нажатию мыши. Если этот атрибут имеет значение **True**, то при нажатии ЛК значение аргумента будет изменено, а при последующем отпускании – восстановлено обратно.

Прямая $\langle \text{Результат} \rangle = \langle \text{Значение} \rangle$ **Ввести & передать** $\langle \text{Результат} \rangle = \langle \text{введенное значение} \rangle$

Значение задается в диалоге, который появляется на экране, если произошло заданное **событие**. Если к аргументу привязан канал, имя этого канала указывается в заголовке диалога.

НЕ-ИЛИ $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle \wedge \langle \text{Значение} \rangle$

\wedge – побитовое "исключающее ИЛИ".

ИЛИ $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle | \langle \text{Значение} \rangle$

$|$ – побитовое "ИЛИ".

И $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle \& \langle \text{Значение} \rangle$

$\&$ – побитовое "И".

Добавить $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle + \langle \text{Значение} \rangle$ **Добавить процент шкалы** $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle + 0.01 * \langle \text{Значение} \rangle * (\text{HL} - \text{LL})$

HL и LL – границы канала, привязанного к аргументу-источнику.

Умножить $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle * \langle \text{Значение} \rangle$ **Разделить** $\langle \text{Результат} \rangle = \langle \text{Источник} \rangle / \langle \text{Значение} \rangle$

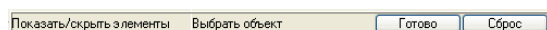
Функция управления видимостью ГЭ

С помощью этой функции можно управлять видимостью одного или нескольких ГЭ.

При добавлении функции **Показать/скрыть элементы** (см. **Функции управления ГЭ**) в списке настроек появляется соответствующая строка:

| События | |
|--------------------------|---------|
| MousePress | |
| Подтверждение | False |
| Сигнал | False |
| Показать/скрыть элементы | Число=0 |
| MouseRelease | |

Для выбора управляемых ГЭ нужно вначале нажать ЛК в поле **Значение** строки **Показать/скрыть элементы**. При этом строка примет следующий вид:



Далее с помощью ЛК на графическом экране нужно выбрать ГЭ (или несколько ГЭ, нажимая на них ЛК при нажатой клавише **Ctrl**). Для завершения процедуры нужно нажать кнопку **Готово** в строке **Показать/скрыть элементы**.

В результате в поле **Значение** появится текст **Число=N**, где **N** – число выделенных ГЭ.

При работе в реальном времени каждое возникновение указанного **события** скрывает/отображает выбранные ГЭ.

Чтобы модифицировать набор выбранных ГЭ, нужно нажать ЛК в поле **Значение** строки **Показать/скрыть элементы** и добавить/снять выделение для соответствующих элементов. Кнопка **Сброс** очищает весь список выбранных ГЭ.

Функция перехода на экран

С помощью этой функции реализуется переход на выбранный экран при наступлении заданного **события** (см. также **ГЭ 'Кнопка'**).

При создании функции управления (см. **Функции управления ГЭ**) в списке настроек появляется атрибут **Перейти на экран**. При нажатии ЛК в поле **Значение** этого атрибута появляется список, в котором можно выбрать экран (шаблон или канал вызова), на который требуется перейти.

Выбор шаблона равнозначен выбору канала вызова этого шаблона с младшим ID.

Если экран не выбран, выполняется переход на предыдущий экран (только для основных экранов).

Раздел **Перейти на экран** содержит атрибут **Пропорционально**. Если **Пропорционально** = TRUE, после перехода на экран его окно прокручивается так, чтобы центр видимой области имел координаты центра ГЭ, инициировавшего переход.

Переключением экранов в реальном времени можно также управлять с помощью каналов их вызова (см. **Особенности вызова графического экрана**).

Функция ввода комментария

Функция отправки комментария позволяет записать произвольное сообщение в отчет тревог (см. **Архивирование / Отчет тревог узла**).

При добавлении функции (см. **Функции управления ГЭ**) в списке настроек появляется строка **Послать комментарий**. Комментарий задается в диалоге, который появляется на экране, если произошло заданное событие.

Функция отправки всплывающей подсказки

Функция отправки подсказки передает значение всплывающей подсказки ГЭ в аргумент (тип данных аргумента должен быть STRING).

При добавлении функции (см. **Функции управления ГЭ**) в списке настроек появляется атрибут **Результат**. Всплывающая подсказка ГЭ записывается в аргумент, привязанный к этому атрибуту.

Функция отправки строки

Функция отправки строки передает значение заданной строки в выбранный аргумент (тип данных аргумента должен быть STRING).

При добавлении функции **Послать строку** (см. **Функции управления ГЭ**) в списке настроек появляются атрибуты **Строка** и **Результат**. Строка, заданная в поле **Значение** атрибута **Строка**, записывается в аргумент, привязанный к атрибуту **Результат**.

Функция 'Выполнить'


Функция **Выполнить** позволяет выполнить программу при наступлении заданного события.

При создании функции управления (см. **Функции управления ГЭ**) в списке настроек появляется атрибут **Выполнить**. При нажатии ЛК в поле **Значение** этого атрибута открывается список, в котором выбирается канал вызова программы (OUTPUT или INPUT OFF).

Описание встроенных графических элементов

Группа ГЭ 'Линии'

ГЭ 'Линия'

ГЭ **Линия**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Группа ГЭ 'Текст'

ГЭ 'Текст'

ГЭ **Текст** ^{ABC} не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

При динамизации атрибуты автоматически привязываются к основной привязке, если она задана.

При задании формата вывода можно добавить поясняющий текст, который отображается в реальном времени:


`Generic` `%g generic` – в ИС;

`3 generic` – в МРВ.

Конфигурация для отображения комментария, размерности и типа сигнала (атрибуты 80, 82 и 83) в консолях: тип данных аргумента – REAL, **Формат = По умолчанию**.

Группа ГЭ ‘Каналы’

ГЭ ‘Канал’

ГЭ **Канал**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ должен быть привязан к аргументу экрана (атрибут **Привязка**).

ГЭ отображает значение привязанного аргумента, а если к аргументу привязан канал (**ch**), то дополнительно может отображаться **ch.NAME** (атрибут 127), **ch.CODE** (79)/**ch.CMNT** (80) и **ch.DIM** (единицы измерения, атрибут 82).

Динамические свойства ГЭ:


- текст мигает дополнительным цветом при (4, **I**) \neq 0;
- при выключении канала цвет текста определяется атрибутом **Цвет текста (OFF)**;
- фон мигает дополнительным цветом при (7, **P**) \neq 0.

Для прекращения мигания нужно нажать ЛК на ГЭ, для квитирования последнего сообщения в ОТ по каналу и посылки 1 в атрибут 46 – ЛК с удержанием CTRL.


При нажатии ПК на ГЭ на экране появляется диалог задания границ привязанного канала.

Группа ГЭ 'Меню'

ГЭ 'Меню управления'

ГЭ **Меню управления**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ представляет собой прямоугольник, разделенный на верхнюю и нижнюю части (в верхней части отображается заданный заголовок). В реальном времени при нажатии ЛК на этом прямоугольнике отображается меню заданных действий управления. Для запуска действия нужно нажать ЛК на соответствующем пункте меню; статус действия (информация о состоянии выполнения) отображается в нижней части прямоугольника.

Действия конфигурируются в одноименном разделе вкладки **Осн. свойства**  окна свойств ГЭ. Чтобы добавить действие, нужно выполнить команду **Действие** из контекстного меню раздела **Действия**, при этом в разделе создается раздел **Действие <n>**. Для удаления действия нужно выполнить команду **Удалить** из контекстного меню раздела конфигурирования этого действия.

В поле **Значение** строки **Действие <n>** задается строка описания действия, которая отображается как пункт меню. Раздел **Действие <n>** содержит подраздел **Посылка** (содержит атрибуты **Привязка** и **Значение**) и подраздел **Проверка** (содержит атрибуты **Привязка** и **Значение**). К специфическим атрибутам ГЭ относится также атрибут **Таймаут, сек.**

При запуске действия в реальном времени:

- значение **Посылка.Значение** записывается в переменную, заданную атрибутом **Посылка.Привязка**;
- одновременно запускается обратный секундомер (начальное значение – **Таймаут секунд**), и 1 раз в секунду значение переменной, заданной атрибутом **Проверка.Привязка**, проверяется на равенство значению **Проверка.Значение**; при этом статус действия имеет следующий формат:


<строка описания действия> – <текущее значение секундомера> сек

В этом состоянии запуск других действий невозможен;


- если в течение **Таймаут секунд** значение переменной **Проверка.Привязка** становится равным значению **Проверка.Значение**, секундомер останавливается без сообщения;
- в противном случае меняется цвет заливки ГЭ.

Группа ГЭ ‘Ломаные и кривые’


ГЭ ‘Ломаная линия’

ГЭ **Ломаная линия**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Многоугольник’


ГЭ **Многоугольник**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Ломаная с заливкой’


Данный ГЭ  представляет собой ломаную линию, для которой, в отличие от ГЭ ‘Ломаная линия’, цвет контура и цвет заливки задаются независимо.

ГЭ не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Разомкнутая кривая’


ГЭ **Разомкнутая кривая**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Замкнутая кривая’


ГЭ **Замкнутая кривая**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ** и **Задание типовых свойств ГЭ**).

Группа ГЭ ‘Прямоугольники’


ГЭ ‘Контур’

ГЭ **Контур**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Прямоугольник’


ГЭ **Прямоугольник**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Панель’

ГЭ **Панель**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), ГЭ **Панель** имеет ряд специфических атрибутов (см. **Специфические атрибуты ГЭ ‘Панель’ и ‘Рамка’**):

ГЭ ‘Рамка’

ГЭ **Рамка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), ГЭ **Рамка** имеет ряд специфических атрибутов (см. **Специфические атрибуты ГЭ ‘Панель’ и ‘Рамка’**):

Специфические атрибуты ГЭ ‘Панель’ и ‘Рамка’

ГЭ **Панель** и **Рамка** имеют ряд специфических атрибутов:


- **Заливка** – если **False**, ГЭ становится прозрачным.

Цветовые атрибуты отображаются в окне свойств только в том случае, если **Системные цвета=FALSE**.


- **Утопленный** – этот атрибут определяет вид ГЭ – **выступающий (False)** или **утопленный (True)**.

Группа ГЭ 'Плоские фигуры'


ГЭ 'Плоский клапан'

ГЭ **Плоский клапан**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ 'Треугольник'


ГЭ **Треугольник**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ 'Овал'


ГЭ **Овал**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**). Помимо стандартных атрибутов, ГЭ имеет несколько специфических.

Атрибут **Высота краев (%)** задает высоту округлых краев в процентах от ширины ГЭ (0-100%). Атрибуты **Верхний край** и **Нижний край** определяют наличие соответствующих округлых краев (TRUE/FALSE).

ГЭ 'Стрелка'

ГЭ **Стрелка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ 'Эллипс, сектор'

ГЭ **Эллипс, сектор**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Специфическими атрибутами эллипса являются **Начальный угол** и **Сектор**. Оба параметра задаются как углы в градусах (0-360) в окне свойств.

Атрибут **Начальный угол** (задается как угол, отложенный против часовой стрелки от оси X) задает направление, от которого отсчитывается отображаемая часть эллипса (задается атрибутом **Сектор** как угол, отложенный против часовой стрелки).


Атрибут **Начальный угол** не изменяет направления осей эллипса, которые, в отсутствие поворота ГЭ, совпадают с

направлением координатных осей экрана:


При размещении ГЭ **Начальный угол** равен 0, эллипс отображается полностью (**Сектор** = 360°).

Динамизация параметра **Сектор** задается с помощью атрибута **Привязка**. При нажатии в поле его значения вызывается стандартный диалог выбора аргумента (см. **Табличный редактор аргументов**).

ГЭ 'Плоский клапан 2'

Данный ГЭ  представляет собой плоский клапан с приводом (см. ГЭ 'Плоский клапан'). ГЭ снабжен типовыми атрибутами конфигурирования формы.


ГЭ 'Плоский насос'

ГЭ **Плоский насос**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**), конфигурируется аналогично ГЭ 'Плоский клапан 2'.

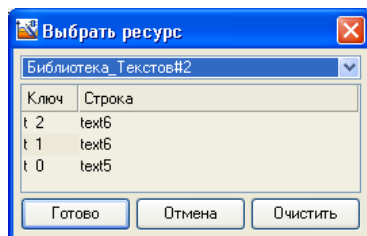
Группа ГЭ 'Ресурсы'

ГЭ этой группы размещаются на графическом экране стандартным способом (см. **Размещение ГЭ**); ресурс из библиотеки выбирается в навигаторе (см. **Операции с ресурсными библиотеками**), который появляется на экране при выборе элемента на панели инструментов **Графические элементы**.

ГЭ 'Текстовый ресурс'

Выбранный в ходе размещения ГЭ **Текстовый ресурс**  ресурс (см. **Группа ГЭ 'Ресурсы'**) устанавливается в качестве статического значения атрибута **Текст** (это единственный специфический атрибут данного ГЭ).


При нажатии ЛК в поле значения этого атрибута на экране появляется следующий диалог:



С помощью этого диалога можно задать другое статическое значение атрибута **Текст**, в том числе пустое (по команде **Очистить**).

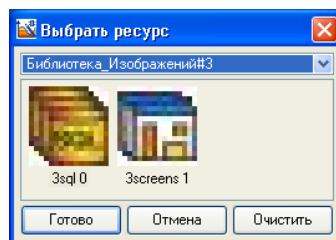
Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.

ГЭ 'Растровое изображение'

Выбранный в ходе размещения ГЭ **Растровое изображение**  ресурс (см. **Группа ГЭ 'Ресурсы'**) устанавливается в качестве статического значения атрибута **Изображение** (это единственный специфический атрибут данного ГЭ).

Если размер рисунка меньше 20 px, при размещении он увеличивается до этой величины.


При нажатии ЛК в поле значения этого атрибута на экране появляется следующий диалог:



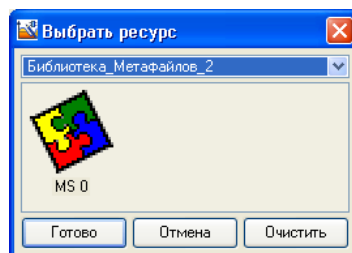
С помощью этого диалога можно задать другое статическое значение атрибута **Изображение**, в том числе пустое (по команде **Очистить**).

Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.

ГЭ 'Векторное изображение'

Выбранный в ходе размещения ГЭ **Векторное изображение**  ресурс (см. **Группа ГЭ 'Ресурсы'**) устанавливается в качестве статического значения атрибута **Изображение** (это единственный специфический атрибут данного ГЭ).


При нажатии ЛК в поле значения этого атрибута в окне свойств на экране появляется следующий диалог:



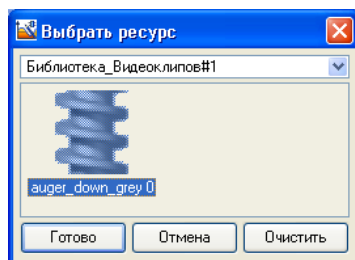
С помощью этого диалога можно задать другое статическое значение атрибута **Изображение**, в том числе пустое (по команде **Очистить**).

Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.


ГЭ 'Видеоклип'

Выбранный в ходе размещения ГЭ **Видеоклип**  ресурс (см. **Группа ГЭ 'Ресурсы'**) устанавливается в качестве статического значения атрибута **Видеоклип**.

При нажатии ЛК в поле значения этого атрибута в окне свойств на экране появляется следующий диалог:




С помощью этого диалога можно задать другое статическое значение атрибута **Видеоклип**, в том числе пустое (по команде **Очистить**).

Помимо атрибута **Видеоклип**, на вкладке  **Осн. свойства** конфигурируются следующие специфические атрибуты данного ГЭ:

- **Привязка** – предназначен для задания условия воспроизведения видеоклипа по значению привязанного аргумента (см. **Динамизация атрибута ГЭ**). Если значение аргумента не равно 0, видеоклип воспроизводится, если 0 – не воспроизводится;
- **Пауза** – определяет паузу между кадрами при воспроизведении клипа. Этот параметр задается в тактах обновления экрана (значение статического параметра по умолчанию – 0);
- **Показывать при остановке** – если этот атрибут имеет значение **True** (значение по умолчанию), после остановки воспроизведения ГЭ отображает первый кадр клипа; если **False**, после остановки воспроизведения клип становится невидимым;
- **Непрерывное воспроизведение** – если этот атрибут имеет значение **True** (значение по умолчанию), клип воспроизводится циклически; если **False**, клип воспроизводится однократно.


Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.

ГЭ ‘Рисунок из файла’

ГЭ **Рисунок из файла**  размещается на графическом экране стандартным способом (см. **Размещение ГЭ**). Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.

Для отображения ищется файл с именем канала, привязанного к используемому аргументу экрана (атрибут **Привязка**). Поиск производится в директории **<имя файла prj без расширения>** (см. **Сохранение проекта для запуска**).

ГЭ ‘Текст из файла’

ГЭ **Текст из файла**  размещается на графическом экране стандарт-

ным способом (см. **Размещение ГЭ**). Типовые свойства ГЭ описаны в разделе **Задание типовых свойств ГЭ**.

Для отображения ищется файл с именем канала (в том числе канала вызова шаблона документа), привязанного к используемому аргументу экрана (атрибут **Привязка**). Поиск производится в папке узла.

Файл должен иметь текстовый (расширение **.txt**) или гипертекстовый формат (расширение **.html**, кодировка должна быть **utf-8**). Формат отображаемого файла должен быть задан для ГЭ (атрибут **Формат файла**).

Если к ГЭ привязан канал вызова шаблона документа, ГЭ отображает вновь сгенерированный документ автоматически, в противном случае для перечитывания файла нужно изменить значение привязанного канала.

ГЭ может быть привязан к аргументу экрана, не привязанному к каналу. В этом случае ГЭ будет отображать текстовый или гипертекстовый файл (из папки узла) с именем канала вызова экрана и указанным расширением.

При отображении гипертекстового файла отображается текст и картинки, отображаются и обрабатываются гиперссылки (на HTML-файлы).

Отображение гиперссылок задается с помощью атрибута **Подчеркивать ссылки** (с подчеркиванием или без подчеркивания).

В реальном времени возможно переключение ГЭ на отображение любого txt- или html-файла – для этого нужно использовать механизмы перепривязки аргументов канала вызова экрана.

Если **Использовать IE=TRUE**, для отображения текста или гипертекста ГЭ использует MS Internet Explorer. В этом случае атрибут **Шрифт** не работает, а для отображения файлов *.txt IE использует свой шрифт по умолчанию.

Использование функций вывода





ГЭ **Текст из файла** позволяет отображать HTML-код, генерируемый в памяти (см. **Номер SubNum**). Конфигурация ГЭ:

- **Формат файла** – **html**;
- **Привязка** – **arg.<SubNum>** (см. **Динамизация атрибута ГЭ**).

Для отображения канала CALL.TVC/CALL.ChGroupReq к аргументу **arg** графического экрана должен быть привязан атрибут 120, **АСК** или 45, **T** канала.

ГЭ 'Стандартный видеоклип'

РПД включает несколько подобных ГЭ, отображающих встроенные видеоклипы:

- **Вентилятор1**  (стандартный видеоклип 0)
- **Вентилятор2**  (стандартный видеоклип 1)
- **Лампочка**  (стандартный видеоклип 2)
- **Пламя**  (стандартный видеоклип 3)

ГЭ размещается на графическом экране стандартным способом (см. **Размещение ГЭ**, а также **Задание типовых свойств ГЭ** и ГЭ 'Видеоклип').

Видеоклип воспроизводится, если выражение, заданное атрибутом **Вид индикации**, равно TRUE (см. **Динамизация атрибута ГЭ**).

Группа ГЭ ‘Объемные фигуры’

Общие специфические атрибуты объемных ГЭ

ГЭ данной группы размещаются в графическом слое стандартным способом (см. **Размещение ГЭ**).

Специфическими атрибутами, общими для всех ГЭ группы **Объемные фигуры**, являются следующие:

| Материал | | Материал | |
|--------------------------|-------|--------------------------|----------------|
| Выбрать из списка | False | Выбрать из списка | True |
| Базовый цвет | | Материалы | Бронза(полиц.) |
| Диффузное отражение | 100 | Стандартные текстуры | 1 |
| Угловая ширина блика | 0 | Прозрачность | 0 |
| Зеркальное отражение | 0 | Текстура | |
| Прозрачность | 0 | Масштабирование текстуры | False |
| Текстура | | Качество (%) | 0 |
| Масштабирование текстуры | False | Толщина стенок | 0 |
| Качество (%) | 0 | | |
| Толщина стенок | 0 | | |

В TRACE MODE используется модель освещения объемных ГЭ одним источником белого света (положение источника задается в диалоге **Параметры экрана** – см. **Задание параметров графического экрана**). Цветовые и отражательные характеристики ГЭ задаются с помощью атрибутов раздела **Материал**. Если значение атрибута **Выбрать из списка** – FALSE (значение по умолчанию), то цветовые и отражательные характеристики ГЭ задаются с помощью следующих атрибутов (соответствуют функциям OpenGL):

- **Диффузное отражение (dc)** – коэффициент (0-100), определяющий цвет диффузно отраженного света. Во внутреннем представлении значения RGB диффузно отраженного света равны соответствующим значениям RGB базового цвета (задаются с помощью атрибута **Базовый цвет**), умноженным на **0.01dc**.
- **Угловая ширина блика (k)** – параметр, задающий угол раскрытия зеркально отраженного света. Угол увеличивается при уменьшении параметра от 128 до 0.
- **Зеркальное отражение (sc)** – коэффициент (0-100), определяющий цвет зеркально отраженного света. Во внутреннем представлении значения RGB зеркально отраженного света равны соответствующим значениям RGB базового цвета (задаются с помощью атрибута **Базовый цвет**), умноженным на **0.01sc**.

Если значение атрибута **Выбрать из списка** – TRUE, то для ГЭ задается материал (с помощью списка **Материалы**, `d3_materials.tmc`) –

предустановленное сочетание значений функций OpenGL, указанных выше, – а также одна из встроенных текстур (список **Стандартные текстуры**).

Если **Стандартные текстуры**=0, поверхность объемного ГЭ может быть текстурирована с помощью атрибута **Текстура**. При нажатии кнопки в разделе конфигурирования атрибута **Текстура** на экране появляется навигатор ресурсной библиотеки растровых рисунков (см. **Операции с ресурсными библиотеками**). В этом навигаторе выбирается рисунок, который используется в качестве текстуры.


Атрибут **Масштабирование текстуры** при значении **True** масштабирует размер изображения текстуры до размеров ГЭ. При значении **False** текстура имеет оригинальный размер.

Атрибут **Качество (%)** определяет степень прорисовки текстуры.


Если атрибут **Толщина стенок** отличен от 0, на экране отображается сечение объемного ГЭ.

В случае некорректного отображения объемных элементов на экране (черный экран, потом возможна перерисовка) и/или при перерисовке изображение «рассыпается» на квадраты, можно попробовать использовать в файле CNF ключ **GRAPH_OPENGL=MEMBUF**.

ГЭ 'Цилиндр'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Цилиндр**  имеет специфические атрибуты **Край1** и **Край2**, с помощью которых можно задавать разнообразную форму оснований.

ГЭ 'Сфера'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Сфера**  имеет специфический атрибут **Отображаемая часть**. Этот атрибут может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:


| Отображаемая часть | Часть | Отображаемая часть | Сектор |
|--------------------|-----------|--------------------|--------|
| Параметр 1 | Полностью | Начальный угол | 90 |
| Параметр 2 | Часть 1 | Угол разворота | 360 |

В первом случае **Параметр1** и **Параметр2** определяют, какая часть

должна быть отображена (если **Параметр1=Полностью**, **Параметр2** игнорируется).

Во втором случае для ГЭ задается сектор отображения. Угол разворота сектора (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и горизонтальным направлением к правой стороне экрана задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются против часовой стрелки.

ГЭ 'Конус'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Конус**  имеет специфические атрибуты **Отображаемая часть** и **Отношение оснований**.

Атрибут **Отображаемая часть** может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:


| Отображаемая часть | Часть | Отображаемая часть | Сектор |
|--------------------|-----------|--------------------|--------|
| Параметр 1 | Полностью | Начальный угол | 90 |
| Параметр 2 | Часть 1 | Угол разворота | 360 |

В первом случае **Параметр1** и **Параметр2** определяют, какая часть должна быть отображена (если **Параметр1=Полностью**, **Параметр2** игнорируется).

Во втором случае для ГЭ задается сектор отображения (на виде сверху). Угол разворота сектора основания (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и горизонтальным направлением к правой стороне вида задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются против часовой стрелки.

Значение атрибута **Отношение оснований**, % задается в процентах и определяет соотношение диаметров оснований (0-100, значение по умолчанию – 50, при значении 100 диаметр меньшего основания равен 1 рх, при значении 0 диаметры оснований равны).

ГЭ 'Тор'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Тор**  имеет специфические атрибуты **Отображаемая часть** и **Толщина**.

Атрибут **Отображаемая часть** может принимать два значения – **Часть** и **Сектор**, в зависимости от которых меняются два нижних списка:


| Отображаемая часть | Часть | Отображаемая часть | Сектор |
|--------------------|-----------|--------------------|--------|
| Параметр 1 | Полностью | Начальный угол | 90 |
| Параметр 2 | Часть 1 | Угол разворота | 360 |

В первом случае **Параметр1** и **Параметр2** определяют, какая часть должна быть отображена (если **Параметр1=Полностью**, **Параметр2** игнорируется).


Во втором случае для ГЭ задается сектор отображения. Угол разворота сектора (задается параметром **Угол разворота**) отсчитывается от некоторого начального направления; угол между начальным направлением и вертикальным направлением к нижней стороне экрана задается параметром **Начальный угол**. Оба угла задаются в градусах (0-360) и отсчитываются по часовой стрелке.

Атрибут **Толщина** определяет толщину тора и задается как число в диапазоне 0-100 (значению по умолчанию – 20, при значении 0 толщина тора равна 1 px, при значении 100 – 200 px).

ГЭ 'Пирамида'

ГЭ **Пирамида**  не имеет собственных специфических свойств (см. **Задание типовых свойств ГЭ** и **Общие специфические атрибуты объемных ГЭ**).


ГЭ 'Емкость'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Емкость**  имеет специфические атрибуты **Высота краев (%)**, **Верхний край** и **Нижний край**.

Атрибут **Высота краев** задает высоту краев емкости в процентах от ширины ГЭ (0-100%).

Атрибуты **Верхний край** и **Нижний край** определяют форму соответствующих краев.

ГЭ 'Клапан'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Клапан**  имеет ряд специфических атрибутов.


С помощью атрибута **Форма клапана** задается форма клапана, с помощью атрибута **Форма привода** – форма привода.

Если атрибут **Цвет фланцев**=TRUE, цвет фланцев клапана всегда соответствует статическому значению атрибута **Базовый цвет**. Если **Цвет фланцев**=FALSE, цвет фланцев соответствует цвету клапана.

ГЭ 'Насос'


По конфигурированию данный ГЭ идентичен ГЭ 'Клапан' за одним исключением – у него отсутствует атрибут **Форма привода**, поскольку атрибут **Форма насоса** определяет форму насоса и привода одновременно.

ГЭ 'Труба'

Данный ГЭ () размещается на графическом экране аналогично элементам группы **Ломаные и кривые** (см. **Размещение ГЭ** и **Позиционирование узловых точек ГЭ**).

Помимо типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Труба** имеет специфические атрибуты **Край 1** и **Край 2**, которые задают разнообразную форму краев трубы.

ГЭ 'Рельефный конус'

ГЭ **Рельефный конус** () аналогичен ГЭ **Конус** (см. **ГЭ 'Конус'**), но позволяет отобразить на поверхности правильный геометрический рельеф в виде линий или точек.


Рельеф формируется с помощью задания сетки на поверхности ГЭ – для этого используются следующие специфические атрибуты:

- **Параметр 1** – угловое расстояние между линиями сетки, расположенными по образующим конуса;
- **Параметр 2** – расстояние (в пикселях) между линиями сетки, перпендикулярными высоте конуса.

Атрибут **Параметр 3** задается в процентах и определяет соотношение диаметров оснований.

Если оба параметра равны 0, рельеф представляет собой набор хаотически расположенных точек. Если оба параметра отличны от 0, на поверхности конуса отображаются узлы сетки, при равенстве одного из параметров 0 – соответствующие линии сетки.

ГЭ 'Криволинейный конус'

ГЭ **Криволинейный конус**  аналогичен ГЭ **Конус** (см. ГЭ 'Конус'), но имеет криволинейную образующую.


Специфические атрибуты ГЭ имеют следующее назначение:

- атрибут **Параметр 1** задает кривизну вогнутой образующей при равенстве 0 атрибута **Параметр 2**;
- второй параметр задает кривизну выпуклой образующей при равенстве первого параметра нулю.

Если оба параметра равны 1 и атрибут **Параметр 3** равен 0, конус вырождается в сферу. При увеличении значений атрибутов **Параметр 1** и **Параметр 2** и атрибуте **Параметр 3**, равном 0, в пределах ограничивающего ГЭ прямоугольника отображается центральная часть увеличенной сферы. Если атрибуты **Параметр 1**, **Параметр 2** и **Параметр 3** не равны 0, ГЭ представляет собой эллипсоид, при этом атрибут **Параметр 3** определяет полуось в плоскости XZ (сфера, «сплюснутая» по Z).

Если атрибуты **Параметр 1** и **Параметр 2** не равны 0, то при больших значениях атрибута **Параметр 3** (> 50) в центре эллипсоида образуется отверстие.

ГЭ 'Градиент'


Данный ГЭ () представляет собой прямоугольник с однокоординатной градиентной заливкой.

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных ГЭ (см. **Общие специфические атрибуты объемных ГЭ**), ГЭ **Градиент** имеет следующие специфические атрибуты:

- **Параметр 1** – оттенок заливки;
- **Параметр 2** – зоны в верхней и нижней частях ГЭ. Размер зон задается значением атрибута (0-255px). Верхняя зона имеет более светлый тон относительно базового цвета, нижняя – более темный;
- **Параметр 3** – градиент (0 – максимум, 100 – минимум).

Группа ГЭ ‘Кнопки’

ГЭ ‘Кнопка’


ГЭ **Кнопка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Для создания в шаблоне экрана **N** кнопки с функцией перехода на экран **M** достаточно перетащить шаблон или канал вызова экрана **M** из навигатора проекта в шаблон экрана **N** (см. **Функция перехода на экран**).

ГЭ **Кнопка** имеет следующие специфические настройки, задаваемые в окне свойств:

- **Два состояния** – если этот флаг установлен, устойчивыми являются оба состояния кнопки (нажатое и отжатое), в противном случае устойчиво только отжатое состояние;
- **Привязка** – определяет аргумент для управления состоянием кнопки (нажатое и отжатое) при установленном атрибуте **Два состояния**. При значении аргумента 0 кнопка отжата, при любом другом значении – нажата;
- **Плоская** – при установке этого флага кнопка становится плоской;
- **Изображение** – кнопка выбора рисунка из ресурсной библиотеки изображений (см. **Операции с ресурсными библиотеками**);
- **Истинный размер изображения** – при установке этого флага картинка принимает свой истинный размер; если флаг не установлен, рисунок сжимается до размера 24*24 пикселя.

ГЭ ‘Группа кнопок’

ГЭ **Группа кнопок**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

В окне свойств задаются следующие специфические атрибуты ГЭ **Группа кнопок**:

- **Привязка** – привязка ГЭ к аргументу экрана (далее – **arg**);
- **Стиль кнопок** – стиль ГЭ, выбирается из меню, которое содержит следующие опции:
 - **Кнопка** – набор кнопок;
 - **Переключатель 1** – набор переключателей типа Radio Button;


- **Переключатель 2** – набор переключателей типа Checkbox;
- **Исключающий** – TRUE/FALSE, от этого атрибута зависит алгоритм работы ГЭ при **Стиль кнопок = Кнопка**;
- **Режим привязки** – выбирается из меню, которое содержит опции **Только управление** и **Управление и индикация**;
- **Плоский** – если для этого атрибута задано значение FALSE, ГЭ стиля **Кнопка** имеет вид панели инструментов. Если для атрибута **Плоский** задано значение TRUE, ГЭ стиля **Кнопка** имеет вид меню (при наведении мыши кнопка отображается);
- **Вид** – вид ГЭ, выбирается из списка:
 - выступающая панель;
 - утопленная панель;
 - утопленная рамка;
 - без рамки;
- **Кнопки** – раздел конфигурирования кнопок/переключателей ГЭ. При нажатии ПК на имени раздела на экране появляется меню, содержащее команду **Кнопка**. При выполнении этой команды кнопка/переключатель добавляется к ГЭ, и в разделе **Кнопки** появляется подраздел с именем вида **Кнопка N**, содержащий следующие инструменты конфигурирования:
 - **Текст** – текст, отображаемый на кнопке (или около переключателя);
 - **Значение** – значение (далее – **Val**), связанное с кнопкой или переключателем.

При нажатии ПК на имени подраздела (**Кнопка N**) на экране появляется меню, содержащее команды удаления/перемещения кнопки.

Алгоритм работы ГЭ:

- при нажатии кнопки $arg = Val$, если **Стиль кнопок = Кнопка** (**Исключающий = TRUE**) или **Стиль кнопок = Переключатель 1**. Кнопка переходит в «нажатое» состояние при $arg = Val$;
- при нажатии кнопки $arg = arg \wedge Val$ (\wedge – операция побитового XOR), если **Стиль кнопок = Кнопка** (**Исключающий = FALSE**) или **Стиль кнопок = Переключатель 2**. Кнопка переходит в «нажатое» состояние при $arg \& Val \triangleleft 0$ ($\&$ – операция побитового AND).

ГЭ ‘Картинка-кнопка’

ГЭ **Картинка-кнопка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ имеет следующие специфические настройки, задаваемые в окне свойств:

- **Два состояния** – если этот флаг установлен, устойчивыми являются оба состояния кнопки (нажатое и отжатое), в противном случае устойчиво только отжатое состояние;
- **Привязка** – определяет аргумент для управления состоянием кнопки при установленном флаге **Два состояния**. При значении аргумента 0 кнопка отжата, при любом другом значении – нажата;
- **Картинка (отжато)** – картинка из библиотеки изображений, отображаемая на кнопке в ее отжатом состоянии;
- **Картинка (нажато)** – картинка из библиотеки изображений, отображаемая на кнопке в ее нажатом состоянии.

Группа ГЭ ‘Выключатели’

ГЭ ‘Выключатель’

Данный ГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ**, а также **Задание типовых свойств ГЭ**) и представляет собой два переключаемых изображения («ON» и «OFF»).


Функции выключателя зависят от значения атрибута **Вид индикации** (см. **Динамизация атрибута ГЭ**).

Если **Вид индикации = Arg & Конст**, при нажатии ЛК на ГЭ вычисляется выражение $\text{arg} \wedge \text{Значение(XOR)}$ (\wedge – операция побитового XOR, **arg** – значение привязанного аргумента), и полученный результат присваивается привязанному аргументу. Если **arg & Константа = TRUE (& – операция побитового AND)**, ГЭ переходит в состояние «ON» (**Инверсия=FALSE**) или «OFF» (**Инверсия=TRUE**); если **arg & Константа = FALSE** – соответственно в состояние «OFF» или «ON».

Если **Вид индикации = Arg >= Конст** или **Вид индикации = Arg = Конст**, при нажатии ЛК на ГЭ привязанному аргументу присваивается значение, заданное атрибутом **Значение**. Если, соответственно, **arg >= Константа** или **arg == Константа**, ГЭ переходит в состояние «ON» (**Инверсия=FALSE**) или «OFF» (**Инверсия=TRUE**); в противном случае, в зависимости от атрибута **Инверсия**, – в состояние «OFF» или «ON».

Группа ГЭ ‘Тренды’

ГЭ ‘Тренд’



ГЭ **Тренд**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ **Тренд** может отображать:


- изменение значения аргументов экрана во времени;
- данные SIAD;
- данные индивидуальных архивов;
- исторические данные, полученные от серверов OPC HDA;
- табличные функции CALL.TVC (см. **Атрибуты канала класса CALL**);
- значения аргументов канала CALL.ChGroupReq (см. **Канал CALL.ChGroupReq** и **Универсальный механизм обмена с электросчетчиками**).

Аргументы с типом данных INT, DINT, REAL и LREAL отображаются на **аналоговой панели** в верхней части тренда, аргументы с другими типами данных – на **дискретной панели**, которая располагается под аналоговой.

Помимо типовых свойств (см. **Задание типовых свойств ГЭ**), тренд имеет значительное количество специфических атрибутов.

Окно свойств ГЭ содержит вкладки – **Осн. свойства**  и **Кривые** .

На вкладке **Осн. свойства** конфигурируются следующие специфические атрибуты:

- **Использовать архив** – если FALSE, тренд отображает только текущие данные (переключатель  недоступен);
- **Сетка** – этот раздел содержит типовые инструменты задания параметров сетки тренда;
- **Легенда** – этот раздел содержит типовые инструменты задания параметров легенды, отображаемой в нижней части тренда. Легенда отображает либо таблицу параметров аналоговых кривых, либо таблицу параметров дискретных кривых (для переключения нужно нажать ЛК на соответствующей панели). Строка таблицы отображает следующие параметры кривой:
 - **Активная** – активная кривая обозначается в этом столбце знаком «+». Чтобы сделать активной некоторую кривую,

нужно нажать ЛК в этом столбце строки кривой;

- **Видимость** – видимая кривая обозначается в этом столбце знаком «+». Чтобы показать/скрыть кривую (и ее ось значений), надо нажатием ЛК установить/снять этот знак в данном столбце строки кривой. Скрыть активную кривую нельзя;
- **Кривые** – образец кривой;
- **Источник** – имя кривой (имя канала, привязанного к аргументу, если для кривой не задан атрибут **Имя**);
- **Значение** – значение кривой в точке, указанной визиром;
- **Ось времени** – этот раздел содержит настройки временной оси:
 - **Показывать** – если TRUE, на оси отображаются значения времени;
 - **Разбиение** – начальное количество делений видимой части оси;
 - **Период подписи** – период подписей на оси (в делениях). Линии сетки без подписи имеют цвет, заданный атрибутом **Доп. цвет** в разделе **Сетка**;
 - **Диапазон** – диапазон значений видимой части оси (0-100);
 - **Единицы** – единицы измерения диапазона. Выбираются из меню: **секунда, минута, час, день**;
 - **Левая граница** и **Правая граница** – аргументы:
 - если **Вид передачи**=Output, в аргументы передаются текущие значения временных границ тренда;
 - если **Вид передачи**=Input, аргументы задают временные границы тренда;
- **Ось значений** – этот раздел содержит настройки оси значений:
 - **Разбиение** – начальное количество делений видимой части оси;
 - **Период подписи** – период подписей на оси (в делениях). Линии сетки без подписи имеют цвет, заданный атрибутом **Доп. цвет** в разделе **Сетка**. Формат подписей – **%g**;
 - **Показывать** – видимость осей значений:
 - **Все оси** – отображаются оси всех видимых кривых; ось активной кривой отображается крайней слева;
 - **Только активная** – отображается только ось активной кривой;
- **Буфер** – количество хранимых в памяти значений каждой кривой для вывода на тренд ($24 \cdot 10^6$, по умолчанию – 500). Кроме того, этот параметр задает максимальное число значений, извлекаемых из архива и отображаемых на тренде при переходе к заданной временной метке.

Буфер можно сохранить в файл – для этого у канала, вызывающего

шаблон экрана с трендом, нужно установить бит 1 (0x2) атрибута 58, **DumpSync** (см. **Общие атрибуты каналов**).

Буфер сохраняется в файл с именем **<ID>.DRG**, где **ID** – идентификатор канала, вызывающего шаблон экрана с трендом. Период сохранения буфера определяется параметром **Период сохранения доп. информации** в настройках узла (вкладка **Отчет тревог / Дамп / Параметры** – см. **Редактор параметров узла**).

При запуске проекта на тренде будут показаны значения последнего сохраненного буфера.

Буфер тренда, входящего в состав объекта, нельзя сохранить в файл.

- **Масштаб дискрет (%)** – этот атрибут задает высоту поля, отводимого на дискретной панели для отображения одной кривой. Высота задается в процентах ($\geq 100\%$) относительно размера базового шрифта (100%, значение по умолчанию).
- **Цвета статусов** – цвета для целочисленных аргументов при их равенстве соответственно 0...7, отображаемых на дискретной панели. Для использования этих цветов атрибуту **Интерпретировать как кривой** должно быть установлено значение **Статус**.

Чтобы получить на экране график, необходимо сконфигурировать хотя бы одну кривую на вкладке **Кривые**.

Для добавления кривой нужно выполнить команду **Добавить** из контекстного меню раздела **Кривые**, для удаления/перемещения в списке – соответствующую команду из контекстного меню раздела **КриваяN** (см. также **Операции с аргументами в РПД**).

Набор свойств кривой зависит от атрибута **Интерпретировать как**. Этот атрибут используется только при конфигурировании дискретной кривой. Он может принимать следующие значения:




- **Значение** – при равенстве аргумента 0 – тонкая линия, в противном случае – толстая линия; при изменении значения отображается метка, содержащая новое значение;
- **Статус** – толстая линия, образованная отрезками; цвет отрезка соответствует значению (задается атрибутом **Цвета статусов**); при изменении значения отображается метка, содержащая новое значение;
- **Изменение (со значением)** – толстая линия, образованная чередующимися отрезками двух различных тонов (тон меняется при изменении значения аргумента). При изменении значения отображается метка, содержащая новое значение;
- **Изменение** – метки изменения значения аргумента.

Полный набор параметров кривой (для дискретной кривой актуальны только имя, привязка, видимость и цвет):


- **Имя** – имя кривой, при задании выводится в легенду, в противном случае в легенде отображает имя канала (0, если отображаемый аргумент привязан к аргументу или не имеет привязки).
- **Привязка** – привязка кривой к аргументу экрана. Для отображения значения атрибута канала (в том числе данных по этому атрибуту из SIAD) к аргументу должен быть привязан этот атрибут; для отображения индивидуального архива – атрибут **1,A** соответствующего канала CALL (см. **Индивидуальный архив** и **MPB как клиент сервера OPC HDA**).
- **Скрыть при старте** – если TRUE, кривая (и ее ось значений) не отображается на тренде при старте;
- **Цвет** – цвет кривой.
- **Стиль линии** – стиль кривой.
- **Толщина линии** – толщина кривой в пикселях.
- **Тип меток** – выделение точки кривой с периодом изменения привязанного аргумента с помощью одной из встроенных меток;
- **Формат** – формат значения аналоговой кривой в легенде (см. **Формат Си вывода чисел**), должен соответствовать типу данных аргумента. Значение дискретной кривой всегда выводится как целое со знаком (DEC);
- **Стиль при I<>0 и W=0** – стиль линии при аппаратной/программной недостоверности значения канала, привязанного к отображаемому аргументу. При нажатии ЛК в этом поле выводится стандартный список стилей (см. **Задание типовых свойств ГЭ / Статические атрибуты ГЭ**).
- **Стиль при I=0 и W=1** – стиль для отображения значений канала при (8) W=1 (см. также **Канал CALL.Writer**).
- **Стиль при I<>0 и W=1** – стиль при аппаратной/программной недостоверности значения канала и при (8) W=1 (см. также **Канал CALL.Writer**).
- **Макс. значение** – верхний предел оси значений.
- **Мин. значение** – нижний предел оси значений.
- **Интерполяция** – этот атрибут может принимать следующие значения:
 - **нет** – нет интерполяции;
 - **по периоду реального времени** – интерполирование методом проведения прямой между текущим значением канала и его значением на предыдущем такте пересчета;
 - **простая линейная** – интерполирование методом проведения прямой между точками, соответствующими изменени-


ям значения канала.

В реальном времени доступны следующие инструменты ГЭ **Тренд**:


-  – возврат к исходному масштабу тренда;
-  – увеличение. Кнопка имеет два положения – нажатое (увеличение включено) и отжатое (увеличение отключено). Для детального просмотра некоторого участка тренда нужно в режиме увеличения выделить его мышью, удерживая ЛК;
-  – вход в диалог редактирования кривых тренда, идентичный вкладке **Кривые** окна **Свойства объекта**. Окно предназначено для добавления/удаления кривых и изменения их настроек в реальном времени. Диалог открывается, если у пользователя есть соответствующие права.


Если на экране непосредственно (не в окне) размещен графический объект, содержащий тренд, то данная функция тренда недоступна.





-  – переход к временной метке. При нажатии на эту кнопку появляется диалог задания даты и времени. После нажатия кнопки **Готово** в диалоге, на тренд выводится информация, начиная с указанного времени. Диалог снабжен флагом **Установить курсор**; если этот флаг установлен, то при переходе визир устанавливается на заданную метку времени (при этом, при необходимости, временная шкала сдвигается);

 – в это окно выводится следующая информация:

- если увеличение не включено – дата и время точки, указанной визиром. Визир (вертикальная линия) появляется в месте нажатия ЛК. Чтобы визир перемещался по меткам активной кривой, нужно удерживать клавишу **SHIFT**;
- если увеличение включено – временной интервал выделяемой области;




-  – показать панель инструментов. Если кнопка не нажата, панель инструментов не отображается;







-  – показать/скрыть легенду;

-  / (, , ) – переключатель источника данных:



 – текущие данные;





, ,  – архивные данные (в этом режиме переключателя)

тель индицирует следующие состояния:  – запрос архива на чтение;  – чтение архива;  – отображение извлеченных архивных данных);

   и    – кнопки для перехода по временной оси графика. Соответственно **к началу, на день назад, на час назад и на начало следующего часа, на начало следующего дня, к концу.**

Для смещения по осям тренда можно также использовать стандартные инструменты прокрутки.

Для перемещения визира по точкам активной кривой в пределах видимого диапазона можно использовать клавиши /.





Чтобы в реальном времени изменить масштаб по любой из осей, надо нажать ЛК в области тренда, а затем нажатием сочетания клавиш **CTRL+///** установить требуемый масштаб (при этом число разбиений оси настраивается автоматически).

На время переключения на другой экран параметры отображения трендов сохраняются.

Если в узле МРВ1 есть архивируемый канал **СН1**, который копируется по сети в узел МРВ2, и в канале **СН1_сору** установлен флаг архивирования в архив, которого в узле МРВ2 нет, то при выводе **СН1_сору** на архивный тренд запрашиваются данные из локального архива узла МРВ1.


Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия МРВ с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа.**

ГЭ 'Архивный тренд'

ГЭ **Архивный тренд**  представляет собой модификацию ГЭ 'Тренд', обеспечивающую графическое представление только архивных данных. Соответственно, индикатор режима архивного тренда имеет три положения ( – запрос архива на чтение;  – чтение архива;  – отображение извлеченных архивных данных).

Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия МРВ с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа.**

ГЭ 'Тренд ХУ'

ГЭ **Тренд ХУ**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ **Тренд ХУ** представляет собой модификацию ГЭ **Тренд** (см. **ГЭ 'Тренд'**) для графического представления функций $y(x)$, заданных параметрически ($x=f(p)$, $y=g(p)$).

Привязки к аргументам экрана, определяющим законы изменения координат, задаются при конфигурировании кривой (атрибуты **Привязка X** и **Привязка Y**) (см. также **Операции с аргументами в РГД**).





При конфигурировании ГЭ **Тренд ХУ** в ИС используются следующие атрибуты, специфичные относительно ГЭ **Тренд**:


- **Буфер** – количество последних по времени точек кривой, хранимых в памяти ($2 \cdot 10^6$, по умолчанию – 500);
- **Единицы времени** – единицы времени для атрибутов **Шаг1**, **Шаг2** и **Интервал выборки**;
- **Шаг1** – величина сдвига репера (T_{max}) по командам **Шаг1 назад** и **Шаг1 вперед**;
- **Шаг2** – величина сдвига репера (T_{max}) по командам **Шаг2 назад** и **Шаг2 вперед**;







T_{max} – значение времени, от которого отсчитывается интервал выборки.


- **Интервал выборки** – временной интервал, из которого выбираются данные для отображения на тренде (отсчитывается назад от T_{max}). Количество отображаемых точек из архива не превышает значение, заданное атрибутом **Буфер**.

При работе в реальном времени используются следующие инструменты ГЭ **Тренд ХУ**, специфичные относительно ГЭ **Тренд**:

 / (, , ) – переключатель источника данных:


 – текущие данные;

, ,  – архивные данные (в этом режиме переключатель индицирует следующие состояния:  – запрос архива на чтение;  – чтение архива;  – отображение извлеченных архивных данных);

 – меню, содержащее команды навигации по буферу/архиву:

- **Шаг2 назад** – сдвиг T_{max} на **Шаг2** назад;
- **Шаг1 назад** – сдвиг T_{max} на **Шаг1** назад;

- **Шаг1 вперед** – сдвиг T_{max} на **Шаг1** вперед;
- **Шаг2 вперед** – сдвиг T_{max} на **Шаг2** вперед;
- **В конец** – присвоение T_{max} значения максимальной метки времени из буфера/архива;
- **К метке времени** – присвоение T_{max} произвольного значения (задается в диалоге, который отображается при выполнении данной команды, – см. **ГЭ ‘Тренд’**).


Для вывода в легенду координат и времени точки кривой нужно подвести к этой точке курсор, который в этом случае принимает форму .



Атрибуты **ГЭ Тренд XY**, определяющие вид кривой и собственно тренда, аналогичны соответствующим атрибутам **ГЭ Тренд** (см. **ГЭ ‘Тренд’**).

ГЭ Тренд XY не поддерживает прокрутку по осям X и Y за пределы, заданные при конфигурировании тренда, поэтому эти пределы должны коррелировать соответственно с областью определения и областью допустимых значений функции.

Для корректной работы **ГЭ** каналы, задающие законы изменения координат, должны иметь одинаковый период пересчета и архивироваться в один и тот же архив.

ГЭ ‘Архивная гистограмма’

Данный оконный **ГЭ**  предназначен для отображения данных из SIAD в виде гистограмм.

Окно свойств **ГЭ** содержит вкладки **Осн. свойства** () и **Гистограммы** ()

Помимо типовых свойств (см. **Задание типовых свойств ГЭ**), на вкладке **Осн. свойства** задаются следующие параметры:

- **Отображать легенду** – если TRUE, в нижней части **ГЭ** отображается легенда. Легенда отображает имя/кодировку канала, цвет его гистограммы и численные значения;
- **Вид** – вид представления значений на гистограммах;
- **Тип** – диапазон шкалы значений:
 - **произвольный** – диапазон задается атрибутами **Минимум** и **Максимум** и не изменяется в реальном времени;
 - **абсолютный** – диапазон автоматически устанавливается от 0 до максимального отображаемого значения;
 - **относительный** – диапазон автоматически устанавлива-

ется от минимального отображаемого значения до максимального;


- **Число разбиений Y** – число разбиений оси значений;
- **Единицы времени** – единицы (с, мин, ч) для атрибута **Временной диапазон**;
- **Временной диапазон** – временной диапазон ГЭ (если не заданы привязки атрибутов **Левая граница** и **Правая граница**);
- **Левая граница** – аргумент, привязанный к этому атрибуту, задает левую временную границу ГЭ;
- **Правая граница** – аргумент, привязанный к этому атрибуту, задает правую временную границу ГЭ;
- **Обновление** – если этот атрибут привязан к аргументу, то при старте МРВ и при любом изменении значения аргумента архивные данные считываются в буфер;
- **Число меток времени** – число меток на оси времени (определяет также число отображаемых значений).

На вкладке **Гистограммы** конфигурируются гистограммы ГЭ. Для добавления гистограммы нужно выполнить команду **Добавить** из контекстного меню раздела **Гистограммы**, для перемещения в списке или удаления – соответствующую команду из контекстного меню раздела конфигурирования гистограммы (см. также **Операции с аргументами в РПД**).



Гистограмма конфигурируется с помощью следующих атрибутов:



- **Привязка** – аргумент экрана (должен быть привязан к архивируемому атрибуту канала);
- **Цвет** – цвет гистограммы;
- **Использовать кодировку** – если FALSE, в легенде отображается имя канала, если TRUE – кодировка;
- **Формат** – формат значения в легенде (см. **Формат Си вывода чисел**).

Если привязки атрибутов **Левая граница** и **Правая граница** не заданы, в реальном времени доступны следующие инструменты ГЭ:



 – переход к временной метке. При нажатии на эту кнопку появляется диалог задания даты и времени (см. **ГЭ 'Тренд'**). Заданное в диалоге значение устанавливается в качестве правой границы временной оси ГЭ;

 – команда/индикатор обновления буфера;

 /  – переход в начало/конец архива;

 /  – переход по времени назад/вперед на величину, заданную ат-


рибутом **Временной диапазон**;

 /  – переход назад/вперед на одно деление временной шкалы.


Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия МРВ с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа**.

Группа ГЭ 'Объекты'

ГЭ 'Объект'

С помощью ГЭ **Объект**  на графическом экране может быть размещен:

- ГО из библиотеки графических объектов (см. **Операции с графическими объектами**);
- ГО из библиотеки избранных элементов (см. **Библиотека избранных ГЭ и библиотека избранных eГЭ**).

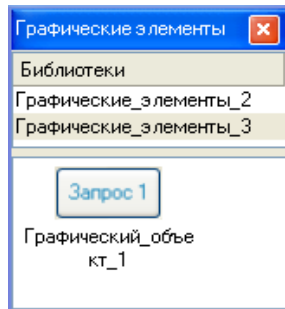
Вкладка **Осн. свойства**  окна свойств данного ГЭ, помимо инструментов задания типовых свойств ГЭ (см. **Задание типовых свойств ГЭ**), содержит инструменты привязки аргументов графического объекта к аргументам экрана.

Если ГО содержит скрываемые слои (см. **Задание параметров графического экрана**), то вкладка содержит также следующие атрибуты:

- **Показывать скрываемый слой** – место отображения скрываемого слоя ГО в реальном времени;
- **Скрываемый слой** – выбор скрываемого слоя ГО для отображения (скрываемый слой отображается в окне). Чтобы отобразить скрываемый слой в реальном времени, нужно нажать ЛК в области прямоугольника, окаймляющего ГЭ нижнего слоя ГО (нижний слой ГО отображается всегда). Нажатие ЛК на оконном ГЭ не приводит к отображению скрываемого слоя. С помощью контекстного меню нижнего слоя ГО можно выбрать скрываемый слой для отображения, а также закрыть окно (заголовок окна также содержит инструмент закрытия). Все подобные окна можно закрыть по команде профайлера. В случае выхода окна скрываемого слоя за правую границу экрана оно отображается слева, если **Показывать скрываемый слой = справа** или **внизу справа**. В случае выхода окна скрываемого слоя за нижнюю границу экрана оно отображается вверху, если **Показывать скрываемый слой = внизу** или **внизу справа**;
- **Заголовок** – информация, выводимая в заголовок окна, в котором отображается скрываемый слой:
 - **Нет** (заголовок окна не отображается)
 - **Имя слоя**
 - **Имя слоя и имя привязки**
 - **Имя слоя и кодировка привязки**
 - **Имя слоя и имя аргумента**

Для размещения ГЭ **Объект** на экране существуют следующие способы:

- способ 1 – стандартный (см. **Размещение ГЭ**); ресурс из библиотеки выбирается в навигаторе, который появляется на экране при выборе ГЭ **Объект** на панели инструментов **Графические элементы** (в навигаторе отображаются только библиотеки ГО слоя **Ресурсы**):



- способ 2 – метод drag-n-drop (см. **Типовые операции редактирования**) из дерева структуры проекта (как из слоя **Ресурсы**, так и из слоя **Библиотеки компонентов**), из показанного выше навигатора или из окна просмотра библиотеки избранных ГЭ;
- способ 3 – способ 2 с удержанием клавиш CTRL+SHIFT;
- способ 4 – способ 2 с удержанием клавиши SHIFT.

Если ГО размещен с помощью способа 1 или 2, его аргументы должны быть привязаны к аргументам экрана вручную.

Способ 3 предназначен для автопостроения аргументов экрана и/или автопривязки к ним аргументов размещаемого ГО. В этих процедурах флаг **Обрабатывать флаг РО как глобальный** (см. **Настройка параметров редактора аргументов**) не анализируется. Под автопостроением в данном контексте понимается создание аргумента экрана, идентичного аргументу ГО по всем параметрам. Под автопривязкой понимается автоматическая привязка аргумента ГО к одноименному аргументу экрана, а если аргумент ГО входит в группу – к аргументу экрана, имеющему имя аргумента ГО с младшим порядковым номером в группе (ср. **Автопостроение каналов по команде редактора аргументов**). При использовании способа 3 на экране появляется меню, содержащее следующие команды:

- **РО** – автопривязка аргументов ГО с флагом **РО** (принадлежность к группе анализируется); автопостроение по аргументам ГО без флага **РО** с дальнейшей автопривязкой (принадлежность к группе не анализируется);
- **Все** – автопривязка аргументов ГО (принадлежность к группе анализируется).

При размещении способом 4 аргументы объекта привязываются к одноименным аргументам экрана с анализом флага **РО**.



Во всех случаях при использовании способов 3 и 4:


- если аргумента экрана, необходимого для автопривязки, нет, он создается;
- аргументы экрана не создаются по аргументам ГО с установленным флагом **NP** (ср. **Флаги NP и PO**), и такие аргументы ГО автоматически не привязываются.

Особенности прозрачности ГЭ **Объект**:

- нет атрибута **Градиент**; элементы ГО имеют градиент, заданный для них при редактировании ГО;
- если прозрачность ГО статическая, она добавляется к прозрачности элементов ГО;
- в случае динамизации прозрачность ГО присваивается элементам.

ГЭ 'Объект в окне'


ГЭ **Объект в окне**  является аналогом ГЭ **Объект**  (см. ГЭ 'Объект').

Отличие заключается в том, что ГЭ **Объект в окне**  размещается на экране в отдельном окне, которое в реальном времени можно масштабировать (если **Сохранить размер** = FALSE) и перемещать по экрану (если разрешен заголовок окна, т.е. если **Показать заголовок** = TRUE). Кроме того, ГЭ **Объект в окне** нельзя разместить на экране методом drag-n-drop из дерева структуры проекта и из окна просмотра библиотеки избранных ГЭ.

Атрибут **Заголовок** может принимать следующие значения:

- **Вручную** – заголовок задается вручную (атрибут **Текст заголовка**);
- **Имя привязки** – в заголовке отображается имя привязки аргумента экрана, к которому привязан первый аргумент объекта;
- **Имя аргумента** – в заголовке отображается имя аргумента экрана, к которому привязан первый аргумент объекта.


ГЭ 'Ссылка на экран'

ГЭ **Ссылка на экран**  представляет собой окно, отображающее другой экран.

Данный элемент размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Для создания в шаблоне экрана **N** ссылки на экран **M** доста-

точно перетащить шаблон или канал вызова экрана **M** из навигатора проекта в шаблон экрана **N**, удерживая **CTRL**.

Отображаемые экраны задаются на вкладке  (**Осн. свойства**) окна **Свойства объекта**. Для добавления экрана в список (или его удаления) используется контекстное меню, вызываемое нажатием ПК мыши на строке **Экраны** (или, соответственно, **Экран**). При нажатии ЛК в поле **Значение** атрибута **Экран** открывается структура проекта, в которой из всех компонентов отображаются только шаблоны экранов и каналы вызова экранов. Для добавления экрана нужно выбрать его. Редактировать список можно с помощью перетаскивания экранов из навигатора проекта: чтобы добавить экран в список, нужно перетащить его на строку **Экраны**, чтобы заменить экран в списке – на строку **Экран**.

Во всех случаях выбор шаблона равнозначен выбору канала вызова этого шаблона с младшим ID.

Ссылка не может быть задана:


- на всплывающий экран;
- при размещении на всплывающем экране – на стартовый экран;
- на экран, на котором размещена.

Если атрибут **Сохранить размер** имеет значение **True**, то в реальном времени ГЭ **Ссылка на экран** принимает размеры отображаемого экрана.


Атрибут **Динамизировать** служит для управления переключением отображаемых экранов. Если задано несколько экранов для отображения, то значение привязанного аргумента определяет отображаемый экран. Для показа первого экрана из списка целая часть значения аргумента должна равняться 0, для второго – 1, и т.д. Если аргумент не указывает ни на один экран из списка, ГЭ не отображается.

Группа ГЭ 'Таблицы'

ГЭ 'Переключатель каналов'

ГЭ **Переключатель каналов**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ **Переключатель каналов** представляет собой таблицу, в которую могут быть выведены объекты базы каналов (группы с установленным флагом **Загрузить**) или каналы (все или указанного объекта базы каналов), и служит для перепривязки аргументов экрана.

Окно свойств ГЭ имеет вкладку  (**Осн. Свойства**).

Поле **Привязка** задает аргумент, выбираемый с помощью стандартного диалога (см. **Табличный редактор аргументов**).

Для правильной работы аргумент должен быть изначально привязан к какому-либо каналу.

С помощью списка **Класс каналов** задается фильтр вывода по классу канала:

- **ALL** – в таблицу выводится вся база каналов;
- **CALL** – каналы класса **CALL**;
- **HEX_16** – каналы класса **HEX16**;
- **HEX_32** – каналы класса **HEX32**;
- **FLOAT** – каналы класса **Float** без обработки;
- **FLOAT_M** – каналы класса **Float** с обработкой;
- **FLOAT_64** – каналы класса **Double Float**;
- ... – зарезервировано;
- **M-RESOURCE** – каналы класса **M-Ресурс**;
- ... – зарезервировано;
- **D-RESOURCE** – каналы класса **D-Ресурс**;
- **USER** – каналы класса **Пользователь**;
- **EVENT** – каналы класса **Событие**;
- **TIME** – каналы класса **Time**;
- **EQUIPMENT** – каналы класса **Единица оборудования**;
- **EMPLOYEE** – каналы класса **Персонал**.

С помощью списка **Условие выборки каналов** задается дополнительный фильтр вывода (см. **Атрибуты каналов, отображаемые про-**

файлером):

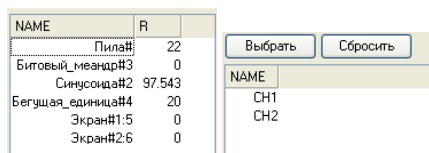
- **нет** – все каналы
- **R==0 ... R==15** – каналы, реальное значение (0, **R**) которых равно соответственно 0...15
- **FA==yes** – каналы с аппаратной недостоверностью (атрибут (4, **I**) равен 1)
- **SC==off** – каналы, находящиеся в состоянии **выключен** (атрибут (3, **C**) равен 1)
- **Interval==0 ... Interval==7** – каналы, атрибут (7, **P**) которых равен соответственно 0...7
- **STS==0 ... STS==7** – каналы, атрибут (133,**STS**) которых равен соответственно 0...7
- **Arg** – все каналы, связанные с аргументами экрана
- **R(Arg)==0 ... R(Arg)==15** – каналы, связанные с аргументами экрана, реальное значение (0, **R**) которых равно соответственно 0...15
- **FA(Arg)==yes** – каналы, связанные с аргументами экрана, с аппаратной недостоверностью (атрибут (4, **I**) равен 1)
- **SC(Arg)==off** – каналы, связанные с аргументами экрана, находящиеся в состоянии **выключен** (атрибут (3, **C**) равен 1)
- **Interval(Arg)==0 ... Interval(Arg)==7** – каналы, связанные с аргументами экрана, атрибут (7, **P**) которых равен соответственно 0...7
- **STS(Arg)==0 ... STS(Arg)==7** – каналы, связанные с аргументами экрана, атрибут (133,**STS**) которых равен соответственно 0...7
- **OBJ** – все объекты базы каналов (группы с установленным флагом **Загрузить**)
- опции вида **OBJ with <substr>** – объекты, в имени которых содержится указанная подстрока.

Если в первом списке выбран класс канала, а во втором – одна из опций, относящихся к выводу на объектном уровне, то в таблицу будут выведены объекты, удовлетворяющие дополнительному фильтру и содержащие хотя бы один канал заданного класса.

Период определяет скорость обновления информации в таблице в тактах пересчета экрана.

Для добавления в таблицу колонки с атрибутом канала (или ее удаления) используется контекстное меню, вызываемое нажатием ПК мыши на строке **Колонки** (или, соответственно, **Атрибут**).

При запуске в реальном времени в таблицу выводится информация, удовлетворяющая начальным настройкам ГЭ.




Если ГЭ не привязан к аргументу, кнопки **Выбрать** и **Сбросить** не отображаются.

Для привязки нужного канала к аргументу, указанному в настройках ГЭ, необходимо выделить канал в столбце **NAME** и нажать кнопку **Выбрать**. При нажатии кнопки **Сбросить** в выбранный канал посылается -1.

Если переключатель каналов (**Класс каналов=CALL**, **Условие выборки каналов=STS(Arg)==0** или **STS(Arg)==1**) привязан к аргументу **HANDLE**, связанному с атрибутом (55, **relink**) некоторого канала **ch**, то возвращается:

- если **ch** – канал вызова экрана, – то, что скопировано в экран;
- если **ch** – **CALL.ChGroupReq**, – то, что он копирует в экран.

ГЭ 'События'

ГЭ **События**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ **События** предназначен для отображения различных ситуаций, фиксируемых каналами класса **Событие** (см. **Канал класса СОБЫТИЕ**), их просмотра в реальном времени и квитиования.

Окно свойств ГЭ содержит вкладку **Осн. свойства**, на которой конфигурируются общие свойства ГЭ, и вкладку **Привязки**, на которой задаются отображаемые параметры.

Раздел **Цвета** содержит инструменты для задания цвета строки при отсутствии, появлении и исчезновении события.

Атрибут **Мигать при возникновении события** устанавливает мигающую индикацию при наступлении события. Атрибут **Мигать при исчезновении события** устанавливает мигающую индикацию при исчезновении события.

Функции мигания поддерживаются только для событий первого типа (см. **Канал класса СОБЫТИЕ**).

Раздел **Заголовки** содержит стандартные инструменты для настройки заголовка ГЭ.

Атрибуты **Показать имена**, **Показать кодировку** и **Показать ком-**

ментарий устанавливает отображение в таблице соответствующих атрибутов каналов.

Вкладка **Привязки** имеет следующий вид:


| Свойство | Значение |
|------------------|----------|
| Привязки | |
| Привязка ARG_000 | |
| Привязка ARG_001 | |

Для добавления привязки (или ее удаления) используется контекстное меню, вызываемое нажатием ПК мыши на строке **Привязки** (или, соответственно, **Привязка**).

Для правильной работы ГЭ каждый добавляемый аргумент должен иметь тип **INPUT** и быть привязан к атрибуту **Реальное значение** канала класса **Событие**; все остальные атрибуты извлекаются автоматически.

Если аргумент привязан к произвольному атрибуту канала произвольного класса, то при изменении такой атрибут считается как строка и отображается в столбце **Статус** ГЭ.

Если для привязок аргументы не заданы, они задаются автоматически в реальном времени.

При запуске в реальном времени в таблицу выводится список каналов класса **Событие**, привязанных к аргументам на вкладке **Привязки**. Если для канала задано положительное значение размера стека аварий (см. **Канал класса СОБЫТИЕ**), то, раскрыв список нажатием ЛК в поле , можно просмотреть историю событий.

| События | | | | | | Квитированные | Неактуальные |
|-----------|-----------|-------------------------|-------------------------|---------------------|------------|---------------|--------------|
| Имя | Кодировка | Время возникновения | Время исчезновения | Время квитирования | Статус | Комментарий | |
| Событие#1 | TM2 | ... | ... | ... | ... | ... | |
| Событие#3 | TM2 | 20.05.2008 10:52:44.852 | ... | ... | E_Оп | ... | |
| ... | ... | 20.05.2008 10:52:35.501 | 20.05.2008 10:52:42.102 | 20.05.2008 10:52:38 | E_Оп+АСК | ... | |
| ... | ... | 20.05.2008 10:52:31.651 | 20.05.2008 10:52:33.851 | ... | E_Оп+члАСК | ... | |
| ... | ... | 20.05.2008 10:52:27.801 | 20.05.2008 10:52:29.451 | ... | E_Оп+члАСК | ... | |
| Событие#4 | TM2 | 20.05.2008 10:52:53.701 | 20.05.2008 10:52:55.902 | ... | E_Оп+члАСК | ... | |

В столбцах таблицы выводятся следующие параметры:

- **Имя** – имя канала класса **Событие**;
- **Кодировка** – кодировка канала;
- **Время возникновения** – дата и время возникновения события (с точностью до миллисекунд);
- **Время исчезновения** – дата и время исчезновения события (с точностью до миллисекунд);
- **Время квитирования** – дата и время квитирования события (с точностью до секунд);

- **Статус** – статус события (см. **Канал класса СОБЫТИЕ**). Для каждого статуса предусмотрена иконка:

- 0 (на вкладке **Компоненты** панели MPB – ...) – 
- 1 (**E_On**) – 
- 2 (**E_Off+ACK**) – 
- 3 (**E_On+ACK**) – 
- 4 (**E_Off+unACK**) – 
- 5 (**E_On+wACK**) – 
- 6 (**E_Off+wACK**) – 
- 7 (**E_On_On**) – 

Строки описания статуса можно задать при помощи соответствующего словаря или в файле *.cnf.

- **Комментарий** – комментарий к каналу.

Для квитирования события надо нажать **Ctrl+ЛК** в соответствующей строке ГЭ.

Методом drag-and-drop можно изменять ширину столбцов, а также менять столбцы местами. Для изменения порядка сортировки нужно нажать ЛК в заголовке соответствующего столбца.

В заголовке ГЭ размещены следующие кнопки:

- **Квитированные** – скрыть/показать квитированные события;
- **Неактуальные** – скрыть/показать неактуальные события (статус отличен от 1, 3, 5 и 7); этот переключатель имеет более высокий приоритет.

При привязке к атрибуту (254, **RST**) канала произвольного класса (см. **Атрибуты каналов, отображаемые профайлером**) ГЭ отображает соответствующую информацию и обеспечивает квитирование:

- атрибут (254, **RST**) считывается как строка и отображается в столбце **Статус**;
- время исчезновения и время квитирования – отсутствуют;
- время возникновения равно значению атрибута (45, **T**) привязанного канала (за исключением канала **FLOAT**, в котором вычисляется интервал, – для такого канала время возникновения равно времени изменения интервала). Если строковое значение атрибута (47, **iDstr**) привязанного канала отлично от «...», время передается без миллисекунд;
- при нажатии **Ctrl+ЛК** (квитирование из ГЭ) в привязанном канале обнуляются **b14** и **b15** и квитируется последнее сообщение канала


в ОТ (если генерация сообщений в ОТ для канала сконфигурирована). Замечание: т.к. кэширование сообщения ОТ с помощью ГЭ **Строка ОТ** или **ОТ узла** не приводит к сбросу **b14** и **b15**, подобная операция никак не отображается в ГЭ **События**.

Особенности использования ГЭ в консолях:


- для запроса временных меток изменений статуса событий нужно установить флаг **Запрос времени значения** в канале вызова экрана.

ГЭ 'Архивная таблица'

Оконный ГЭ **Архивная таблица**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Вкладка **Осн. свойства**  окна свойств ГЭ содержит следующие специфические атрибуты:


- атрибуты разделов **Диапазон выборки** и **Шаг выборки** и атрибут **Обработка** – параметры выборки (идентичны тем же атрибутам таблицы архивных значений документа – см. **Задание параметров выборки** в разделе **Конфигурирование таблицы архивных значений**);
- **Обновление** – если этот атрибут привязан к аргументу, то при старте МРВ и при любом изменении значения аргумента архивные данные считываются в буфер.

На второй вкладке  задаются столбцы таблицы (в столбце отображаются архивные данные одного канала). Назначение столбца **Время** фиксировано – в нем отображаются времена архивных значений.

Чтобы добавить столбец в таблицу, нужно выполнить команду **Добавить столбец** из контекстного меню раздела **Столбцы**. Для удаления или изменения порядка столбцов нужно использовать команды **Вверх**, **Вниз** и **Удалить** из контекстного меню раздела конфигурирования столбца.

Раздел конфигурирования столбца содержит следующие специфические атрибуты:

- **Привязка** – привязка к аргументу (к аргументу должен быть привязан архивируемый атрибут канала);
- **Видимость** – если TRUE, столбец отображается в таблице в реальном времени;
- **Использовать кодировку** – если TRUE, в реальном времени в заголовке столбца отображается кодировка канала; если FALSE – имя канала;
- **Формат** – формат значения (см. **Формат Си вывода чисел**).









Для обновления буфера вручную нужно нажать кнопку  (отображается на ГЭ в реальном времени).

В процессе обновления буфера на месте кнопки индицируется процент выполнения.

При нажатии ПК на ГЭ на экране появляется диалог управления видимостью столбцов.

Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия МРВ с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа**.

ГЭ 'Архивная таблица 2'

Аналогично ГЭ 'Архивная таблица', данный ГЭ  предназначен для отображения данных из SIAD, однако имеет инструменты перемещения по буферу (кнопки       ), аналогичные соответствующим командам ГЭ 'Тренд ХУ'.

Атрибуты **Левая граница** и **Правая граница** аналогичны одноименным атрибутам ГЭ 'Тренд'.

Специфические атрибуты ГЭ:

- **Отображать дату** – если FALSE, отображаемые временные значения не содержат даты;
- **Отображать миллисекунды** – если FALSE, отображаемые временные значения не содержат миллисекунд.

Столбцы ГЭ конфигурируются аналогично столбцам ГЭ 'Архивная таблица'.

Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия МРВ с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа**.

ГЭ 'База данных'

Оконный ГЭ **База данных**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ предназначен для отображения данных, извлеченных из БД, а также для записи значений (из полученной выборки) в аргументы/каналы:



- подключение к БД и инструкция SELECT конфигурируются в ка-

нале CALL.SQLQuery, который должен быть привязан к аргументу экрана (пусть этот аргумент имеет имя **scr_arg**). В канале CALL.SQLQuery создаются аргументы (OUTPUT), которые используются как подстановки в инструкции SELECT (пусть эти аргументы имеют имена **DB_arg<n>**; см. также **Подстановки в SQL-запросе**).

Канал CALL.SQLQuery должен содержать только один SQL-запрос (номер запроса не имеет значения).

- ГЭ **База данных** должен быть привязан к **scr_arg** (атрибут **Привязка**).

В реальном времени доступны следующие инструменты ГЭ:

- кнопка **Запрос** () – извлечь данные из БД (SQL-запрос выполняется, но в **DB_arg<n>** данные не записываются). При успешном выполнении запроса ГЭ отображает таблицу-выборку (столбцы имеют имена **<имя таблицы БД>.<имя столбца таблицы БД>**). Если атрибут **В виде дерева** = TRUE, данные отображаются в виде дерева (первый столбец инструкции SELECT располагается на верхнем уровне, а остальные столбцы отображаются как дочерние первого);
- кнопка **Стоп** () – прервать выполнение запроса;
- кнопка **Запись** – записать значения из выбранной строки в **DB_arg<n>**.

В строке ГЭ вверху справа отображается текущее состояние взаимодействия с БД.

Группа ГЭ 'ActiveX'

В эту группу входит единственный ГЭ **Компонент ActiveX** .

При нажатии ЛК на этой кнопке на экране появляется список ActiveX-компонентов, зарегистрированных в операционной системе.

ГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ**). Разработка и использование компонентов ActiveX описаны в разделе **ActiveX в TRACE MODE 6**.

Группа ГЭ ‘Свободные формы’



В эту группу входит единственный ГЭ **Свободные формы** .

В рамках графической базы узла можно создавать графические элементы, не имеющие координат и не выводимые на экранах. Эти ГЭ используются для управления с помощью заданного канала видимостью других ГЭ, привязкой ГЭ к другим каналам.

ГЭ **Свободные формы** сохранен в TRACE MODE 6 только для совместимости с версией 5. В проекте TRACE MODE 6 этот ГЭ использовать не следует, после конвертирования проекта TM5 в TM6 его нужно удалить.



Для того, чтобы воспользоваться ГЭ **Свободные формы**, необходимо установить флаг **Использовать устаревшие функции** в разделе **РПД** окна **Настройки** (см. **Задание параметров РПД**).

При выборе ГЭ **Свободные формы** на экран выводится окно настроек.

Первая вкладка  **Отображение** этого окна используется для управления видимостью других ГЭ. Вкладка  **Перепривязка** предназначена для управления привязкой ГЭ к различным аргументам.




Управление отображением ГЭ

Эти ГЭ позволяют управлять видимостью других ГЭ на текущем экране по значению аргумента. При его равенстве 0 управляемые ГЭ видимы, в противном случае – невидимы.

Для добавления/удаления свободных форм предусмотрены типовые средства  и  (см. **Типовые средства редактирования**). После создания в окне появляется следующая строка:

| Имя | Источник данных | Графические элементы |
|-----|-----------------|----------------------|
| Ф1 | ARG_000 | Всего = 1 |
| Ф2 | ARG_001 | Всего = 2 |
| Ф3 | ARG_000 | Всего = 0 |



В поле **Имя** задается название формы. **Источник данных** задает управляющий аргумент – при двойном нажатии ЛК открывается стандартное окно выбора аргумента (см. **Табличный редактор аргументов**).

Для добавления/удаления управляемых ГЭ нужно дважды нажать в поле **Графические элементы**, при этом оно примет вид . Теперь с помощью мыши нужно выбрать на экране нужные ГЭ. Для добавления/удаления сразу нескольких ГЭ нужно последовательно выделять их, удерживая клавишу **Ctrl**. По окончании выбора для сохранения выделения нужно нажать кнопку . Для очистки списка выбранных ГЭ предусмотрена кнопка .

После окончания редактирования поле **Графические элементы** примет вид **Всего = N**. Число **N** означает количество выбранных ГЭ.




Перепривязка ГЭ

Эти ГЭ позволяют перепривязать другие элементы к аргументам в зависимости от значения управляющего аргумента. ГЭ **Перепривязка** связан с экраном, при редактировании которого был создан. Аргументы, на которые переключается управляемый ГЭ, образуют список привязок. Значение управляющего аргумента (это аргумент, с которым связан ГЭ **Перепривязка**), указывает номер строки в этом списке, начиная с 0, определяя аргумент для управляемого ГЭ.

Для добавления/удаления свободных форм предусмотрены типовые средства  и  (см. **Типовые средства редактирования**). После создания в окне появляется следующая строка:

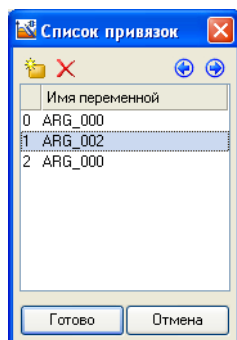
| Имя | Источник данных | Графические элементы | Список привязок |
|-----------|-----------------|----------------------|-----------------|
| Новый ... | | Всего = 0 | Всего = 0 |

В поле **Имя** задается название формы. **Источник данных** задает управляющий аргумент – при двойном нажатии ЛК открывается стандартное окно выбора аргумента (см. **Табличный редактор аргументов**).



Для добавления/удаления управляемых ГЭ нужно дважды нажать в поле **Графические элементы**, при этом оно примет вид . Теперь с помощью мыши нужно выбрать на экране нужные ГЭ. Для добавления/удаления сразу нескольких ГЭ нужно последовательно выделять их, удерживая клавишу **Ctrl**. По окончании выбора для сохранения выделения нужно нажать кнопку . Для очистки списка выбранных ГЭ предусмотрена кнопка .

После окончания редактирования поле **Графические элементы** примет вид **Всего = N**. Число **N** означает количество выбранных ГЭ.

Для создания списка перепривязываемых аргументов нужно дважды нажать в поле **Список привязок**, при этом откроется одноименной окно:




Число перед именем аргумента показывает номер строки (начиная с 0). При равенстве управляющего аргумента данному числу происходит привязка ГЭ к соответствующему аргументу списка.

Для добавления/удаления аргументов в списке предусмотрены типовые средства  и  (см. **Типовые средства редактирования**). При добавлении аргумента открывается стандартное окно выбора аргументов (см. **Табличный редактор аргументов**).

Группа ГЭ 'Приборы'

ГЭ 'Ползунок'

ГЭ **Ползунок**  используется для задания и/или индикации значения привязанного аргумента, а также для индикации выполнения заданного условия.

Атрибуты ГЭ **Ползунок** конфигурируются на вкладке **Осн. свойства** окна свойств данного графического элемента. Помимо типовых свойств (цвет и толщина контура, фон и т.п.), конфигурирование которых описано в разделе **Задание типовых свойств ГЭ**, ползунок имеет значительное количество специфических атрибутов.

Атрибуты **Отображаемая величина** и **Задаваемая величина** определяют функциональное назначение ГЭ.

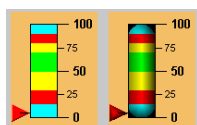
Чтобы ГЭ индицировал значение привязанного аргумента или выполнение некоторого условия, нужно динамизировать его атрибут **Отображаемая величина** (см. **Динамизация атрибута ГЭ**).




Атрибут **Отображаемая величина** автоматически привязывается к основной привязке (если она задана).

При раскрытии списка в разделе конфигурирования атрибута **Задаваемая величина** появляется настройка выбора аргумента, чье значение ползунок будет задавать при работе в реальном времени.

Атрибут **Положение ползунка** задает положение ползунка относительно шкалы и соответствующее расположение штрихов и чисел на шкале.

В разделе **3D-эффекты** при установке значения **True** цветовая полоса и ползунок принимают объемный вид. **Края полосы** могут быть заданы как **Квадратные** или **Круглые**:



Для отображения ползунка на шкале надо установить атрибуту **Ползунок** значение **True**. Форма ползунка (**Квадрат** , **Треугольник** , **Домик** ) , его цвет и размер (в пикселях) определяются соответствующими атрибутами. **Длина риски (%)** на ползунке задается в процентах относительно размера ползунка.

Раздел **Полоса** содержит следующие атрибуты, определяющие парамет-

ры цветовой полосы шкалы:

| | |
|-------------------------|---|
| Полоса | True |
| Ширина | 10 |
| Верхний предел шкалы | 100 |
| + HL | 90 |
| + HA | 80 |
| + HW | 70 |
| + LW | 50 |
| + LA | 30 |
| + LL | 5 |
| Нижний предел шкалы | 0 |
| Цвет (HW, LW) |  |
| Цвет (HA, HW), (LW, LA) |  |
| Цвет (HL, HA), (LA, LL) |  |
| Цвет >HL, <LL |  |

Для отображения цветовой полосы шкалы нужно установить атрибуту **Полоса** значение **True** (по умолчанию установлено).

Ширина цветовой полосы задается в пикселях.


Цветовая полоса используется для цветовой кодировки интервалов значений отображаемой/задаваемой величины. По аналогии с каналом, интервалы определяются при помощи задания 6 границ (см. **Границы и интервалы канала FLOAT** и **Канал класса DOUBLE FLOAT**).

Для различных интервалов можно задать свои цвета на полосе.

Если для ГЭ задана основная привязка к значению канала с корректно заданными границами, границы полосы автоматически устанавливаются в соответствии с границами канала.

Верхний предел шкалы и **Нижний предел шкалы** задаются как целые числа, при этом верхний предел может быть как больше, так и меньше нижнего.

Раздел **Шкала** содержит параметры разбиения и подписей шкалы.

| | |
|---------------------|---|
| Шкала | True |
| Ширина | 1 |
| Цвет текста |  |
| + Уровень 1 | |
| Использовать | True |
| Число делений | 5 |
| Длина штриха | 12 |
| Показывать значения | True |
| Шрифт | Arial,10,жирный |
| Десятичные знаки | 2 |
| + Уровень 2 | |
| + Уровень 3 | |

Атрибуты **Ширина** (этот параметр задается в пикселях) и **Цвет текста**

определяют соответственно ширину и цвет штрихов и текста подписей всех уровней разбиения шкалы. Атрибут **Ширина** определяет одновременно толщину контура цветовой полосы шкалы.

Для шкалы могут быть заданы 3 уровня разбиения. Для каждого уровня (принадлежность к уровню указывает число в наименовании раздела) могут быть заданы следующие параметры:

- **Использовать** – при установке этому атрибуту значения **True** разбиение данного уровня используется, соответствующие штрихи отображаются на шкале.
- **Число делений** – число делений данного уровня.
- **Длина штриха** – длина штриха данного уровня (в пикселях).
- **Показать значения** – при установке этому атрибуту значения **True** отображаются численные значения, соответствующие штрихам данного уровня.
- **Шрифт** – шрифт, используемый для отображения численных значений данного уровня.
- **Десятичные знаки** – количество десятичных знаков в численных значениях данного уровня.


Если атрибуту **Использовать** присвоено значение **False**, остальные атрибуты для этого разбиения игнорируются.

При конфигурировании этих атрибутов нужно учитывать приоритет уровней разбиения:

- Атрибут **Число делений N** задает число делений, на которое разбивается каждое деление ближайшего более высокого уровня (**N-1** или **N-2**) при его использовании. Если ни один из более высоких уровней разбиения не используется, атрибут **Число делений N** задает число делений, на которое разбивается весь диапазон шкалы.

Разделы **Фон** и **Рамка** с помощью стандартных инструментов позволяют редактировать внешний вид ГЭ.

ГЭ 'Стрелочный прибор'

ГЭ **Стрелочный прибор**  предназначен для отображения текущего значения канала. Цветовая полоса этого ГЭ отображает интервалы значений канала (см. **Границы и интервалы канала FLOAT** и **Канал класса DOUBLE FLOAT**)

Помимо типовых свойств (цвет и толщина контура, фон и т.п.), конфигурирование которых описано в разделе **Задание типовых свойств ГЭ**, стрелочный прибор имеет значительное количество специфических атрибутов.

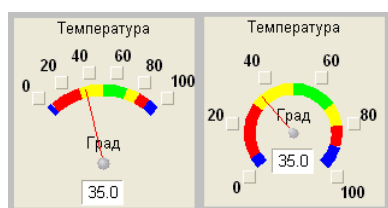
Динамизируемый атрибут **Отображаемая величина** задает значение

для отображения на приборе.

Атрибут **Отображаемая величина** автоматически привязывается к основной привязке (если она задана).

Атрибут **Положение** задает ориентацию стрелки прибора относительно шкалы и соответствующее расположение штрихов и чисел на шкале.

Угол разворота шкалы может принимать значения от 45 до 135 градусов:



Раздел **Заголовок** содержит стандартные атрибуты для настройки текста заголовка, его цвета и шрифта.

Разделы **Полоса** и **Шкала** имеют атрибуты, аналогичные атрибутам ГЭ **Ползунок** (см. ГЭ 'Ползунок').

Если для стрелочного прибора задана основная привязка к значению канала с корректно заданными границами, границы полосы автоматически устанавливаются в соответствии с границами канала.

В разделе **Указатель** можно установить параметры стрелки, такие как **Толщина**, **Цвет** и **Цвет основания**. Толщина стрелки задается в пикселях.


Настройки раздела **Единицы** определяют параметры выводимой единицы измерения.

При помощи раздела **Индикатор** можно расположить на ГЭ индикатор текущего значения. Для него можно установить свойства **Цвет**, **Шрифт**, **Десятичные знаки**.

Разделы **Фон** и **Рамка** с помощью стандартных инструментов позволяют редактировать внешний вид ГЭ.

Группа ГЭ ‘**Диаграммы**’

ГЭ ‘**Круговая диаграмма**’

ГЭ **Круговая диаграмма**  отображает абсолютный или относительный вклад выбранных аргументов в их общую сумму в виде секторов на круговой диаграмме. Если диаграмма отображает два и более аргументов, их значения должны быть неотрицательными.

Для размещения ГЭ используется стандартная процедура – см. **Размещение ГЭ**.

Окно свойств круговой диаграммы содержит вкладки **Осн. Свойства** и **Секторы**.

Помимо типовых свойств (цвет и толщина контура, фон и т.п.), конфигурирование которых описано в разделе **Задание типовых свойств ГЭ**, круговая диаграмма имеет значительное количество специфических атрибутов.

Параметры заголовка и подписи можно изменить с помощью разделов **Заголовок** и **Подпись**.

Для отображения **Легенды** рядом с диаграммой нужно в одноименном разделе установить значение **True**. Ее расположение (**Слева, Справа, Сверху, Снизу**) определяется атрибутом **Расположение**, а шрифт – **Шрифт**.

Атрибут **Десятичные знаки** задает количество десятичных знаков отображаемых в легенде.

Атрибут **Показывать описания** позволяет выводить рядом с секторами подписи с их именами и текущее значение привязанного аргумента.

Атрибут **Толщина диаграммы** задает толщину диаграммы в пикселях.

Начальный угол задает начало отсчета углов на диаграмме (относительно перпендикуляра к правой стороне экрана, 0-360 градусов, угол отсчитывается по часовой стрелке).

Угол наклона определяет наклон диаграммы к плоскости экрана (0-90 градусов).

Атрибут **Тип диаграммы** позволяет выбрать между **относительным** и **абсолютным** отображением данных. В первом случае диаграмма отображает относительный вклад аргументов в их общую сумму (в этом режиме игнорируются атрибуты **МИН** и **МАКС**).

При выборе **абсолютного** типа отображения доступны для задания параметры **МИН** и **МАКС**. При этом возможны 2 варианта диаграммы.

- **Вариант 1:** $\text{МИН}=0$, **МАКС** больше или равно максимально возможной сумме аргументов. В этом случае диаграмма отображает абсолютный вклад аргументов в их общую сумму.
- **Вариант 2:** $\text{МАКС} > \text{МИН} > 0$. Этот вариант предназначен для решения специальных задач отображения. Примером такой задачи может служить отображение в заданном диапазоне уровней несмещающихся жидкостей в емкости, если в аргументы передаются толщины слоев жидкостей. Диапазон отображаемых уровней задается параметрами **МИН** и **МАКС**.

Разделы **Фон** и **Рамка** с помощью стандартных инструментов позволяют редактировать внешний вид ГЭ.

На вкладке **Секторы** расположен список секторов диаграммы.


Секторы располагаются на диаграмме один за другим по часовой стрелке в соответствии с их позицией в списке (угол первого сектора отсчитывается от начала отсчета, угол второго – от конца первого и т.д.). Сектор, для которого привязка к аргументу не задана, считается привязанным к 0.

Для добавления (удаления) сектора используется команда контекстного меню раздела **Секторы** (раздела **Сектор**).


Для редактирования свойств сектора предусмотрены следующие инструменты:

- **Имя**– это поле предназначено для задания имени сектора. Для перехода к редактированию имени надо дважды нажать ЛК в данном поле;
- **Цвет**– задание цвета сектора; при нажатии ЛК в этом поле на экране появляется стандартный диалог выбора цвета;
- **Стиль**– задание стиля заливки сектора; при нажатии ЛК в этом поле в него выводится выпадающий список стилей заливки;
- **Привязка** – задание привязки сектора к аргументу экрана. При нажатии ЛК в данном поле на экране появляется стандартный диалог выбора аргумента.

ГЭ ‘Гистограмма’

ГЭ **Гистограмма**  отображает значения привязанных аргументов экрана в виде столбцов.

ГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Окно свойств гистограммы содержит вкладки **Осн. свойства**  и **Столбцы**:

На вкладке **Осн. свойства** конфигурируются следующие специфические атрибуты гистограммы (см. также **Задание типовых свойств ГЭ**):

- в разделе **Заголовок** задаются типовые параметры верхнего колонтитула ГЭ;
- в разделе **Подпись** задаются типовые параметры нижнего колонтитула ГЭ;
- в разделе **Легенда** задаются типовые параметры легенды. В легенду выводятся названия и цвета столбцов и текущие значения соответствующих аргументов;
Число десятичных знаков значений, отображаемых в легенде, задается атрибутом **Дробная часть** этого раздела;
- атрибут **Показывать описания** позволяет выводить в верхней части столбца текущее значение привязанного аргумента;
В разделе **Показывать описания** задаются типовые параметры шрифта и число десятичных знаков отображаемых значений;
- **Тип диаграммы** – если для этого атрибута задано значение **Абсолютный**, пределы шкалы устанавливаются атрибутами **Минимум** и **Максимум** и не изменяются при работе в реальном времени.
Если для атрибута **Тип диаграммы** задано значение **Относительный**, пределы шкалы изменяются в реальном времени. Нижний предел устанавливается по минимальному текущему значению среди всех отображаемых аргументов, верхний предел – по максимальному.
- **Толщина** – толщина столбцов в пикселях по оси Z (ось Z направлена вглубь ГЭ);
- **Количество делений шкалы** – этот атрибут задает число разбиений шкалы;
- **Цвет диаграммы** – цвет поверхностей YZ и XY шкалы;
- **Цвет основания** – цвет поверхности XZ шкалы;
- в разделе **Рамка** задаются типовые атрибуты контура ГЭ (цвет, толщина и стиль), а также вид контура (атрибут **Тип**) – стандартный, выпуклый или утопленный.


На вкладке **Столбцы** задаются столбцы, отображаемые на гистограмме.

Для добавления столбца нужно выполнить команду **Добавить столбец** из контекстного меню раздела **Столбцы**, для удаления столбца – команду **Удалить** из контекстного меню раздела **Привязка**.

В разделе **Привязка** задаются цвет и стиль заливки столбца, а также отображаемое в легенде название (атрибут **Название**) и привязка к аргументу экрана (атрибут **Привязка**).

Группа ГЭ 'Дата и время'

ГЭ 'Дата и время'

ГЭ **Дата и время**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

ГЭ **Дата и время** предназначен для отображения значения аргумента с временным или 4-байтовым целочисленным типом данных, а также для изменения значения аргумента с временным типом данных.

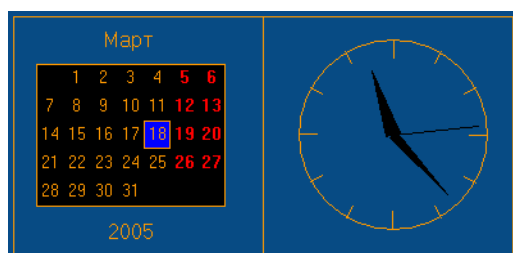
Наряду с типовыми параметрами (см. **Задание типовых свойств ГЭ**), ГЭ **Дата и время** имеет ряд специфических атрибутов.

Атрибут **Вид** определяет вид ГЭ – **базовый** или **расширенный** (если **Показывать** = **Интервал**, ГЭ имеет базовый вид).

Базовый вид:



Расширенный вид:



Тип привязки определяет источник формирования отображаемого значения. Из меню выбирается отображение текущего значения даты и/или времени (**Текущие дата/время**) или отображение значения аргумента (**Привязка к каналу**). Аргумент задается с помощью атрибута **Привязка**.

Атрибут **Только чтение** при установке ему значения **False** позволяет управлять значением привязанного аргумента. При работе в реальном времени по нажатию ЛК на ГЭ открывается окно ввода соответственно даты и времени или интервала.

При установке атрибуту **Тип привязки** значения **Текущие дата/время** разрывается связь с аргументом (если привязка была ранее задана), а атрибуту **Только чтение** устанавливается значение **True**.

Тип отображения устанавливается при помощи атрибута **Показывать**:

- **Дата** – только дата в формате **%d %B %Y** (см. **Формат Си вы-**


вода даты и времени).

- **Время** – только время в формате %H:%M:%S.
- **Дата и время** – дата и время в формате %d %B %Y %H:%M:%S.
- **Интервал** – значение аргумента (тип данных аргумента должен быть временным) в формате h...hH:MM:SS.

Атрибут **Отображать как** позволяет выводить значения даты и времени в текущем региональном часовом поясе (**Дата/время региональное**) или по Гринвичу (**Дата/время по Гринвичу**).

Группа ГЭ 'Отчет тревог'

ГЭ 'Строка ОТ'

ГЭ **Строка ОТ**  предназначен для отображения последнего по времени сообщения отчета тревог, удовлетворяющего заданным при конфигурировании ГЭ условиям, и его квитиования (см. **Формат строки ОТ**).

Окно свойств ГЭ содержит единственную вкладку **Осн. свойства** .


Наряду с типовыми параметрами (см. **Задание типовых свойств ГЭ**), на вкладке конфигурируются специфические атрибуты:

- **Фильтр по имени канала** – этот атрибут задает фильтрацию сообщений ОТ по имени канала. Значение атрибута задается как строковое выражение, которое может содержать стандартные знаки замены символов. Например, для вывода сообщения по любому из каналов с именами вида
`<строка1><один произвольный символ><строка2><несколько произвольных символов>`
можно задать фильтр в виде
`<строка1>?<строка2>*`
- **Фильтр по категории сообщения** – этот атрибут задает фильтрацию сообщений ОТ по категории. Значение атрибута выбирается из списка категорий (см. также **Редактор словарей сообщений**).

Для отображения последней строки ОТ при старте монитора, нужно установить флаг **Считывать при старте** в разделе конфигурирования отчета тревог узла (вкладка **Отчет тревог/Дамп/Параметры** – см. **Задание параметров узла**).

Для квитиования сообщения нужно нажать ЛК, удерживая клавишу CTRL.

ГЭ 'ОТ узла'

ГЭ **ОТ узла**  предназначен для отображения сообщений отчета тревог узла, удовлетворяющих заданным при конфигурировании ГЭ условиям, и их квитиования (см. **Формат строки ОТ**).

Окно свойств ГЭ содержит вкладки **Свойства** и **Цвета**.

На вкладке **Свойства** конфигурируются следующие специфические атрибуты ГЭ:

- **Показать кодировку** – TRUE – отображение кодировки канала разрешено, FALSE – запрещено;
- **Использовать фильтры** – TRUE – отображать только те сообщения, которые удовлетворяют фильтрам (группам условий), заданным в разделе **Фильтры**; FALSE – отображать все сообщения;
- **Комбинирование фильтров** – логическое комбинирование заданных фильтров (**OR** или **AND**);
- **Фильтры** – в этом разделе задаются фильтры (группы условий) для отображения сообщений. Для создания фильтра нужно выполнить команду **Добавить фильтр** из контекстного меню раздела. Для удаления фильтра нужно выполнить команду **Удалить** из его контекстного меню.

В фильтре могут быть заданы следующие условия отбора сообщений ОТ для отображения:

- по времени – для отображения сообщений из определенного временного интервала используются параметры **Начальное время** и **Конечное время** (для разрешения их использования нужно задать значение TRUE соответственно для параметров **Использовать начальное время** и **Использовать конечное время**);
- по тексту – для задания условий отображения сообщений по имени канала, кодировке канала или тексту сообщения используются соответственно параметры **Имя**, **Кодировка** и **Текст**. Значения этих параметров задаются аналогично фильтрам ГЭ **Строка ОТ** (см. ГЭ **‘Строка ОТ’**);
- по категории – для задания условий отображения сообщений по их категории используется параметр **Категория** (см. также **Редактор словарей сообщений**);
- по квитированию – для отображения только неквитируемых сообщений нужно установить значение TRUE для параметра **Только неквитируемые**.

Если **Не применять это фильтр** = TRUE, заданный фильтр игнорируется.

Вкладка **Цвета** содержит атрибут **Цвета по умолчанию**. Если значение этого атрибута – TRUE, то фон сообщения имеет цвет, заданный для категории сообщения по умолчанию (в файле **gr_settings.xml**). Если значение атрибута **Цвета по умолчанию** – FALSE, на вкладке доступны инструменты задания произвольного цвета фона для всех категорий сообщений.

В реальном времени все параметры ГЭ могут быть изменены – чтобы открыть окно свойств, нужно нажать ПК на ГЭ.

Для квитирования сообщения нужно нажать ЛК в соответствующей строке ГЭ, удерживая клавишу CTRL.

Для отображения сообщений ОТ при старте монитора, нужно установить флаг **Считывать при старте** в разделе конфигурирования отчета тревог узла (вкладка **Отчет тревог/Дамп/Параметры** – см. **Задание параметров узла**).


Для корректной работы ГЭ **ОТ узла** рекомендуется задать формат даты как **%d.%m.%Y** или **%d/%m/%Y** (на вкладке **Отчет тревог/Дамп/Параметры** редактора узла).

ГЭ отображает дату и время в форматах, заданных на вкладке **Отчет тревог/Дамп/Параметры** (время квитирования – без десятых долей секунды).

Число считываемых строк ОТ может быть задано в файле *.cnf (см. **Задание параметров работы мониторов**).

Группа ГЭ 'T-FACTORY'

ГЭ 'Диаграмма Ганта'

ГЭ **Диаграмма Ганта**  предназначен для планирования работ и управления их выполнением.

Окно свойств ГЭ содержит вкладки **Осн. свойства** и **Привязки**.

На первой вкладке, наряду с типовыми параметрами (см. **Задание типовых свойств ГЭ**), конфигурируются следующие специфические атрибуты ГЭ:

- **Левая граница шкалы** – начальное значение левой границы временной шкалы (шкала отображается в правой части ГЭ);
- **Правая граница шкалы** – начальное значение правой границы временной шкалы;
- **Показать секунды** – если для этого атрибута установлено значение TRUE, метки времени на диаграмме отображаются с точностью до секунд, в противном случае – до минут;
- **Столбцы** – набор отображаемых в левой части ГЭ атрибутов каналов, привязанных к используемым аргументам экрана. Для добавления столбца нужно выделить раздел **Столбцы** и выполнить команду **Добавить** из контекстного меню. Для удаления столбца нужно выделить его и выполнить команду **Удалить** из контекстного меню.

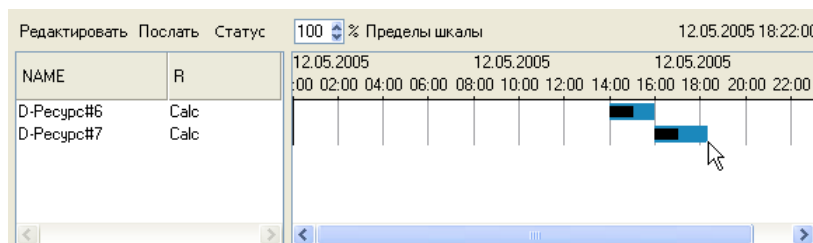
На второй вкладке задается перечень привязок (работ, которые будут планироваться с помощью ГЭ). Для добавления привязки нужно выделить раздел **Привязки** и выполнить команду **Добавить** из контекстного меню. Для удаления привязки нужно выделить ее и выполнить команду **Удалить** из контекстного меню.

Аргументы экрана, используемые данным ГЭ, должны быть привязаны к каналам D-РЕСУРС. Привязка аргумента к каналу ЕДИНИЦА ОБОРУДОВАНИЯ, для которого сконфигурированы ТО, равнозначна привязке к первому каналу D-РЕСУРС, автоматически создаваемому в этом случае монитором (см. **Канал класса ЕДИНИЦА ОБОРУДОВАНИЯ** и **Канал класса D-РЕСУРС**).

Диаграмма Ганта в реальном времени

В реальном времени диаграмма Ганта может находиться в режиме редактирования значений атрибутов 133, 134 и 135 привязанных каналов D-РЕСУРС или режиме отображения значений этих атрибутов и управления статусом работы.

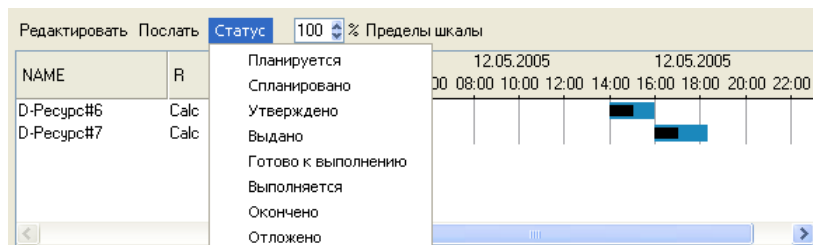
Для перехода в режим отображения нужно перевести кнопку **Редактировать** в отжатое состояние (при старте монитора диаграмма по умолчанию находится в режиме отображения). В этом режиме кнопка **Послать** деактивирована. Каждая работа на диаграмме отображается в виде двух полосок, наложенных одна на другую:



Нижняя полоска имеет цвет, заданный атрибутом **Цвет (факт)** (см. вкладку **Основные свойства** окна свойств ГЭ на рисунке выше), и отображает работу как время от ее начала (атрибут 133) до окончания (атрибут 134). При подведении курсора к левому/правому краю этой полоски время начала/окончания работы отображается в правой верхней части ГЭ.

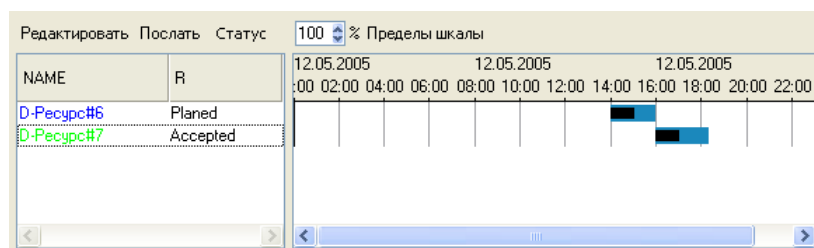
Верхняя полоска имеет цвет, заданный атрибутом **Цвет (план)**, и отображает плановую длительность работы (от значения атрибута 133 до суммы значений атрибутов 133 и 135). При подведении курсора к правому краю этой полоски с удержанием клавиши CTRL в правой верхней части ГЭ отображается плановая длительность работы (значение атрибута 135).

Для одновременного управления статусами всех работ используется меню **Статус**:




При изменении статуса имя канала на диаграмме Ганта меняет свой цвет. Цвета статусов заданы в файле **gr_settings.xml** и могут быть изменены. Файл **gr_settings.xml** размещается в директории MPB (ИС).

Для управления статусом отдельной работы нужно перейти в режим редактирования (перевести кнопку **Редактировать** в нажатое состояние), дважды нажать ЛК в поле, в которое выводится значение атрибута **R** или **In**, ввести в это поле численное значение статуса, нажать кнопку ОК (на клавиатуре) и нажать кнопку **Послать** (при переходе в режим редактирования эта кнопка активируется). Для просмотра заданного статуса нужно перейти в режим отображения:

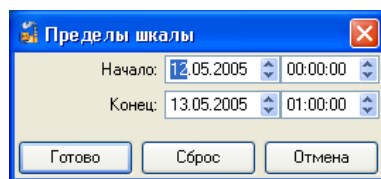


В режиме редактирования возможно изменение значений атрибутов 133, 134 и 135 привязанных каналов D-РЕСУРС. Для этого нужно выполнить следующие действия:

- методом drag-and-drop переместить края полосок на нужное время (при перемещении полоски, отображающей плановую длительность, нужно удерживать клавишу CTRL). В процессе перемещения время (для атрибутов 133 и 134) и длительность (для атрибута 135) отображаются в правой верхней части ГЭ;
- нажать кнопку **Послать**.

С помощью инструмента  временная шкала диаграммы масштабируется, при этом число делений и, соответственно, цена деления могут изменяться.


При нажатии кнопки **Пределы шкалы** на экране появляется следующий диалог:



С помощью этого диалога пределы шкалы могут быть изменены. Для установки пределов шкалы по умолчанию (от 00:00:00 текущих суток до 00:00:00 следующих) нужно нажать кнопку **Сброс**.

Группа ГЭ 'Зоны'

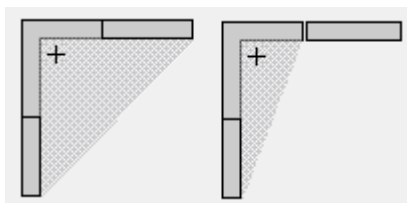
ГЭ 'Зона'

ГЭ **Зона**  размещается в графическом слое при установке точки привязки и не имеет специфических атрибутов (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

ГЭ представляет собой многоугольник, контур которого определяется автоматически по одному или нескольким ГЭ-ограничителям, окружающим точку привязки. К ГЭ-ограничителям относятся:

- ГЭ группы **Ломаные и кривые**;
- ГЭ группы **Элементы зданий**.

Если в контуре, образуемом ГЭ-ограничителями, обнаруживается разрыв более 1px, зона закольцовывается (на рисунке ниже крест обозначает точку привязки):




В отличие от обычного многоугольника, зона хранит только точку привязки, но не границы, поэтому может «потерять» свой контур при удалении или перемещении ГЭ-ограничителей (в этом случае зона не отображается в реальном времени). Однако если вокруг зоны, потерявшей свой контур, расположить один или несколько ГЭ-ограничителей, задающих другой контур, зона заполнит его в реальном времени.

В реальном времени, после начального определения своего контура, зона сохраняет свое расположение и конфигурацию (т.е. игнорируются любые изменения контура, создаваемого ГЭ-ограничителями зоны – например, при динамическом перемещении ограничителя).


Зона не может иметь более одного контура.

Группа ГЭ 'Элементы зданий'

ГЭ 'Стена'

ГЭ **Стена**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

ГЭ 'Ломаная стена'

ГЭ **Ломаная стена**  аналогичен ГЭ '**Ломаная с заливкой**'; дополнительно имеет возможность задания толщины контура.

ГЭ 'Окно'

Встроено несколько модификаций этого ГЭ; они не имеют специфических свойств и размещаются в графическом слое стандартным способом (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

ГЭ 'Дверь'

Встроено несколько модификаций этого ГЭ; они не имеют специфических свойств и размещаются в графическом слое стандартным способом (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

ГЭ 'Соединитель стен'


Встроено несколько модификаций этого ГЭ; они не имеют специфических свойств и размещаются в графическом слое стандартным способом (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

ГЭ 'Лестница'


Встроено несколько модификаций этого ГЭ, все они размещаются в графическом слое стандартным способом (см. **Размещение ГЭ и Задание типовых свойств ГЭ**).

Специфическими атрибутами лестниц (за исключением винтовой) являются **Ширина ступеней** и **Шаг ступеней** (оба параметра задаются в пикселях).

ГЭ ‘Прямоугольное помещение’

ГЭ **Прямоугольное помещение**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ ‘Забор’

ГЭ **Забор**  представляет собой ломаную линию с равномерно расположенными на ней метками и размещается в графическом слое аналогично элементам группы **Ломаные и кривые** (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ имеет следующие специфические атрибуты:

- **Тип меток** – встроенная метка (выбирается из меню);
- **Размер меток** – размер меток (px);
- **Шаг меток** – шаг меток (px).

Группа ГЭ 'Электрические элементы зданий'

ГЭ 'Трансформатор'

Встроено несколько модификаций этого ГЭ, все они размещаются в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ состоит из 1..3 окружностей (обмоток) одинакового диаметра (атрибут **Диаметр**) и толщины (атрибут **Толщина**). Другие параметры окружностей конфигурируются независимо с помощью атрибутов секций **Часть1...Часть3**. Каждая такая секция, помимо стандартных атрибутов, содержит следующие специфические атрибуты:


- **Тип** – форма;
- **Значок** – внутренний значок;
- **Выводы** – количество электрических выводов (0..3, изображаются как линии, примыкающие к окружности). Линии имеют толщину, заданную атрибутом **Толщина**, и цвет, соответствующий статическому значению цвета окружности.

Секция **Обрамляющий прямоугольник** содержит стандартные атрибуты конфигурирования опционального прямоугольника, обрамляющего все окружности. Атрибуты доступны, если **Обрамляющий прямоугольник** = TRUE.

Атрибут **Шаг выводов** задает расстояние между выводами для всех окружностей.

Все размеры задаются в пикселях.

ГЭ 'Рубильник'

ГЭ **Рубильник**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

ГЭ состоит из двух подводящих линий, рубильника (изображается как замыкающая полоска) и двух опциональных кнопок.


ГЭ имеет следующие специфические атрибуты:

- **Состояние ВЫКЛ** – рубильник в состоянии ГЭ **ВЫКЛЮЧЕН** (сместить или скрыть). В состоянии ГЭ **ВКЛЮЧЕН** рубильник всегда видим и размещается между подводящими линиями;
- Секция **Рубильник** – стандартные атрибуты конфигурирования рубильника;
- Секция **Подводящие линии** – стандартные атрибуты конфигу-

рирования подводящих линий;


- Секция **Вход** содержит атрибуты **Привязка** (аргумент экрана) и **Константа**. Если значение привязанного аргумента равно заданной константе, ГЭ переходит в состояние ВКЛЮЧЕН, в противном случае – в состояние ВЫКЛЮЧЕН;
- Секция **Выход** содержит атрибуты **Привязка** (аргумент экрана), **Константа ON** и **Константа OFF**. При нажатии кнопки **ON** привязанный аргумент принимает значение **Константа ON**, при нажатии кнопки **OFF** – значение **Константа OFF**;
- Секция **Кнопки** содержит стандартные атрибуты конфигурирования кнопок. Кнопки отображаются, если **Показать** = TRUE.

ГЭ 'Заземление'

ГЭ **Заземление**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задавание типовых свойств ГЭ**).

Группа ГЭ 'Кондиционирование'


ГЭ 'Поток'

ГЭ **Поток**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), ГЭ имеет следующие специфические атрибуты:

- **Шаг, рх** – расстояние между стрелками потока и ширина стрелок (в пикселях);
- **Привязка On/Off** – привязка к аргументу экрана. Если значение аргумента отлично от 0, ГЭ анимирован, в противном случае – статичен.

ГЭ 'Заслонка'

ГЭ **Заслонка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), ГЭ имеет следующие специфические атрибуты:

- **Ориентация** – положение открытой заслонки;
- **Шаг, рх** – расстояние между заслонками (в пикселях);
- **Привязка On/Off** – привязка к аргументу экрана, задающая поворот заслонки:
 - в аналоговом режиме (**Режим = Аналоговый**) значение привязки задает угол поворота в градусах;
 - в дискретном режиме (**Режим = Дискретный**): 0 – открыто, 1 – закрыто.

ActiveX в TRACE MODE 6

С помощью ActiveX-компонентов можно реализовать следующие функции:

- отображение/изменение значений аргументов графического экрана, содержащего компонент;
- отображение последнего сообщения отчета тревог.

TRACE MODE взаимодействует с ActiveX-компонентом либо через интерфейс **IDispatch**, либо через описанные ниже собственные интерфейсы. К собственным относятся **custom**-интерфейсы и дополнительные интерфейсы.

TRACE MODE 6 поддерживает те же интерфейсы, что и TRACE MODE 5, однако в TRACE MODE 6 контейнер оперирует только аргументами экрана, в который вставлен компонент.

Свойства РПД как ActiveX-контейнера

РПД предоставляет следующие свойства окружения:

- **BackColor**
- **ForeColor**
- **UserMode**
- **Palette**
- **LocaleID**

Переменная **UserMode** устанавливается в состояние **TRUE**, **BackColor** имеет значение цвета фона, а **ForeColor** – черный цвет. Все ActiveX-компоненты уведомляются о его изменении.

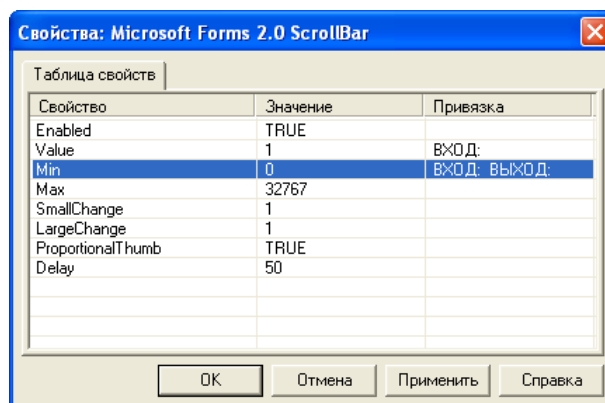
При вставке и последующих вызовах экрана, содержащего компонент, ActiveX активизируется в режиме **OLEIVERB_SHOW**. При исчезновении с экрана компонент деактивируется.

Взаимодействие с компонентами ActiveX

Интерфейс IDispatch

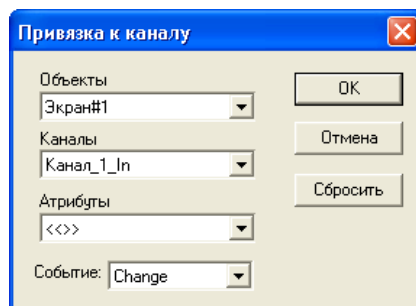
Пусть используется готовый ActiveX-компонент, не поддерживающий спецификации TRACE MODE и имеющий интерфейс **IDispatch**.

В этом случае при вставке и редактировании свойств компонента РПД добавляет дополнительную вкладку **Таблица свойств**. В нее включаются только те свойства, которые имеют метод их изменения. Метод может быть числовым, логическим или текстовым.



Свойства, имеющие числовой или логический метод изменения (**Value**, **Min**, **Max** и т.п.), могут быть связаны с любым числовым аргументом экрана. Свойство с текстовым методом изменения (**Text** и т.п.) можно сконфигурировать, в том числе, для отображения последнего сообщения отчета тревог.

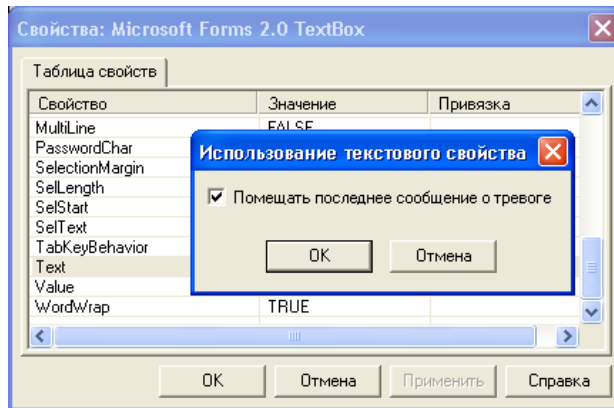
Для свойства, имеющего числовой или логический метод изменения, надо нажать ПК в соответствующем поле **Привязка** таблицы. В появившемся меню надо выбрать **Вход** (для отображения значения) или **Выход** (для задания значения). После этого на экран выводится следующий диалог:



Данный диалог доступен, если в ОС зарегистрирована библиотека **ChBase.dll**.

С помощью этого диалога нужно привязать свойство к аргументу экрана (аргумент выбирается в списке **Каналы**), а также установить событие, по которому осуществляется передача данных. В списке **Атрибуты** нужно выбрать **<<>>**.

Для отображения последнего сообщения отчета тревог надо нажать ПК в поле **Привязка** свойства с текстовым методом изменения, в меню выбрать **Вход** и задать вывод последнего сообщения ОТ в появившемся диалоге:



Отображение последнего сообщения ОТ компонентом **Microsoft Forms 2.0 TextBox** показано на рисунке:

| | | | |
|-----------------------|--------|-----|-----|
| 09.09.2005 11:47:28.3 | Пила#1 | ТС5 | =41 |
|-----------------------|--------|-----|-----|

Custom-интерфейсы для ActiveX

Пусть разрабатывается новый ActiveX-компонент для использования в TRACE MODE.

Если описанной выше функциональности недостаточно, то можно реализовать в компоненте интерфейс **IChInfo** (см. **tmxaux.idl**).

Для привязки к аргументу можно использовать страницу свойств **tmChanPage**. При этом надо указать ее явно в списке поддерживаемых.

Кроме того, можно использовать непосредственно интерфейс **ItmChBrowser** (см. **tmxaux.idl**). В этом случае надо написать свой интерфейс пользователя для привязки к нескольким аргументам. Если в ActiveX-компоненте используется собственный интерфейс привязки к аргументам и **tmChanPage** не требуется, то следует реализовать метод **GetChInfoEx** интерфейса **IChInfo** с кодом возврата **S_OK**.

Интерфейс **ItmChBrowser** не может быть создан явным образом (вызовом **CoCreateInstance**), а передается методом **SetChanBrowser** интерфейса **IChInfo**.

Для изменения значения одного или нескольких аргументов одновременно служит интерфейс **ItmDataLink**. Он предоставляется контейнером аналогично **ItmChBrowser** и имеет всего один метод, служащий именно

этой функции. Интерфейс **ItmGrChBrowser**, специфицированный в том же файле, используется для внутренних нужд РПД.

Структуры данных, используемых методами упомянутых интерфейсов, описаны в файле **rttypes.idl**. Они подробно прокомментированы.

Если компонент готов к работе с TRACE MODE через специализированные интерфейсы, то его можно зарегистрировать в категории **TraceMode Controls** (см. пример), что позволит использовать фильтр при выборе компонента для вставки.

Комплект файлов для разработчика

В поддиректории **TMX** директории ИС находится набор файлов для разработчика ActiveX-компонента:

- **tmxaux.idl** – спецификация интерфейсов **IchInfo**, **ItmChBrowser**;
- **rttypes.idl** – ряд вспомогательных структур данных;
- **ChBase.tlb** – библиотека типов, содержащая страницы свойств **tmChanPage** и **TmxAmbPage**.
- Файлы примера.

Пример компонента

В качестве иллюстрации простого компонента, реализующего **IchInfo**, предлагается **TraceMode Spin Control (Tmxlib.dll)**, исходные тексты которого находятся в поддиректории **TMX** каталога инсталляции ИС. Этот компонент представляет реализацию комбинации окон **static** и **UpDown**, то есть вариацию традиционной пары **Edit** и **UpDown**. Он принимает измененные значения аргумента и может управлять им при нажатиях соответствующих стрелок элемента **Up/Down**. Данный компонент разработан с использованием MS Visual C++5.0 и библиотеки ATL 2.0. Необходимая функциональность в данном случае могла бы быть достигнута и с использованием **IDispatch**.

Дополнительные интерфейсы для ActiveX

Дополнительный набор дуальных интерфейсов предназначен для разработки ActiveX-компонентов с помощью инструментов типа Delphi и Visual Basic.

Служебные файлы и примеры на VB и Delphi находятся в поддиректории **TMX\VB_Delphi** директории установки ИС:

- **rtmdata.h** – определения констант;
- **rtmd.idl** – определения интерфейсов;
- **rtmd.tlb** – результат компиляции **rtmd.idl**;

- **TmVbCtl***. * – пример на Visual Basic;
- **TmDelCtl***. * – пример на Delphi.

Особенности интерфейсов

При выполнении обычной (единичной) перепривязки будет вызван метод **writeTags** с одной новой привязкой.

Если необходима перепривязка компонента в реальном времени по его инициативе, то в нем следует инициировать событие с идентификатором **DISPID_REFRESH** (-550). После этого сервер запросит **readTags** и возобновит режим обмена с сервером реального времени. В примере на Delphi реализована такая возможность.

Разработанные на основе описываемых в этом разделе интерфейсов компоненты менее эффективны, чем компоненты, написанные на C++ с использованием **custom**-интерфейсов, – как по быстродействию, так и по расходованию памяти.

Delphi и VB не предоставляют возможности зарегистрировать компонент в произвольной категории, в частности, в **TraceMode Controls**.

Описание прилагаемых примеров

В качестве примера реализован компонент **Data View**, который демонстрирует:

- просмотр аргументов;
- произвольный набор аргументов для отображения их значений;
- произвольный набор аргументов для одновременной посылки значения;
- применение флагов **TMXF_NEEDALARM** и **TMXF_HIDEONRUN**.

Пример на Delphi демонстрирует сохранение указанных флагов, а также возможность изменения набора отображаемых аргументов в реальном времени.

Данные примеры проверялись на Delphi 5 и Visual Basic 6.0.

Операции с графическими объектами

Редактор представления данных имеет мощный механизм тиражирования готовых решений для создания интерфейса оператора. Для этого используются графические объекты. Оформленные в виде объектов типовые графические фрагменты могут вставляться в графические экраны любых проектов.

В дереве структуры проекта графические объекты размещаются в группах (библиотеках) **Графические элементы** слоя **Ресурсы** или слоя **Библиотеки компонентов** (см. **Классификация слоев**, **Классификация компонентов**, **Группы слоя 'Библиотеки компонентов'**, **Сохранение проекта для редактирования**, а также **Копирование и вставка объекта структуры**).

Для размещения графических объектов на экране используются ГЭ **Объект** и ГЭ **Объект в окне** (см. **Группа ГЭ 'Объекты'**). Графический объект может быть размещен на экране посредством перетаскивания из дерева структуры проекта (см. **ГЭ 'Объект'**, а также **Специальные операции с графическими экранами**).

Графический объект может содержать несколько слоев, однако при вставке в экран все его ГЭ будут размещены в одном и том же слое.

Результаты редактирования графического объекта в библиотеке отражаются на всех экранах, где этот объект используется.

Графический объект может быть сохранен в исполняемый файл ***.res** как экран или, для повышения быстродействия графики, как картинка (см. **Задание параметров графического экрана** и **Файлы узла, создаваемые при экспорте**).

В состав поставки TRACE MODE 6 включено несколько групп ГО (см. **Поставляемая пользовательская библиотека компонентов**).


Операции с ресурсными библиотеками


В проекте могут быть использованы ресурсы из внешних файлов (тексты, рисунки, видеоклипы). Такие ресурсы должны быть предварительно собраны в соответствующие **ресурсные библиотеки** (компоненты проекта). Для операций с библиотеками используется навигатор проекта (см. **Редактирование структуры проекта**).

Для редактирования и использования ресурсных библиотек в интегрированную среду встроены соответствующие редакторы и навигаторы, а также панель инструментов **Ресурсные библиотеки**:



Эта панель, помимо типовых инструментов редактирования (см. **Типовые средства редактирования**), содержит следующие команды:

 – перейти в режим редактирования выделенного текстового ресурса (для перехода к редактированию можно также дважды нажать ЛК в строке описания ресурса);

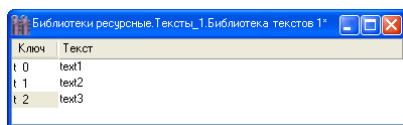
 – заменить выделенный ресурс ресурсом из файла;


 – импортировать в библиотеку ресурс из файла.

Кроме того, в редакторах и навигаторах поддерживаются соответствующие типовые операции редактирования (см. **Типовые операции редактирования**).

Библиотеки текстов


Библиотека текстов отображается в соответствующем редакторе в виде таблицы, содержащей столбцы **Ключ** и **Текст**. В поле **Ключ** выводится идентификатор ресурса (этот параметр присваивается тексту автоматически при его добавлении в библиотеку и недоступен для редактирования). В поле **Текст** выводится собственно текст. Эти два параметра составляют **строку описания ресурса**. Примерный вид библиотеки в редакторе показан на рисунке.



Чтобы вручную добавить ресурс в библиотеку, нужно нажать кнопку 


(**Создать**) панели инструментов **Ресурсы библиотеки** и ввести текст с клавиатуры (в поле **Текст**).

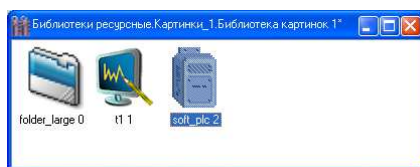
Тексту, добавленному в библиотеку вручную, присваивается идентификатор вида **t n**, где **n** – порядковый номер (начиная с 0).

Чтобы импортировать в библиотеку текст из файла (файл должен иметь текстовый формат и расширение **.txt**), нужно нажать кнопку  (**Импорт**) панели инструментов **Ресурсы библиотеки** и указать в появившемся диалоге требуемый файл. Импортированному тексту присваивается идентификатор вида **имя_файла_без_расширения n**, где **n** – порядковый номер.

Библиотеки рисунков и видеоклипов

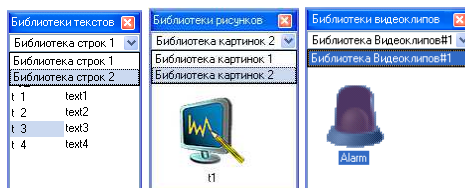
Операции по редактированию библиотек рисунков и видеоклипов идентичны.

Чтобы импортировать в соответствующую библиотеку рисунок (формата **.bmp**, **.png**, **.jpg** или **.xpm**) или видеоклип (**.mpg** или **.avi**), нужно нажать кнопку  (**Импорт**) панели инструментов **Ресурсы библиотеки** и указать в появившемся диалоге требуемый файл. Импортированному ресурсу, как и при импорте текста в соответствующую библиотеку, присваивается идентификатор вида **имя_файла_без_расширения n**:



Использование библиотечных ресурсов

При выборе графического элемента (см. **Размещение ГЭ** и **Типовые средства редактирования**), для которого возможно использование библиотечных ресурсов какого-либо типа, на экран выводится соответствующий **навигатор**. Навигатор содержит список библиотек соответствующего типа и отображает ресурсы выбранной библиотеки:



Для выбора ресурса нужно нажать на нем ЛК (ресурс при этом выделяет-

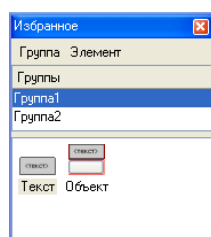
ся).

Результаты редактирования ресурса в библиотеке отражаются на всех экранах, где этот ресурс используется.

Библиотека избранных ГЭ и библиотека избранных еГЭ

Любой графический элемент (объект) после размещения на графическом экране/панели и конфигурирования свойств может быть помещен соответственно в библиотеку избранных ГЭ/библиотеку избранных еГЭ для последующего использования при разработке графического интерфейса.

Видимостью окна просмотра библиотеки избранных элементов управляет команда **Избранное** меню **Вид** (см. **Главное меню и панели инструментов РПД**):



Если в ИС активен РПД, в окне отображается содержимое библиотеки избранных ГЭ; если еРПД – содержимое библиотеки избранных еГЭ.

Меню **Группа** этого окна содержит следующие команды:

- **Новый** – создать группу в библиотеке. Список групп отображается в верхней части окна (**Группы**). Объекты группы, выделенной в списке, отображаются в нижней части окна. Для перехода к редактированию имени выделенной группы нужно нажать на ней ЛК;
- **Удалить** – удалить выделенную группу;
- **Сохранить** – сохранить библиотеку в файл. Библиотека избранных ГЭ сохраняется в файл **%TRACE MODE%\grext\patterns.bin**, библиотека избранных еГЭ – в файл **%TRACE MODE%\grext\patterns_emb.bin**. При запуске ИС загружает обе библиотеки.

Меню **Элемент** содержит типовые команды для работы с элементами внутри библиотеки.

Чтобы поместить выделенный элемент (объект) в текущую группу библиотеки, нужно выполнить команду **Копировать в избранное** из контекстного меню элемента (объекта).

Для размещения элемента библиотеки на экране/панели используется метод drag-n-drop (особенности размещения описаны в разделе **ГЭ 'Объект'**).

Графические панели

Графический интерфейс оператора для узлов, которые исполняются мониторами семейства **Embedded MPB** на аппаратных средствах с ограниченной производительностью (например, в контроллерах с ОС Windows CE), разрабатывается в виде набора **графических панелей**.

Профайлер без поддержки графических экранов (**rtmg32.exe**) обеспечивает отображение графических панелей.

Для создания шаблона панели нужно выполнить команду **Графическая панель** из контекстного меню слоя **Шаблоны экранов**.

Для вызова шаблона графической панели используется канал класса CALL с типом вызова 5, **Panel** (см. **Атрибуты канала класса CALL**).

Если в узле вызывается несколько панелей, то их видимостью можно управлять в реальном времени – для этого входному значению соответствующего канала вызова нужно присвоить значение 2. Панель для отображения при запуске узла выбирается так же, как экран (см. **Особенности вызова графического экрана**).

Если для канала CALL вызова графической панели задан тип пересчета **FAST EXE** (см. **Период пересчета канала**), такая панель, если она видима, перерисовывается с повышенным приоритетом. При этом перерисовка низкоприоритетной панели может быть затруднена (возможно «мергание»).

Для редактирования шаблонов панелей в ИС встроен редактор **еРПД**, представляющий собой модификацию **РПД**. По функциям редактирования **еРПД** аналогичен **РПД**. К особенностям **еРПД** относятся следующие:

- **еРПД** поддерживает слои и Z-позиционирование для удобства редактирования панелей, однако слои и Z-позиционирование не поддерживаются в **Embedded MPB**;
- **еРПД** содержит собственный набор встроенных элементов (**еГЭ**). Выбор **еГЭ** для размещения аналогичен выбору **ГЭ** (см. **Размещение ГЭ**);
- поворот **еГЭ** не поддерживается в **Embedded MPB**;
- графическая панель не может быть всплывающей (см. **Задание параметров графического экрана**);
- фон панели типа **Изображение** не поддерживается в **Embedded MPB**.

В реальном времени:

- положение границ панели можно изменять с помощью мыши;
- прокрутка дерева объектов узла в графической оболочке монитора не влияет на отображение графической панели;
- для перемещения по eГЭ управления можно использовать кнопку **SPACE**.

Ошибка 16 свидетельствует о нехватке памяти для исполнения узла; а если в протоколе профайлера **<имя файла prj>_<ordinal>.txt** появляется сообщение «**Static picture for <имя канала> = Error**», то, вероятнее всего, в узле необходимо сократить число графических панелей, содержащих растровые изображения и видеоклипы.

Операции с аргументами в eРПД

В eРПД автоматически выполняются следующие операции при перетаскивании (drag-n-drop) выделенного аргумента (выделенных аргументов) из редактора аргументов на eГЭ, размещенный на панели:

- на eГЭ **'Текст'** – задание привязки eГЭ;
- на объемные eГЭ – если атрибут **Базовый цвет** не динамизирован – динамизация **Arg >= Конст. (Константа = 0)**. Если атрибут динамизирован – замена привязки (вид индикации и значение константы не изменяются);
- на eГЭ **'Кнопка'** и eГЭ **'Кнопка XOR'** – привязка eГЭ;
- на eГЭ **'Тренд'** – добавление кривых в соответствии с возрастанием порядковых номеров аргументов в выделенной группе;
- на eГЭ **'Дата и время'** – привязка eГЭ.

Описание элементов графических панелей

К отличиям типовых свойств eГЭ и ГЭ (см. **Задание типовых свойств ГЭ** и **Операции с аргументами в eРПД**) относятся следующие:


- eГЭ имеют меньший набор типовых статических атрибутов (нет атрибутов **Прозрачность**, **Видимость при старте**, **Подсказка**, **Выделение в MPB** – см. **Статические атрибуты ГЭ**);
- eГЭ не имеют функций управления (см. **Функции управления ГЭ**). Взамен предусмотрены отдельные eГЭ, с помощью которых реализуется управление из графики;
- eГЭ не имеют динамических свойств (см. **Динамические свойства ГЭ**), и, кроме того, число динамизируемых атрибутов eГЭ минимально (см. **Динамизация атрибута ГЭ**). Взамен предусмотрены отдельные динамические eГЭ;
- при наведении мыши на eГЭ управления элемент активируется,

при этом подсвечивается его контур;

- для выполнения команды управления можно нажать ЛК или **ENTER**;
- Embedded MPB не поддерживает виды индикации **Arg в интервале**, **Атр.46 в диапазоне** и **Набор {Arg = Конст}**;
- мигание поддерживается только для индикации **Arg в диапазоне**;
- кнопки, тренд, строка отчета тревог, растровое изображение и видеоклип отображаются всегда поверх других **еГЭ**;
- динамические и динамизированные **еГЭ** отображаются всегда поверх статических;
- цветовая заливка **еГЭ** всегда сплошная, другие стили заливки Embedded MPB не поддерживает;
- виды начертания шрифта **Наклонный**, **Жирный**, **Подчеркнутый** поддерживаются только в **еГЭ Текстовый ресурс**.


Группа **еГЭ** ‘Линии’

еГЭ ‘Линия’

еГЭ Линия  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Группа **еГЭ** ‘Текст’

еГЭ ‘Текст’

еГЭ Текст  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

При задании привязки динамизируется атрибут **Текст** данного **еГЭ** (см. **Динамизация атрибута ГЭ**).

Ограничения Embedded MPB:

- стиль контура **еГЭ** – всегда сплошная линия;
- толщина контура – всегда 1 px.


Группа еГЭ 'Меню'

еГЭ 'Меню управления'


еГЭ Меню управления  аналогичен ГЭ 'Меню управления'.

Группа еГЭ 'Ломаные и кривые'


еГЭ 'Ломаная линия'

еГЭ Ломаная линия  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ и Задание типовых свойств ГЭ**).


еГЭ 'Многоугольник'

еГЭ Многоугольник  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ и Задание типовых свойств ГЭ**).


еГЭ 'Ломаная с заливкой'

Данный еГЭ  аналогичен ГЭ 'Ломаная с заливкой'.

еГЭ 'Разомкнутая кривая'


еГЭ Разомкнутая кривая  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ и Задание типовых свойств ГЭ**).

еГЭ 'Замкнутая кривая'


еГЭ Замкнутая кривая  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Positionирование узловых точек ГЭ и Задание типовых свойств ГЭ**).

Группа еГЭ ‘Прямоугольники’

еГЭ ‘Контур’


еГЭ **Контур**  не имеет специфических свойств и размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

еГЭ ‘Панель’

еГЭ **Панель**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), еГЭ **Панель** имеет ряд специфических атрибутов (см. **Специфические атрибуты еГЭ ‘Панель’ и ‘Рамка’**).

еГЭ ‘Рамка’

еГЭ **Рамка**  размещается в графическом слое стандартным способом (см. **Размещение ГЭ**).

Наряду с типовыми свойствами (см. **Задание типовых свойств ГЭ**), еГЭ **Рамка** имеет ряд специфических атрибутов (см. **Специфические атрибуты еГЭ ‘Панель’ и ‘Рамка’**).

Специфические атрибуты еГЭ ‘Панель’ и ‘Рамка’

еГЭ **Панель** и **Рамка** имеют те же специфические атрибуты, что и ГЭ **Панель** и **Рамка** (см. **Специфические атрибуты ГЭ ‘Панель’ и ‘Рамка’**).

Группа еГЭ ‘Плоские фигуры’

Общие специфические атрибуты плоских еГЭ


еГЭ данной группы размещаются в графическом слое стандартным способом (см. **Размещение ГЭ**).

Специфическими атрибутами, общими для всех еГЭ группы **Плоские фигуры**, являются следующие:


- атрибуты раздела **Вид индикации**, с помощью которых для заливки еГЭ конфигурируется вид индикации (см. **Динамизация**

атрибута ГЭ).


еГЭ 'Прямоугольник'

Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Прямоугольник**  не имеет специфических свойств.


еГЭ 'Плоский клапан'

Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Плоский клапан**  не имеет специфических свойств.


еГЭ 'Треугольник'

Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Треугольник**  не имеет специфических свойств.


еГЭ 'Овал'

Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Овал**  имеет те же специфические атрибуты, что и ГЭ **Овал** (см. **ГЭ 'Овал'**).

еГЭ 'Стрелка'


Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Стрелка**  не имеет специфических свойств.

еГЭ 'Эллипс, сектор'

Кроме типовых свойств (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех еГЭ группы **Плоские фигуры** (см. **Общие специфические атрибуты плоских еГЭ**), еГЭ **Эллипс, сектор**  имеет те же специфические атрибуты, что и ГЭ **Эллипс, сектор** (см. **ГЭ 'Эллипс, сектор'**).

Группа eГЭ 'Ресурсы'


eГЭ 'Текстовый ресурс'

eГЭ **Текстовый ресурс**  аналогичен ГЭ **Текстовый ресурс** (см. ГЭ 'Текстовый ресурс'). В отличие от ГЭ **Текстовый ресурс**, eГЭ **Текстовый ресурс** не имеет динамизируемых атрибутов.

eГЭ 'Растровое изображение'

eГЭ **Растровое изображение**  аналогичен ГЭ **Растровое изображение** (см. ГЭ 'Растровое изображение').

eГЭ 'Видеоклип'

eГЭ **Видеоклип**  аналогичен ГЭ **Видеоклип** (см. ГЭ 'Видеоклип'). В отличие от ГЭ **Видеоклип**, eГЭ **Видеоклип** воспроизводится только непрерывно.

eГЭ 'Стандартный видеоклип'

Данный eГЭ аналогичен ГЭ 'Стандартный видеоклип'.


Группа eГЭ 'Объемные фигуры'

Общие специфические атрибуты объемных eГЭ


eГЭ данной группы размещаются в графическом слое стандартным способом (см. **Размещение ГЭ**).

Объемные eГЭ имеют те же общие специфические атрибуты, что и объемные ГЭ (см. **Общие специфические атрибуты объемных ГЭ**)


eГЭ 'Цилиндр'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных eГЭ (см. **Общие специфические атрибуты объемных eГЭ**), eГЭ **Цилиндр**  имеет те же специфические атрибуты, что и ГЭ **Цилиндр** (см. ГЭ 'Цилиндр').


еГЭ 'Сфера'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Сфера**  имеет те же специфические атрибуты, что и ГЭ **Сфера** (см. **ГЭ 'Сфера'**).


еГЭ 'Конус'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Конус**  имеет те же специфические атрибуты, что и ГЭ **Конус** (см. **ГЭ 'Конус'**).


еГЭ 'Тор'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Тор**  имеет те же специфические атрибуты, что и ГЭ **Тор** (см. **ГЭ 'Тор'**).


еГЭ 'Пирамида'

еГЭ **Пирамида**  не имеет собственных специфических свойств (см. **Задание типовых свойств ГЭ** и **Общие специфические атрибуты объемных еГЭ**).


еГЭ 'Емкость'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Емкость**  имеет те же специфические атрибуты, что и ГЭ **Емкость** (см. **ГЭ 'Емкость'**).

еГЭ 'Клапан'


Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Клапан**  имеет те же специфические атрибуты, что и ГЭ **Клапан** (см. **ГЭ 'Клапан'**).

еГЭ 'Насос'


Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Насос**  имеет те же специфические атрибуты, что и ГЭ 'Насос'.

еГЭ 'Труба'


Данный еГЭ размещается на графической панели аналогично элементам группы **Ломаные и кривые** (см. **Размещение ГЭ** и **Позиционирование узловых точек ГЭ**).

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Труба**  имеет те же специфические атрибуты, что и ГЭ **Труба** (см. **ГЭ 'Труба'**).


еГЭ 'Рельефный конус'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Рельефный конус**  имеет те же специфические атрибуты, что и ГЭ **Рельефный конус** (см. **ГЭ 'Рельефный конус'**).

еГЭ 'Криволинейный конус'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Криволинейный конус**  имеет те же специфические атрибуты, что и ГЭ **Криволинейный конус** (см. **ГЭ 'Криволинейный конус'**).

еГЭ 'Градиент'

Кроме типовых атрибутов (см. **Задание типовых свойств ГЭ**) и специфических атрибутов, общих для всех объемных еГЭ (см. **Общие специфические атрибуты объемных еГЭ**), еГЭ **Градиент**  имеет те же специфические атрибуты, что и ГЭ **Градиент** (см. **ГЭ 'Градиент'**).

Группа еГЭ 'Кнопки'


Для выполнения функции, заданной при конфигурировании кнопки, нуж-

но нажать и отпустить ЛК на кнопке.

Специфическими атрибутами, общими для кнопок, являются следующие:

- **Два состояния** – если этот флаг установлен, устойчивыми являются оба состояния кнопки (нажатое и отжатое), в противном случае устойчиво только отжатое состояние;
- **Вид представления** – если для этого атрибута задано значение **Текст**, доступны типовые атрибуты задания текста для вывода на кнопку. Если для атрибута **Вид представления** задано значение **Изображение**, доступны следующие атрибуты:
 - **Картинка (отжато)** – картинка из библиотеки изображений, отображаемая на кнопке в ее отжатом состоянии;
 - **Картинка (нажато)** – картинка из библиотеки изображений, отображаемая на кнопке в ее нажатом состоянии;
- **Привязка** – выбор аргумента для привязки.

еГЭ 'Кнопка'

еГЭ **Кнопка**  выполняет функцию прямой передачи значения в привязанный аргумент.

Данный еГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).


еГЭ **Кнопка** имеет следующие специфические атрибуты (см. также **Группа еГЭ 'Кнопки'**):

- **Передать значение** – значение для передачи в аргумент.

При установленном флаге **Два состояния** кнопка является также индикатором (**arg** – значение привязанного аргумента):

- если **arg** = **Передать значение**, кнопка «нажата»;
- если **arg** <> **Передать значение**, кнопка «отжата».

еГЭ 'Кнопка XOR'


еГЭ **Кнопка XOR**  выполняет функцию **arg ^ val** (^ – операция побитового XOR, **arg** – значение привязанного аргумента, **val** – значение, заданное атрибутом **Значение**) и присваивает полученный результат привязанному аргументу.

Данный еГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ**, **Задание типовых свойств ГЭ**, а также **Группа еГЭ 'Кнопки'**).

При установленном флаге **Два состояния** кнопка является также индикатором:

- если установлены все биты **arg**, соответствующие маске **Значение**, кнопка «нажата»;
- если сброшен хотя бы один бит **arg**, соответствующий маске **Значение**, кнопка «отжата».

еГЭ 'Переход'

еГЭ **Переход**  выполняет функцию перехода на графическую панель, заданную атрибутом **Канал**.

Данный еГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ, Задание типовых свойств ГЭ**, а также **Группа еГЭ 'Кнопки'**).

еГЭ имеет следующие специфические атрибуты:

- **Канал** – шаблон/канал вызова панели. Выбор шаблона равнозначен выбору канала вызова этого шаблона с младшим ID.

Группа еГЭ 'Выключатели'


еГЭ 'Выключатель'

еГЭ **Выключатель** представляет собой аналог **ГЭ 'Выключатель'**.

В Embedded MPV данный еГЭ всегда имеет размеры по умолчанию.

Группа еГЭ 'Тренды'

еГЭ 'Тренд'

еГЭ **Тренд**  представляет собой аналог **ГЭ 'Тренд'**.

По основным свойствам еГЭ **Тренд** имеет в мониторе следующие отличия от **ГЭ Тренд** (см. также **Операции с аргументами в еРПД**):

- нет дискретной панели (тренд отображает изменение аргументов целочисленного типа данных на аналоговой панели);
- конфигурирование тренда в реальном времени не поддерживается, и нет соответствующих инструментов;
- не используются атрибуты, предназначенные для работы с архивными значениями, и нет соответствующих инструментов;
- ориентация – только горизонтальная;
- для уменьшения масштаба по времени нужно нажать ЛК на тренде; для возврата к заданной временной шкале нужно нажать ЛК по-

вторно;

- масштабирование по оси значений не поддерживается;
- нет заголовка;
- в легенде отображаются только имена кривых;
- нет визира;
- дополнительный цвет сетки не используется;
- не поддерживается передача значений левой и правой временных границ тренда в аргументы;
- ось значений – одна, ее подписи соответствуют диапазону первой кривой, хотя заданные для каждой кривой диапазоны обрабатываются;
- сохранение буфера в файл не поддерживается;
- не используются атрибуты **Масштаб дискрет**, **Цвета статусов**.


По свойствам кривых eГЭ **Тренд** имеет отличия от ГЭ **Тренд**:

- не поддерживаются псевдонимы кривых; в легенде всегда отображаются имена каналов;
- не поддерживается выделение точек кривых с помощью меток;
- стиль кривой – сплошная линия;
- не используются атрибуты **Стиль при I<>0 и W=0**, **Стиль при I=0 и W=1** и **Стиль при I<>0 и W=1**;
- не используется значение **Статус** атрибута **Использовать как**.

В браузере (см. **Особенности выполнения узла в браузере**) большинства этих ограничений нет.

Группа eГЭ 'Дата и время'

eГЭ 'Дата и время'


eГЭ **Дата и время**  предназначен для отображения значения аргумента с временным или 4-байтовым целочисленным типом данных (задается атрибутом **Привязка**). Формат отображения задается атрибутом **Формат**:

- **Дата и время** – дата и время;
- **Дата** – только дата;
- **Время** – только время.
- **Интервал** – интервал в формате **h...hN:MM:SS**.

eГЭ **Дата и время** размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Группа еГЭ 'Значение аргумента'

еГЭ 'Ввод значения'

еГЭ **Ввод значения**  предназначен для изменения значения аргумента, заданного атрибутом **Основная привязка**.


Новое значение вводится с клавиатуры после активации еГЭ. При вводе значения даты и времени в качестве разделителей используются:

- между элементами даты – точка или слеш;
- между элементами времени – точка, двоеточие или слеш;
- между датой и временем – двоеточие, слеш или без пробела.

еГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**).

Группа еГЭ 'Гистограммы'

еГЭ 'Гистограмма'

еГЭ **Гистограмма**  предназначен для отображения значения привязанного аргумента числового формата в виде закрашенной области (слоя).

еГЭ размещается в графическом слое стандартным способом (см. **Размещение ГЭ** и **Задание типовых свойств ГЭ**) и имеет форму прямоугольника.

еГЭ имеет следующие специфические атрибуты:

- **Направление** – направление заливки (**вверх, вниз, вправо, влево**).
- **Имя** – имя слоя. Для перехода к редактированию нужно дважды нажать ЛК в этом поле.
- **Привязка** – выбор аргумента.
- **Мин, Макс** – пределы шкалы. Например, если выбрано направление заливки **влево**, **Мин** соответствует правой, а **Макс** – левой границе графического элемента.

Группа еГЭ 'Отчет тревог'

еГЭ 'Строка ОТ'

еГЭ **Строка ОТ**  аналогичен ГЭ **Строка ОТ** (см. ГЭ 'Строка ОТ').

Для квитирования сообщения нужно нажать ЛК или ENTER.

еГЭ 'ОТ узла'

еГЭ **ОТ узла**  аналогичен ГЭ **ОТ узла** (см. **ГЭ 'ОТ узла'**).

Ограничения Embedded MPB:

- не поддерживаются фильтры.

Глава 9

Архивирование

Отчет тревог узла

Отчет тревог (ОТ) – это текстовый файл (ASCII), в который заносятся сообщения, генерируемые в различных ситуациях при работе АСУ (см. **Отчет тревог и генерация сообщений**).

Отчет тревог конфигурируется для узла (на вкладке **Отчет тревог/Дамп/Параметры** редактора узла – см. **Задание параметров узла**).

В отчет тревог могут быть записаны сообщения следующих видов:

- системные сообщения;
- сообщения по каналам;
- сообщения, генерируемые с помощью системной переменной **@Message** (группа СИСТЕМНЫЕ);
- интерактивные сообщения оператора;
- другие виды сообщений.

Монитор генерирует predetermined сообщения первых трех видов в случае отсутствия соответствующих словарей в узле (см. **Классификация компонентов** и **Редактор словарей сообщений**).

Сообщение о событии заносится в отчет тревог в виде отдельной строки.

Сохранение сообщений в ОТ реализовано в виде отдельного потока с более низким приоритетом по сравнению с основным потоком (см. **Потоки монитора**). В реальном времени для управления передачей/приемом сообщений используются переменные **@Logging** и **@Net_DDE** (группа СИСТЕМНЫЕ).

Монитор формирует две очереди сообщений – из первой сообщения записываются в файл ОТ и передаются по направлению **G**, из второй передаются по всем остальным направлениям. По умолчанию размер очереди равен 64000 строк. Для индикации/изменения размеров очередей в реальном времени используется переменная **@q_Queue_Alarms** (группа ДИАГНОСТИКА).

Если интенсивность генерации сообщений превышает скорость их выборки из очереди, очередь начинает расти. При достижении предельного размера очереди новые сообщения записываются поверх самых старых.

Сообщение теряется, если по каким-либо причинам его не удалось вставить в очередь. Число потерянных таким образом сообщений для первой очереди индицирует переменная **@q_Lost_Alarms** (группа ДИАГНОСТИКА).

Максимальный размер файла ОТ ограничивается максимальным поддерживаемым размером файла в файловой системе. По достижении предель-

ного размера файла, новые сообщения начинают записываться с начала ОТ. Для управления размером файла ОТ в реальном времени используется переменная **@AR_Length** (группа СИСТЕМНЫЕ).

Предельно допустимое число строк в ОТ задается при его конфигурировании (см. **Задание параметров узла**). При переполнении новые сообщения записываются с начала ОТ (со второй строки).

Для диагностики ОТ предусмотрена переменная **@e_Alarm_Report** (группа ДИАГНОСТИКА), для создания резервной копии файла – **@Copy_AR** (группа СИСТЕМНЫЕ) – см. **Системные переменные TRACE MODE 6**.

К особенностям отчета тревог консолей относятся следующие:

- из консоли нельзя квитировать сообщение удаленного ОТ.

ОТ конфигурируются с помощью ключей в файле *.cnf (см. **Отчет тревог** в разделе **Задание параметров работы мониторов**).

Формат строки ОТ

Строка сообщения в отчете тревог содержит следующие предопределенные поля, разделенные пробелами:

Date Time Category Name Coding Text UserID T_ack N

где

- **Date Time** – поле вывода даты и времени в форматах, заданных для ОТ в настройках узла (см. **Задание параметров узла**). В зависимости от формата, под значение отводится соответствующее число знакомест. Отображаемое значение зависит от наличия/отсутствия ключа **ALR_FRMT_MSM=BOTH** в файле *.cnf:
 - если ключ не задан, в поле отображается дата и время события по часам данного узла (**T_{node}**), и события в ОТ отсортированы по **T_{node}**;
 - если ключ задан, в поле отображается значение атрибутов (45, **T**) и (88, **ms**) канала, при этом события в ОТ отсортированы тоже по **T_{node}**;
- **Category** – категория сообщения, 1 знакоместо (см. **Редактор словарей сообщений**);
- **Name** – для сообщения по каналу – имя канала; для системного сообщения, не связанного с каналом, – **<имя файла prj без расширения>_<порядковый номер узла>**. При вводе пользовательского комментария в это поле записывается имя соответствующего канала класса **Пользователь**. На поле **Name** отводится 32 знакоместа;

- **Coding** – кодировка и время изменения канала (31 знакоместо).
Для управления выводом времени предусмотрены следующие ключи в файле *.cnf:
 - **ALR_FRMT_MSM=1 .. 5** – формат времени изменения:
 - 1 – дата и время;
 - 2 – время;
 - 3 – время и миллисекунды (3 цифры);
 - 4 – время и миллисекунды (1 цифра);
 - 5 – **mm:ss.ms** (3 цифры);
 - **ALR_FRMT_MSM=OFF** – отключение вывода времени;
 - **ALR_FRMT_MSM=ON** – пишется время для каналов с установленным флагом **Запрос времени значения** (атрибут 50);
 - **ALR_FRMT_MSM=STD** – пишется время для каналов с установленным флагом **Запрос времени значения** или обработкой в цикле FAST;
- **Text** – текст сообщения или пользовательский комментарий (48 знакомест);
- **UserID** – идентификатор пользователя, квитирававшего сообщение (5 знакомест). Сообщения могут квитироваться с помощью графических элементов;

Квитирование – это подтверждение того, что оператор видел данное сообщение.

- **T_ack** – время квитиования сообщения в формате **DD_НН:ММ:СС**;
- **N** – порядковый номер строки в ОТ в формате HEX (8 символов).

В последние три поля первой строки отчета тревог во всех случаях записываются служебные значения.

Если для канала определена функция записи в отчет тревог информации об изменении атрибутов (дополнительно установлен флаг **Атрибуты**), то сообщение об изменении атрибута содержит в поле **Text** выражение **<ATTR>=<NEW_VAL>**, где **ATTR** – короткое имя атрибута, **NEW_VAL** – его новое значение.

Другие ключи:

- при наличии следующих ключей для каналов соответствующего типа вместе с сообщением автоматически выводится значение канала:
 - **ALR_HEX_ADDVALUE=ON** – для HEX16/HEX32;
 - **ALR_FLOAT_ADDVALUE=ON** – для FLOAT/DOUBLE

- FLOAT;
- **ALR_DEC_ADDVALUE=ON** – для HEX16/HEX32 с представлением DEC;
- **ALR_ADDERROR=ON/OFF**:
 - **OFF** – при недостоверности выводится системное сообщение о недостоверности (по умолчанию);
 - **ON** – категория сообщения становится "?" при недостоверности;
- **ALR_DIMENSION=ON** – для FLOAT/DOUBLE FLOAT вместе со значением выводится размерность канала (если задана);
- **ALR_CEVTWOTIME=ON** – вывод в сообщение времени с миллисекундами на события ON и OFF канала EVENT;
- **ALR_ORDER_BY=DEF** –
- **ALR_ORDER_BY=TIME** –

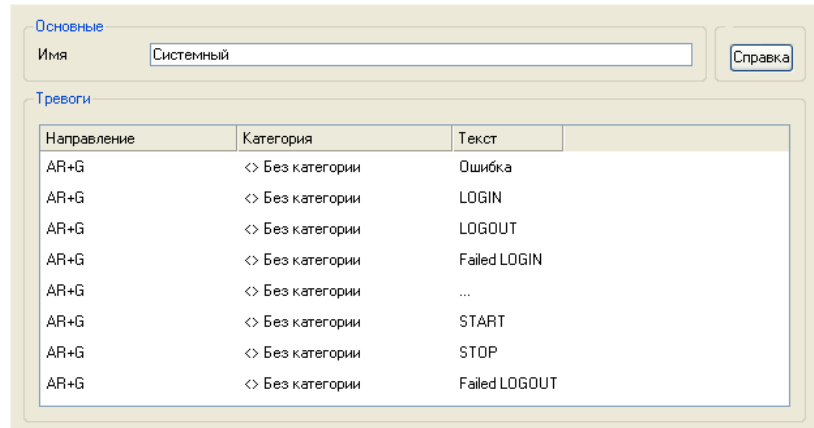
Системные сообщения

К системным относятся следующие события (слева указан текст сообщения, генерируемый по умолчанию при возникновении события):

- события, связанные с каналами:
 - **Error** – установка какому-либо каналу признака аппаратной недостоверности;
 - **Login** – корректная авторизация при старте;
 - **Logout** – корректное окончание сеанса пользователем;
 - **Failed Login** – некорректная авторизация при старте;
 - **Failed Logout** – некорректная авторизация при окончании сеанса пользователем;
- события, не связанные с каналами:
 - **START** – запуск монитора;
 - **CONTINUE** – невозможность выполнения файловой операции – например, при удалении или переименовании файла ОТ в реальном времени. В этой ситуации монитор создает новый файл ОТ с именем, заданным в настройках узла, и записывает сообщение CONTINUE в первой строке этого файла. Данное сообщение отсутствует в системном словаре, оно относится к специальной категории системных сообщений (эта категория обозначается знаком звездочки – *);
 - **STOP** – останов монитора;
- прочие события:
 - ... – неопределенное событие. К таким событиям относятся, в том числе, некорректные действия с атрибутами каналов. Например, сообщение «...» заносится в ОТ при попытке

непосредственно изменить значение вычисляемого атрибута (7, P) канала **Событие**.

Для использования сообщений из системного словаря в узле должен быть создан компонент **Словарь системный**. Редактор этого словаря (см. **Редактор словарей сообщений**) показан на рисунке:



Сообщения по каналам

Сообщение, сгенерированное монитором по каналу, записывается в атрибут 254, **RST** этого канала (см. **Атрибуты каналов, отображаемые профайлером**).

Сообщения генерируются по результатам анализа атрибута (0, **R**) канала (для большинства классов каналов этот атрибут имеет название **Реальное значение**), при этом критерии генерации различны для каналов различных классов, а для каналов **FLOAT** и **DOUBLE FLOAT**, кроме того, зависят от настроек канала.

Если для канала определена функция записи в отчет тревог и, помимо флага **Отчет тревог**, для него установлен флаг **Атрибуты**, в ОТ дополнительно заносятся сообщения об изменении некоторых атрибутов из числа тех, которые не вычисляются в канале. Сообщения об изменении атрибутов не могут быть заданы в словарях, они имеют вид **<короткое имя атрибута> = <новое значение атрибута>** и категорию **R** (см. **Редактор словарей сообщений**).

Для привязки словаря используется окно свойств канала (см. **Вкладка 'Информация'**) или его редактор (см. **Общие атрибуты каналов**).

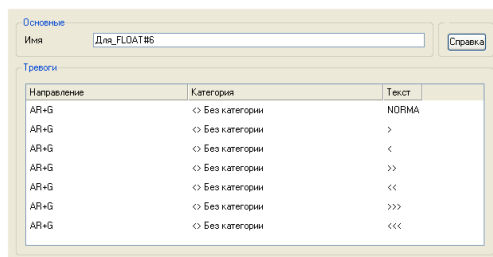
Сообщения по каналам **FLOAT** и **DOUBLE FLOAT**

Сообщения по результатам анализа реального значения каналов этих классов генерируются по следующим критериям:

- канал не связан со словарем:
 - анализ границ разрешен (**BNDR=0** – см. **Канал класса **FLOAT** и Канал класса **DOUBLE FLOAT****), границы не заданы – сообщения в ОТ не выводятся;
 - анализ границ разрешен, границы заданы – если вследствие изменения реального значения канала изменяется номер интервала (**P**), в ОТ выводится сообщение вида «=**<число>**», где **<число>** – реальное значение канала;
 - анализ границ запрещен (**BNDR=1**) – при каждом изменении реального значения канала в ОТ выводится сообщение вида «=**<число>**», где **<число>** – реальное значение канала;
- канал связан со словарем:
 - анализ границ запрещен – при каждом изменении реального значения канала в ОТ выводится сообщение для интервала 0 («**NORMA**» по умолчанию);
 - анализ границ разрешен – если вследствие изменения реального значения канала изменяется номер интервала (**P**), в ОТ из словаря выводится сообщение с порядковым номером,

равным номеру интервала.

Редактор словаря для канала FLOAT показан на рисунке:



Для каналов FLOAT и DOUBLE FLOAT в ОТ могут быть внесены сообщения об изменении границ, периода, состояния и настроек первичной обработки.

Чтобы после рестарта МРВ избежать повторной записи в ОТ сообщения по каналу, информация о котором сохраняется в дампе, нужно в этом канале установить флаг **Отработать**.

Сообщения по каналам HEX16 и HEX32

Сообщения по результатам анализа реального значения каналов этих классов генерируются по следующим критериям.

Если канал не связан со словарем, сообщение генерируется при каждом изменении реального значения канала. В качестве текста сообщения используется выражение вида **=<число>**, где **<число>** – величина реального значения канала (DEC).

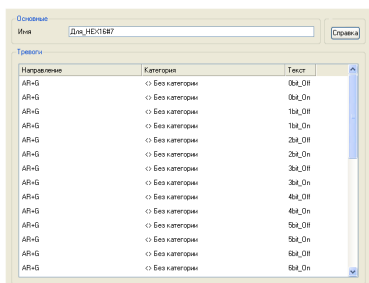
Если канал HEX32 связан со словарем, сообщение генерируется при изменении любого бита реального значения. В этом случае при каждом изменении реального значения в общем случае генерируется несколько сообщений – их количество определяется числом битов, изменивших свое значение. В словарях предусмотрены различные сообщения для случаев изменения бита с 0 на 1 и с 1 на 0.

Если канал HEX16 связан со словарем, алгоритм генерации сообщений зависит от вида представления канала (флага **DEC** (84, **HD**) – см. **Канал класса HEX16**):

- если **HD=0** (вид представления HEX), сообщение генерируется при изменении любого бита реального значения;
- если **HD=1** (вид представления DEC), сообщение генерируется при каждом изменении реального значения канала. Текст сообщения зависит от нового значения канала:
 - при **R=0...31** в ОТ выводится сообщение из словаря с соответствующим номером;
 - при других значениях **R** в ОТ выводится выражение вида

=<число>, где <число> – величина реального значения канала (DEC).

Редакторы словарей для каналов этих классов однотипны:



Для каналов HEX16 и HEX32 в ОТ могут быть внесены сообщения об изменении периода и состояния.

Сообщения по каналу TIME

Монитор генерирует сообщение при каждом изменении реального значения канала этого класса. В качестве текста сообщения в этом случае используется значение даты и времени (атрибут 0, **R**) в определенном для канала формате (атрибут 7, **P**).

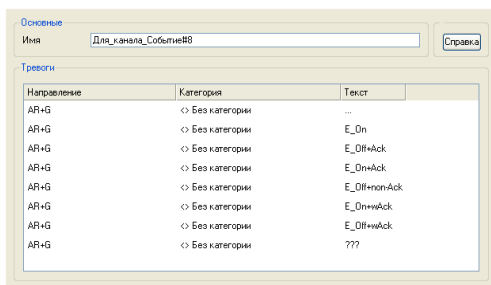
Для каналов TIME в ОТ могут быть внесены сообщения об изменении периода и состояния.

Для канала класса TIME словарь сообщений не предусмотрен.

Сообщения по каналу СОБЫТИЕ

Монитор генерирует сообщение при каждом изменении реального значения канала этого класса (атрибут 0, **R** индицирует статус события).

Редактор словаря для канала **Событие** показан на рисунке:



Тексты сообщений могут быть также изменены с помощью ключей в файле *.cnf (см. **Задание параметров работы мониторов**).

Для каналов **Событие** в ОТ могут быть внесены сообщения об измене-

нии периода и состояния.

Сообщения по каналам ПЕРСОНАЛ и ПОЛЬЗОВАТЕЛЬ

Монитор генерирует сообщение при каждом изменении реального значения каналов этих классов (атрибут 0, **R** индицирует статус работника).

Для каналов **Персонал** и **Пользователь** используется общий словарь – **Словарь сообщений для персонала**, редактор которого показан на рисунке:

| Направление | Категория | Текст |
|-------------|------------------|----------------|
| AR+G | <> Без категории | Неизвестно |
| AR+G | <> Без категории | Работа |
| AR+G | <> Без категории | Проставляет |
| AR+G | <> Без категории | Свободен |
| AR+G | <> Без категории | Болеет |
| AR+G | <> Без категории | Прогул |
| AR+G | <> Без категории | Начал работать |
| AR+G | <> Без категории | Отсутствует |

Для каналов **Персонал** и **Пользователь** в ОТ могут быть внесены сообщения об изменении периода и состояния.

Сообщения по каналу ЕДИНИЦА ОБОРУДОВАНИЯ

Монитор генерирует сообщение при каждом изменении реального значения канала этого класса (атрибут 0, **R** индицирует статус единицы оборудования).

Редактор словаря для канала **Единица оборудования** показан на рисунке:

| Направление | Категория | Текст |
|-------------|------------------|-----------------|
| AR+G | <> Без категории | Нет данных |
| AR+G | <> Без категории | Работа |
| AR+G | <> Без категории | Проставляет |
| AR+G | <> Без категории | Резерв |
| AR+G | <> Без категории | Ошибка |
| AR+G | <> Без категории | На обслуживании |
| AR+G | <> Без категории | Ремонт |
| AR+G | <> Без категории | Выключено |

Для каналов этого класса в ОТ могут быть внесены сообщения об изменении периода и состояния.

Сообщения по каналу M-РЕСУРС

Монитор генерирует сообщения по этому каналу в следующих ситуациях:

- при изменении количества ресурса (атрибута **000, R** – см. **Атрибуты канала M-РЕСУРС**). В качестве текста сообщения используется:
 - выражение вида «**(+)** **<In>** **<Price_In** или **Def_Price** **>** **<service_var>**» – в случае прихода ресурса;
 - выражение вида «**(-)** **<Cost_Out>** **<Q>** **<service_var>**» – в случае выдачи ресурса;
- при переходе из одного интервала в другой (для канала M-РЕСУРС могут быть заданы границы **LA** и **HA**, которые разбивают весь диапазон значения атрибута **R** канала на 3 интервала). В этом случае в качестве текста сообщения используется выражение «**=R**»,
- при отклонении запроса на выдачу ресурса. В этом случае в поле **Text** сообщения выводится значение, поданное в атрибут **Q** канала.

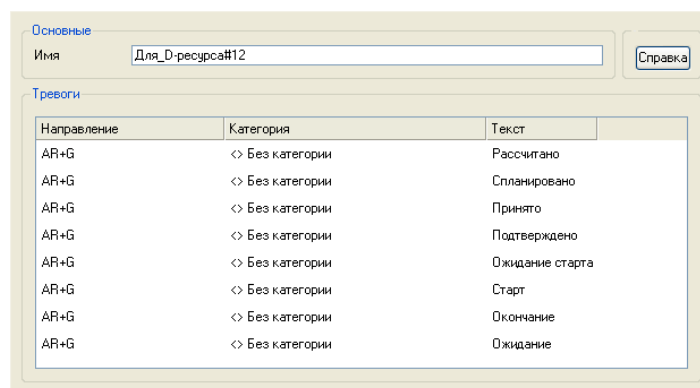
Пользовательские сообщения по событиям канала могут быть заданы в соответствующем словаре, редактор которого показан на рисунке:

| Направление | Категория | Текст |
|-------------|------------------|-------------------------|
| AR+G | <> Без категории | Изменилось значение |
| AR+G | <> Без категории | Запрос - Выдача |
| AR+G | <> Без категории | Запрос - Нет количества |
| AR+G | <> Без категории | Норма |
| AR+G | <> Без категории | >> |
| AR+G | <> Без категории | << |

Для каналов этого класса в ОТ могут быть внесены сообщения об изменении периода и состояния.

Сообщения по каналу D-РЕСУРС

Редактор словаря для канала **D-Ресурс** показан на рисунке:

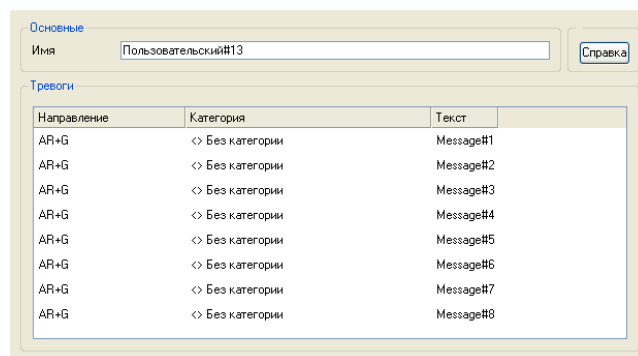


Другие виды сообщений

Генерация сообщений с помощью переменной MESSAGE

При изменении выходного значения ((9,Q)=1...8) канала типа OUTPUT с установленным флагом **Отчет тревог**, связанного с системной переменной **@Message** типа OUTPUT и словарем **Пользовательский словарь сообщений**, генерируется словарное сообщение с соответствующим порядковым номером (1...8).

Редактор пользовательского словаря сообщений показан на рисунке:



Запись в отчет тревог сообщений оператора

Для записи в ОТ произвольного сообщения (комментария) оператора используется любой графический элемент с настроенной функцией **Послать комментарий** (см. **Функция ввода комментария**), а также команда **Ввести комментарий** меню **Действия** MPB (см. **Профайлер с поддержкой графических экранов**).

Пользовательский комментарий записывается в ОТ отдельной строкой (см. **Формат строки ОТ**).

Запись аргументов CALL.STRING в отчет тревог

В ОТ могут быть записаны значения аргументов канала CALL.STRING (см. **Запись длинных строк в канал CALL**).

Архивы SIAD

Мониторы TRACE MODE поддерживают функцию записи значений атрибутов каналов в архивы SIAD (см. **Архивирование каналов узла**). Эти архивы конфигурируются для узла (см. **Задание параметров узла**, а также **Задание параметров работы мониторов**).

Разрешение на архивирование и набор архивируемых атрибутов задаются для канала при его конфигурировании в редакторе или окне свойств (см. **Общие атрибуты каналов** и **Вкладка 'Флаги'**).

Если для канала задано архивирование, а флаг **Атрибуты** не установлен, в архив записываются сообщения об изменении атрибута 0, **R**.

Если для канала задано архивирование и установлен флаг **Атрибуты**, то, помимо записи сообщений об изменении атрибута 0, **R**, в архив записываются сообщения об изменении границ и атрибутов обработки канала.

Архивирование в регистратор описано в разделе **Архивирование в регистратор**.

Каналы могут архивироваться принудительно – см. описание атрибута (6, **D**) в разделе **Атрибуты каналов, отображаемые профайлером** и раздел **Канал CALL.Writer**.

Каждое сообщение содержит идентификаторы канала и атрибута, новое значение атрибута и время его изменения, а также некоторые другие параметры.

Сохранение сообщений в архив реализовано в виде отдельного потока с более низким приоритетом по сравнению с основным потоком (см. **Потоки монитора**).

Для уменьшения времени операций с архивом файл кэшируется.

Для управления архивированием, мониторинга состояния архивов и выполнения операций с архивами в реальном времени используются меню МРВ (см. **Профайлер с поддержкой графических экранов** и **Профайлер без поддержки графических экранов**) и следующие системные переменные (см. **Системные переменные TRACE MODE 6**):

- группа СИСТЕМНЫЕ:
 - **@Copy_SIAD**
 - **@Data_from_SIAD**
 - **@Logging**
- группа ДИАГНОСТИКА:

- @e_SIAD
- @q_SIAD_Lost
- @q_SIAD_Q

Данные из SIAD могут быть отображены на графическом экране, а также выведены в генерируемый документ. Если у канала CALL вызова шаблона установлен флаг **Регистратор** (см. **Общие атрибуты каналов**), архивные данные по каналу, привязанному к его аргументу, запрашиваются у регистратора; в противном случае – у узла, к которому канал принадлежит.

Уменьшение объема SIAD при записи с нарушением порядка временных меток

Если при записи в SIAD нарушается порядок временных меток, МРВ может создавать до $n < 1024$ очередей записи, каждая из которых характеризуется собственной минутной меткой времени. Значение n задается с помощью ключа **QSIAD** = $\langle n \rangle$ в файле *.cnf. Для записи используются также следующие ключи в файле *.cnf:

- **TSIAD** = $\langle k \text{ минут} \rangle$ – шаг между метками времени очередей;
- **CSIAD** = $\langle t \text{ секунд} \rangle$ – период проверки каждой очереди на заполнение блока для передачи в буфер записи.

Уменьшение записей канала FLOAT в SIAD

Для уменьшения числа записей канала FLOAT в SIAD и SIAD регистратора (а также, если флаг **Регистратор** используется для сервера МЭК-104 – см. **МРВ как сервер протокола МЭК 60870-104**) могут использоваться атрибуты **ВГ** и **НГ** (см. **Канал класса FLOAT**), при этом у таких каналов сокращается число интервалов (1 и 2 отсутствуют – см. **Границы и интервалы канала FLOAT**).

Для DOUBLE FLOAT интервал равен 0 и не анализируется (поскольку в канале только 2 границы – см. **Канал класса DOUBLE FLOAT**).

Если реальное значение канала $R \geq \text{ВГ}$ или $R \leq \text{НГ}$, то R передается в архив и регистратор, и меняются пороги передачи:

$$\text{ВГ} = R + (\text{ВГ} - \text{НГ}) / 2$$

$$\text{НГ} = R - (\text{ВГ} - \text{НГ}) / 2$$

Для задания такого режима работы канал в ИС нужно сконфигурировать следующим образом:

- у канала отсутствует автопосылка (не установлен флаг **Автопосылка.Включить** – см. **Общие атрибуты каналов**);

- **ВГ<НГ** (в MPB неравенство изменится на обратное) – для FLOAT и DOUBLE FLOAT, или для FLOAT может быть задан гистерезис **Hyst<0** (в MPB **Hyst** меняет знак и используется как гистерезис).

Выборка и обработка данных SIAD

Для выборки/обработки данных из архива в реальном времени используются каналы класса **CALL** с **Параметр**>=32 и соответствующими типами вызова (каналы-инициаторы выборки, см. **Атрибуты канала класса CALL**). Тип пересчета таких каналов должен быть **цикл IDLE**.

Различные каналы CALL выборки из SIAD могут выполнять одинаковые функции – например, для вычисления интервальных средневзвешенных значений по данным локального архива можно использовать как CALL.LocalQuick, так и CALL.LocalList.

Если в описании канала-инициатора не указан другой порядок отработки, то:

- канал типа OUTPUT обрабатывается при изменении значения;
- канал типа INPUT обрабатывается **R** раз со своим периодом (**R** – значение атрибута 0).

В канале-инициаторе по завершении выборки **R=0** автоматически.

В случае ошибки соответствующее сообщение записывается в файл **tm6_log.txt** (если установлен бит 11 (0x800) переменной **@Debug**, информация об ошибке записывается также в протокол профайлера), канал-инициатор сбрасывается в исходное состояние, в нем устанавливается признак аппаратной недостоверности и обнуляется входное значение, а код ошибки записывается в атрибут (240, **ERR**):

- 1 – системная ошибка;
- 2 – выборка данных не завершена в течение 2 минут;
- 4 – недостаточно памяти;
- 5 – ошибка экспорта (например, при генерации документа; сообщение – «EXPORT: WRONG»);
- 6 – неправильный параметр;
- 7 – в архиве нет запрашиваемых данных либо заданный интервал находится вне временного диапазона архива.

Если архив не обнаружен, монитор выключает канал-инициатор.

В ряде случаев результаты выборки/обработки записываются в аргументы канала-инициатора, при этом некоторые аргументы возвращают два пара-

метра – значение и время. Если к такому аргументу привязан канал, то значение записывается в его вход, а время – в атрибут **Время изменения**.

Атрибут 120, **АСК** индицирует состояние процесса:

- в каналах CALL.TVC/CALL.ChGroupReq, привязанных к аргументам каналов-инициаторов, **АСК** принимает значение 1 по завершении приема данных;
- в каналах-инициаторах **АСК** принимает значение 1 после извлечения всех данных (по всем интервалам и выбираемым параметрам);
- в каналах-инициаторах и каналах CALL.TVC/CALL.ChGroupReq, привязанных к аргументам каналов-инициаторов, **АСК** принудительно обнуляется перед началом выборки.

В каналах-инициаторах и каналах CALL.TVC/CALL.ChGroupReq, привязанных к аргументам каналов-инициаторов, фиксируется время присвоения 1 атрибуту 120, **АСК** (метка времени записывается в атрибут 45, **T**).

В аргументы канала CALL.TVC, в которые записываются результаты выборки/обработки архивных значений, записывается также признаки «достоверность», «состояние» и «подключение» этих значений.

Ключ **TVC_RSZ=<n>** в файле *.cnf задает максимальное число аргументов каналов CALL.TVC/CALL.ChGroupReq, привязанных к аргументам каналов-инициаторов (1024 по умолчанию).

При извлечении срезов из удаленного архива с редкими записями в архивные таблицы экрана или документа может возникнуть ситуация, когда записей какого-либо канала в некоторых интервалах нет. В этом случае данные в таблице могут отображаться некорректно. Чтобы этого избежать, нужно делать принудительные записи с периодом, соизмеримым с интервалом разбиения таблицы.

Экспорт архивных данных

Существует несколько способов экспорта архивных данных (см. **Профайлер с поддержкой графических экранов**, **Профайлер без поддержки графических экранов**, **Конфигурирование таблицы архивных значений**, **Канал CALL.ChGroupReq** и описание переменной **@Data_from_SIAD** в разделе **Группа СИСТЕМНЫЕ**).

Временной интервал выборки

Диапазон (T_FROM, T_TO) задают **arg0** и **arg1** (тип данных этих аргументов должен быть DATE_AND_TIME).

MPB автоматически устанавливает бит 5 (0x20) атрибута **Параметр** канала CALL выборки данных из архива, если

тип данных **arg0** и **arg1** этого канала – DATE_AND_TIME.

Существуют следующие зарезервированные значения **arg0** и **arg1** с типом данных DATE_AND_TIME (если эти аргументы не имеют привязки):

- **arg0:**
 - 0 – **T_FROM**=<**T_TO** – 1 минута>;
 - 1 – **T_FROM**=<начало часа, заданного **T_TO**>;
 - 2 – **T_FROM**=<**T_TO** – 1 час>;
 - 3 – **T_FROM**=<начало текущего дня>;
 - 4 – **T_FROM**=<**T_TO** – 1 день>;
 - 5 – **T_FROM**=<**T_TO** – 2 дня>;
 - 6 – **T_FROM**=<**T_TO** – 3 дня>;
 - 7 – **T_FROM**=<**T_TO** – 4 дня>;
 - 8 – **T_FROM**=<**T_TO** – 5 дней>;
 - 9 – **T_FROM**=<**T_TO** – 1 неделя>;
 - 10 – **T_FROM**=<начало текущего месяца>;
 - 11 – **T_FROM**=<начало предыдущего месяца>;
 - 12 – **T_FROM**=<2 месяца назад от начала текущего месяца>;
 - 13 – **T_FROM**=<1 квартал назад от текущего времени, выравнивание на начало месяца>;
 - 14 – **T_FROM**=<**T_TO** – 3 месяца, выравнивание на начало месяца>;
 - 15 – **T_FROM**=<**T_TO** – 1 месяц, выравнивание на начало месяца>;
 - 16 – **T_FROM**=<начало текущего года>;
 - 17 – **T_FROM**=<начало предыдущего года>;
 - 32 – если в проекте нет пользователей, **T_FROM**=<время запуска МРВ>, в противном случае **T_FROM**=<начало текущей сессии>;
 - 33 – **T_FROM**=<начало предыдущей сессии>;
 - 34..45 – **T_FROM**=<начало соответствующего месяца> (34 – январь);
- **arg1:**
 - 0 – **T_TO**=<начало текущей минуты>;
 - 1 – **T_TO**=<начало текущего часа>;
 - 2 – **T_TO**=<начало текущего дня>;
 - 3 – **T_TO**=<начало предыдущего часа>;
 - 4 – **T_TO**=<начало предыдущего дня>;
 - 6 – **T_TO**=<начало текущей недели> (недели отсчитываются с начала года);

- 8 – **T_TO**=<начало предыдущей недели (понедельник)>;
- 9 – **T_TO**=<начало текущей недели>;
- 10 – **T_TO**=<начало текущего месяца>;
- 11 – **T_TO**=<начало предыдущего месяца>;
- 12 – **T_TO**=<2 месяца назад от начала текущего месяца>;
- 13 – **T_TO**=<начало текущего квартала>;
- 16 – **T_TO**=<начало текущего года>;
- 17 – **T_TO**=<начало предыдущего года>;
- 33 – **T_TO**=<окончание предыдущей сессии>;
- 34...45 – **T_TO**=<начало соответствующего месяца> (34 – январь).

С помощью зарезервированных значений **arg0** и **arg1** для **T_FROM** и **T_TO** нельзя установить будущее время относительно текущего.

Отображение значений **arg0** и **arg1** зависит от флага **Запрос времени значения** канала:

- если флаг не установлен, при успешной выборке данных предустановленные значения заменяются абсолютными временными метками;
- если флаг установлен, при успешной выборке данных предустановленные значения заменяются абсолютными временными метками, которые вновь заменяются предустановленными значениями спустя приблизительно 30с.

*Разбиение (**T_FROM**, **T_TO**) на интервалы*

Значение младшего полубайта атрибута **Параметр** канала-инициатора выборки задает величину интервалов, на которые разбивается диапазон (**T_FROM**, **T_TO**) (соответственно, число интервалов равно результату деления диапазона на интервал):

- 0 – нет разбиения на интервалы;
- 1 – 60с;
- 2 – 1ч;
- 3 – 5мин.;
- 4 – 15мин.;
- 5 – 20мин.;
- 6 – 30мин.;
- 7 – 2ч;
- 8 – 4ч;

- 9 – 6ч;
- 10 – 8ч;
- 11 – 12ч;
- 12 – 24ч;
- 13 – 1 неделя (недели отсчитываются с начала года);
- 14 – 1 декада (декады отсчитываются с начала месяца);
- 15 – 1 месяц (число дней в месяцах соответствует календарным значениям).

Если интервал меньше 1ч, границы диапазона выборки должны совпадать с разбиениями часа на интервалы. Например, в случае 20-минутного интервала минуты T_FROM/T_TO могут быть только следующими: 0 мин., 20 мин., 40 мин.

Обработка данных локального архива по каналу

Для статистической обработки данных локального архива по заданному каналу в диапазоне (T_FROM, T_TO) используются каналы CALL с типом вызова (29) **LocalStatistic** (см. **Временной интервал выборки и Выборка и обработка данных SIAD**).

Параметр=32

Конфигурация CALL.LocalStatistic:

- канал, чьи архивные значения должны быть обработаны, должен быть привязан к CALL.LocalStatistic;
- значение младшего полубайта атрибута **Параметр** канала CALL.LocalStatistic задает величину интервалов, на которые разбивается диапазон (T_FROM, T_TO) (см. **Временной интервал выборки**). Если задан единственный интервал, статистические параметры записываются в аргументы CALL.LocalStatistic;
- начиная с **arg2**, к аргументам CALL.LocalStatistic привязываются каналы CALL.ChGroupReq или CALL.TVC, в аргументы которых записываются интервальные статистики.

Вычисляемые статистические параметры:

- **Arg_002** – первое значение, время;
- **Arg_003** – минимальное значение, время;
- **Arg_004** – максимальное значение, время;
- **Arg_005** – сумма всех достоверных значений;
- **Arg_006** – сумма всех значений;
- **Arg_007** – суммарное время недостоверности значения канала;
- **Arg_008** – суммарное время достоверности значения канала;
- **Arg_009** – интеграл:

$$I = \sum_{i=1}^n \frac{VAL_i + VAL_{i-1}}{2} (t_i - t_{i-1}).$$

где $VAL_0 \dots VAL_n$ – значения, $t_0 \dots t_n$ – время значений;

Здесь и далее разность времен – это число секунд.

- **Arg_010** – среднеквадратическое отклонение:

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=0}^n (VAL_i - M)^2}$$

где $VAL_0 \dots VAL_n$ и M – значения и их среднее;

- **Arg_011** – среднее значение:

$$M = \frac{1}{n+1} \sum_{i=0}^n VAL_i$$

- **Arg_012 – Arg_018** – суммарное время пребывания значения в интервале:
 - **Arg_012** – между LW и HW (интервал 0);
 - **Arg_013** – между HW и HA (1);
 - **Arg_014** – между LA и LW (2);
 - **Arg_015** – между HA и HL (3);
 - **Arg_016** – между LL и LA (4);
 - **Arg_017** – больше HL (5);
 - **Arg_018** – меньше LL (6);
- **Arg_019** – диапазон, реально обнаруженный в архиве на основании заданного (T_FROM , T_TO);
- **Arg_020** – суммарное время пребывания канала в выключенном состоянии;
- **Arg_021** – максимальное значение скорости возрастания значения, время. Значение первой производной в точке оценивается по формуле:

$$D_k = \frac{VAL_k - VAL_{k-1}}{t_k - t_{k-1}}$$
- **Arg_022** – максимальное значение скорости убывания значения, время;
- **Arg_023** – второе по величине минимальное значение, время;
- **Arg_024** – второе по величине максимальное значение, время;
- **Arg_025** – число переходов значения через 0, время последнего перехода;
- **Arg_026** – последнее значение, время;

- **Arg_027** – время первого значения;
- **Arg_028** – время последнего значения;
- **Arg_029** – время максимального значения;
- **Arg_030** – время минимального значения;
- **Arg_031** – **T_FROM**;
- **Arg_032** – **T_TO**;
- **Arg_033** –
- **Arg_034** – **Arg_065** – время, в течение которого значение равно соответственно 0...31.

Быстрая выборка данных из локального архива

Для быстрой выборки данных локального архива в диапазоне (**T_FROM**, **T_TO**) используются каналы **CALL** с типом вызова (30) **LocalQuick** (см. **Временной интервал выборки** и **Выборка и обработка данных SIAD**).

Параметр >=32

Канал **CALL.LocalQuick** производит выборку из локального архива, в ряде случаев – с разбиением диапазона (**T_FROM**, **T_TO**) на интервалы.

Начиная с **arg2**, к аргументам **INPUT** канала **CALL.LocalQuick** привязываются каналы **CALL.TVC** или **CALL.ChGroupReq**, в аргументы которых записываются результаты выборки.

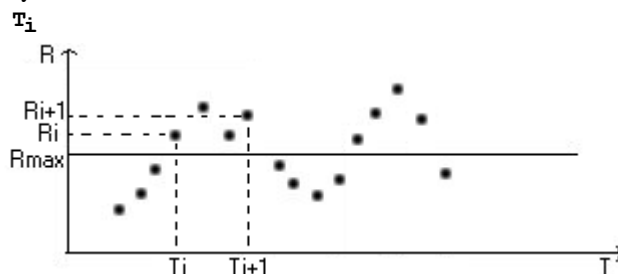
Если требуется выборка по одному каналу, этот канал привязывается к **CALL.LocalQuick**. Если требуется выборка по нескольким каналам, они привязываются к каналам **CALL.TVC** или **CALL.ChGroupReq**, привязанным к аргументам **CALL.LocalQuick** (в этом случае сам канал **CALL.LocalQuick** не должен иметь привязки).

Атрибут **CALL.LocalQuick.A** задает номер первого обрабатываемого **CALL.TVC/CALL.ChGroupReq** (нумерация каналов начинается с 0). Атрибут **CALL.LocalQuick.In** задает количество обрабатываемых **CALL.TVC/CALL.ChGroupReq** (одновременно **CALL.LocalQuick.In** < 0 является командой отработки канала **CALL.LocalQuick**).

Алгоритм выборки зависит от привязанного атрибута **CALL.TVC/CALL.ChGroupReq**:

Выбор архивного значения по условию изменения атрибута (**7, P**): из диапазона (**T_FROM**, **T_TO**) в канал **CALL.TVC / CALL.ChGroupReq** записывается первое по времени архивное значение, при котором **P** < 0, а также все последующие архивные значения вне зависимости от условия **P** < 0.

- 0, **R** – разбиения на интервалы нет, извлекается каждое реальное значение, при котором изменился атрибут (3, **C**), (4, **I**) или (7, **P**);
- 1, **A** и 9, **Q** – разбиение на интервалы по общему закону, вычисляется средневзвешенное (в случае 1, **A**) или среднее арифметическое (в случае 9, **Q**), а также максимальное и минимальное значение в каждом интервале (для записи всех параметров в CALL.TVC нужно сконфигурировать 3 кривых);
- 2, **In** – разбиение на интервалы по общему закону, извлекается каждое реальное значение, при котором изменился атрибут (3, **C**), (4, **I**) или (7, **P**), а также если это значение является экстремумом на интервале (в этом случае извлекается также следующее реальное значение);
- 4, **I** – разбиения на интервалы нет, извлекается каждое реальное значение, при котором изменился атрибут (3, **C**) или (4, **I**);
- 5, **FRQ** – разбиение на интервалы по общему закону, вычисляется последнее значение в каждом интервале, а также разность между этим значением и последним значением в предыдущем интервале (в первой точке разность равна 0);
- 6, **D** – разбиения на интервалы нет, извлекаются все реальные значения;
- 7, **P** – разбиение на интервалы в соответствии со значением архивированного атрибута 7, **P**, в каждом интервале вычисляются R_i и $\int_{T_i}^{T_{i+1}} R(T)dT$ (см. рис.).



Эти параметры записываются в CALL.TVC с двумя кривыми (для канала FLOAT $R_{max} = HW$).

Если последний аргумент канала CALL.LocalQuick имеет тип OUTPUT, и к этому аргументу привязан канал FLOAT (или CALL.ChGroupReq), то в атрибуты канала FLOAT (аргументы канала CALL.ChGroupReq) записываются следующие параметры (если CALL.LocalQuick вычисляет соответствующие значения):

- среднее значение по всему диапазону – в (0, **R**) (в arg0 канала CALL.ChGroupReq);
- среднее значение интервальных максимумов – в (28, **HA**) (в arg1);

- среднее значение интервальных минимумов – в (29, **LA**) (в arg2);
- среднее значение разностей интервальных максимумов и минимумов – в (30, **HW**) (в arg3);
- среднеквадратическое отклонение по всему диапазону – в (31, **LW**) (в arg4).

Выборка данных из локального архива по каналу

Для выборки первых записей локального архива по заданному каналу из диапазона (T_FROM, T_TO) используются каналы CALL с типом вызова (31) **LocalList** (см. **Временной интервал выборки** и **Выборка и обработка данных SIAD**).

Атрибут 92, **I2** индицирует число архивных записей, помещенных в канал.

Параметр >=32

Канал CALL.LocalList извлекает архивные данные по заданному каналу в диапазоне (T_FROM, T_TO), разбивает диапазон на заданные интервалы (см. **Временной интервал выборки**) и вычисляет минимальные, максимальные и средневзвешенные значения в этих интервалах.

Если дополнительно установлен бит 4 (0x10) атрибута **Параметр**, значение, вычисленное по первому интервалу, вычитается из всех остальных значений.

Если требуется выборка по одному каналу, этот канал привязывается к CALL.LocalList. Если требуется выборка по нескольким каналам, они привязываются к каналам CALL.TVC или CALL.ChGroupReq, привязанным к аргументам CALL.LocalList (в этом случае сам канал CALL.LocalList не должен иметь привязки).

В первом случае:

- если разбиения на интервалы нет, вычисленные значения записываются в аргументы, начиная с **arg2** (создаются парами, первый аргумент пары должен иметь соответствующий числовой тип данных, второй – DATE_AND_TIME);
- если, начиная с **arg2**, аргументы канала LocalList имеют тип данных REAL, и к ним привязаны каналы ChGroupReq/TVC, вычисленные интервальные характеристики записываются в аргументы того канала ChGroupReq/TVC, который привязан к аргументу с номером (LocalList.A + 2) канала LocalList (см. **Запись вектора в CALL.ChGroupReq** и **CALL.TVC**). После отработки канала LocalList его атрибут 1, **A** обнуляется.

При записи интервальных характеристик в канал TVC: средневзвешенные значения образуют первую кривую, максимальные

значения – вторую кривую, минимальные значения – третью кривую (если соответствующие кривые сконфигурированы в канале).

Во втором случае:

- атрибут **LocalList.A** задает номер первого обрабатываемого TVC/ChGroupReq (нумерация каналов начинается с 0);
- атрибут **LocalList.In** задает количество обрабатываемых TVC/ChGroupReq (одновременно **LocalList.In** < 0 является командой отработки канала **LocalList**).

Расширенная быстрая выборка данных из локального архива

Для расширенной быстрой выборки данных локального архива из диапазона (T_FROM, T_TO) используются каналы CALL с типом вызова (32) **LocalQuick-T** (см. **Временной интервал выборки** и **Выборка и обработка данных SIAD**).

Конфигурация канала CALL.LocalQuick-T:

- **Параметр** & 0x20 = 1 (установлен бит 5);
- к каналу должен быть привязан канал CALL.ChGroupReq, CALL.TVC или CALL.AS_DATA (**CCC**), к которому должен быть привязан архивируемый канал (**ch**);
- аргументы:
 - **arg0** и **arg1** (DATE_AND_TIME) задают диапазон выборки (T_FROM, T_TO);
 - **arg2** – к этому аргументу привязывается канал CALL.ChGroupReq, к аргументам которого привязываются архивируемые каналы (**chch_k**);
 - **arg3** – число секунд;
 - к последующим аргументам привязываются каналы CALL.ChGroupReq (**CGR_m**).

Алгоритм отработки канала CALL.LocalQuick-T:

- извлекаются ((T_{ch})_i, **ch_i**) из (T_FROM, T_TO) и запоминаются в **CCC**;
- извлекаются срезы каналов **chch_k** из интервалов, формируемых в зависимости от значения полубайта 2 (0xF00) атрибута 0, **R** канала CALL.LocalQuick-T:
 - 0 – T_FROM_i = (T_{ch})_i – 60с, T_TO_i = (T_{ch})_i;
 - 1 – T_FROM_i = (T_{ch})_i, T_TO_i = <(T_{ch})_{i+1}, округленное до минут влево> – 1с;
 - 2 – T_FROM_i = <(T_{ch})_i, округленное до минут влево> – 1мин., T_TO_i = (T_{ch})_i;

- 3 – $T_FROM_i = (T_{ch})_i - \text{arg3}$, $T_TO_i = (T_{ch})_i$;
- срезы каналов **chch_k** обрабатываются в зависимости от значения полубайта 1 (0xF0) атрибута 0, **R** канала CALL.LocalQuick-T:
 - 0 – срезы каналов **chch_k** записываются в соответствующие по порядку **CGR_k**;
 - 1 – в **CGR_m** записываются поканальные суммы срезов каналов **chch_k** из интервалов, в которых $ch_i = \text{CGR}_m.A$;
 - 4 – аналог алгоритма 0x10, но если среди **CGR_m** нет каналов, для которых $A = ch_i$, то ищется канал **CGR_M** с $A = 0$, **CGR_M.A** присваивается значение ch_1 и далее выполняется алгоритм 0x10.

Для извлечения срезов каналов **chch_k** из произвольных интервалов можно отключить **ССС** от источника и задать его аргументы вручную.

Принудительная запись в SIAD

Канал CALL.Writer (см. **Выборка и обработка данных SIAD**) используется для принудительной записи в SIAD сообщений по архивируемым каналам. Сообщение может содержать, в том числе, произвольное значение и произвольную метку времени, а также признак ручной записи (таким признаком является наличие в сообщении флага 8, **W**=1).

Алгоритм работы канала CALL с типом вызова **Writer** зависит от его атрибута **Параметр**:

- **Параметр** = 0, 7, 1, 8, 2, 9 – запись сообщений по каналам, которые привязаны к аргументам канала **CALL.Writer** (если **Параметр** = 0, 1, 2, сообщения содержат признак ручной записи, если **Параметр** = 7, 8, 9 – не содержат):
 - **Параметр** = 0, 7 – сообщение по каналу содержит значение аргумента, к которому привязан этот канал, и метку времени, в качестве которой используется значение ближайшего аргумента DATE_AND_TIME, предшествующего аргументу с привязкой к каналу. Если значение такого аргумента DATE_AND_TIME меньше 1000с, сообщение не записывается в SIAD. Если аргумента DATE_AND_TIME нет, время записи равно времени изменения значения канала (**<канал>.45**), а в случае его отсутствия – времени отработки канала CALL (**<CALL.Writer>.45**);
 - **Параметр** = 2, 9 – аналог **Параметр** = 0, 7, но в качестве записываемого значения используется значение аргумента, следующего за аргументом с привязкой к каналу;
 - **Параметр** = 1, 8 – запись значений каналов в случае их аппаратной недостоверности. Метка времени для таких значений – **<канал>.45+1 секунда**;

- **Параметр** = 3, 10, 4, 11 – групповая запись в SIAD значений всех архивируемых каналов. Реальное значение канала **CALL.Writer** даст номер архива; нулевой аргумент (**arg0**, DATE_AND_TIME) – метку времени для записи; первый аргумент (**arg1**, UDINT) – условие записи сообщения по каналу (**arg0 – <канал>.T > arg1**):
 - **Параметр** = 3 – запись значений атрибутов 0, **R** каналов;
 - **Параметр** = 10 – запись значений атрибутов 9, **Q** каналов;
 - **Параметр** = 4, 11 – аналог **Параметр** = 3, 10; значение **arg0** (метка времени) присваивается атрибутам 45, **T** каналов.

Запись в SIAD производится при обработке канала CALL. Для отображения подобных архивных данных на тренде можно задать специальный стиль.

Аргумент, задающий метку времени для записи, не должен иметь привязки в канале **CALL.Writer**.

Дифференциальный срез локального архива

Для получения дифференциальных срезов локальных архивов используются каналы класса CALL с типом вызова (25) **DifSnap** (см. **Выборка и обработка данных SIAD**).

После успешного завершения выборки **CALL.DifSnap.In=0** и **CALL.DifSnap.b14=1** (см. описание атрибута (46, **QE**) в разделе **Атрибуты каналов, отображаемые профайлером**).

Канал *CALL.DifSnap*

Атрибут (92, **I2**) анализируется только в том случае, если значение младшего полубайта атрибута **Параметр** равно 0.

$$(92, I2) = 0$$

Байт 0 значения канала **DifSnap** задает номер архива, байт 1 – тип обработки, аргументы **arg0** и **arg1** – времена срезов.

Если **Параметр** = 0 и тип обработки = 0, то последующие пары аргументов используются следующим образом: к первому привязывается архивируемый канал, во второй записывается разность между значениями срезов по этому каналу (дифференциальный срез, **VAL_{arg1} – VAL_{arg0}**).

Если **Параметр** = 0 и тип обработки = 2, то последующие аргументы используются группами по 3: к первому привязывается архивируемый канал, во второй записывается срез по этому каналу на время **arg0**, в третий

– срез на время **arg1**.

Если **Параметр** = 0 и тип обработки = 4, то последующие аргументы используются группами по 4: к первому привязывается архивируемый канал, во второй записывается срез по каналу на время **arg0**, в третий – срез на время **arg1**, в четвертый – дифференциальный срез.

Если **Параметр** = 0, тип обработки = 1, 3, 5 или 7, и архивируемый канал привязан к аргументу с номером **k**, то срез на время **arg0** по этому каналу записывается в аргумент с номером **k**, увеличенным соответственно на 1, 2, 3 или 4.

$(92, I2) \diamond 0$

Конфигурация CALL.DifSnap:

- **arg0** и **arg1** задают диапазон (T_FROM, T_TO) (см. **Временной интервал выборки**);
 - **arg0** < 86400 указывает на величину диапазона выборки;
 - **arg1** < 86400 указывает смещение диапазона выборки от текущего времени;
 - если **arg0** и/или **arg1** не привязаны, то они возвращают соответственно реальные T_FROM и T_TO;
- атрибут (92, **I2**) задает число (<256) извлекаемых срезов в диапазоне (T_FROM, T_TO). Если установлен бит 5 (0x20) атрибута **Параметр**, то $I2 = (\text{arg1} - \text{arg0}) / \langle \text{длина интервала} \rangle$ (длина интервала – значение младшего полубайта атрибута **Параметр**);
- в реальном времени атрибут (91, **I1**) индицирует номер интервала (растет до **I2**, а потом сбрасывается в 0);
- байт 0 значения CALL.DifSnap задает номер архива;
- байт 1 значения CALL.DifSnap задает тип обработки срезов:
 - 9 – без обработки;
 - 11 – первый срез вычитается из всех остальных;
 - 13 – из каждого среза вычитается предыдущий;
 - 15 – аналог 13, но первая точка – это значение первого среза;

В случае **N** интервалов вычисляется (**N**+1) значение для типов обработки 9 и 15 и **N** значений для типов обработки 11 и 13 (в последнем случае первый срез не записывается).

- начиная с **arg2**, аргументы задаются парами:
 - к первому привязывается атрибут (0,**R**) или (9,**Q**) канала, чьи значения должны быть извлечены из архива;
 - ко второму привязывается канал CALL.ChGroupReq, CALL.TVC или CALL.Program, в аргументы которого записываются обработанные срезы по каналу, привязанному к

первому аргументу (см. **Запись вектора в CALL.ChGroupReq и CALL.TVC**). Сумма срезов записывается в атрибут (1, **A**) привязанного канала CALL. При необходимости MPB корректирует количество аргументов привязанных каналов CALL. Для формирования значений кривой в канале CALL.TVC ко второму аргументу может быть привязан один из атрибутов CALL.TVC.142, CALL.TVC.143 и т.д.

- после требуемого количества пар в канале CALL.DifSnap может быть создан один аргумент типа OUTPUT. К этому аргументу должен быть привязан канал CALL.ChGroupReq или CALL.TVC, в аргументы которого записываются суммы всех обработанных срезов на одно и то же время (если к аргументу CALL.DifSnap привязан атрибут (9, **Q**) архивируемого канала, срезы по этому каналу вычитаются).

После отработки CALL.DifSnap его значение сбрасывается в 0 и устанавливается b14 (бит 0 атрибута 46).

Запись аргументов канала CALL в SIAD

Для записи в SIAD аргументов канала CALL.TVC или CALL.ChGroupReq (далее – **call**) нужно привязать определенный атрибут (**attr**) этого канала к атрибуту 2, **In** локального архивируемого числового канала (**ch**).

CALL.TVC

На каждом такте пересчета **ch** выполняются следующие действия:

- из всех временных меток, содержащихся в аргументах **call**, выбирается ближайшая справа от **ch.T (t)**;
- в точке **t** выбирается значение (**v**) кривой, заданной **attr (attr=142 – кривая 1, attr=143 – кривая 2 и т.д.)**;
- **ch.In == v, ch.T == t**;
- **ch** пересчитывается и записывается в архив.

CALL.ChGroupReq

Алгоритм аналогичен алгоритму для CALL.TVC, но дополнительно выполняются следующие функции:

- **attr** задает число кривых и выбираемую кривую:
 - 142 – одна кривая;
 - 146 и 147 – две кривых (146 – выбирается кривая 1, 147 – кривая 2);
 - 150, 151 и 152 – три кривых (150 – выбирается кривая 1 и т.д.);

- 155...158 – четыре кривых (155 – выбирается кривая 1 и т.д.);
- временные метки значений вычисляются по атрибутам 59 и 252 канала **call** (см. **Атрибуты каналов, отображаемые профайлером**).

При других значениях **attr**>142 выполняется обычное присвоение **ch.in == call.attr**.

Архивирование в регистратор

Для узла, передающего данные в регистратор, должно быть задано **При старте=Включен** (см. **Задание параметров узла**, раздел **Регистратор**).

Для конфигурирования используются следующие ключи в файле *.cnf:

- **USELOGGER**=<DEC номер узла> – номер узла регистратора.
- **LOG_SEND**=<протокол отправки данных в регистратор>:
 - **TCP** – TCP;
 - **IP** – UDP;
 - **GPRS** – GPRS;
 - **RS** – зарезервировано;
 - **MODEM** – зарезервировано;
 - **SMS** – зарезервировано;
 - **ON** – разрешить отсылку в регистратор;
 - **OFF** – запретить отсылку в регистратор.
- **LOG_SENDTYPE**=<1..7> – этот ключ нужно задать, если узел не может сам выбрать интерфейс обмена с регистратором (интерфейс остается 0).
- **LOG_QSIZE**=<DEC число блоков> – максимальный размер очереди отправки регистратору (по умолчанию 1024).
- **LOG_CHECK_TIME**=<число секунд, DEC> – период добавления блока в очередь на отсылку даже если блок не полон (по умолчанию 60с).

Один блок отсылки хранит до 90 значений.

Данное время может быть задано также с помощью системной переменной **@e_Logger** (15.2) с **Параметр=10** (см. **Группа ДИАГНОСТИКА**);

- **LOG_SEND_SLOW**=<0 .. 15с> – задержка отсылки в регистратор после предыдущей отсылки (1с по умолчанию); в реальном времени этот параметр задается с помощью переменной **@e_Logger**

(15.2) с **Параметр**=11;

- **LOG_STORY_TIME** – время хранения блока в очереди отсылки в регистратор. Возможные форматы ключа:
 - **LOG_STORY_TIME**=<число секунд>;
 - **LOG_STORY_TIME**=**HOUR** (1ч);
 - **LOG_STORY_TIME**=**DAY** (24ч).

Время хранения может быть задано также с помощью системной переменной **@e_Logger** (15.2) с **Параметр**=20 (см. **Группа ДИАГНОСТИКА**);

- **LOG_SEND_TIME**=<число секунд> – период отсылки данных в регистратор (см. ниже описание процедуры отсылки данных).

Значение ключа передается регистратору.

- **LOG_SEND_OFFSET**=<число секунд от начала дня> – смещение для **LOG_SEND_TIME**>=86400с (см. ниже описание процедуры отсылки данных);
- **LOG_DISCONNECT**=**ON/OFF** – разорвать/не разрывать соединение после передачи данных регистратору по TCP;
- **LOG_SAVE_FILE**=**ON** (по умолчанию)/**OFF** – разрешение/запрет сохранения данных в файл;
- **LOG_USE_PORT**:
 - если **LOG_USE_PORT**=**STD**, то для обмена используется порт 1027 (стандартный), но создается отдельный поток для отсылки в регистратор;
 - если **LOG_USE_PORT**=**LOG**, то для обмена используется порт 1027+4096, создается отдельный поток для отсылки в регистратор, и в узле регистратора создается отдельный поток для приема;
- **PERIOD_BLOCK_TIME_ADD**=<число секунд> – если в течение этого времени в блок не добавляются значения, формируется неполный блок.

Панель MPB отображает информацию об архивировании в регистратор следующим образом:

- в узле, посылающем данные:

L<маска, HEX>:<статус> **bs**:<число отправленных блоков> **ts**:<время последней отправки> **bq**:<очередь отправки (число блоков)> **be**:<число блоков, не доставленных в очередь>
- в регистраторе:

L:<число подключенных узлов (только при LOG_USE_PORT=LOG)> **bs:**<число принятых блоков>
ts:<время последнего приема> **wr:**<число значений, записанных в архив>

Условия архивирования канала в регистратор:

- в канале не установлен флаг **Автопосылка.Включить** – см. **Общие атрибуты каналов**;
- в канале установлен флаг **Регистратор** – см. **Общие атрибуты каналов**;
- значения каналов HEX16 и HEX32 отсылаются в регистратор при любом изменении или изменении достоверности;
- значения каналов FLOAT и DOUBLE FLOAT отсылаются в регистратор в следующих случаях (см. также **Уменьшение записей канала FLOAT в SIAD** в разделе **Архивы SIAD**):
 - изменилась достоверность;
 - значение перешло через границу (если границы используются, и установлен флаг **Отчет тревог**);
 - значение изменилось (если границы не используются, и не установлен флаг **Отчет тревог**).

Отсылка в регистратор:

1. **LOG_SEND=IP** или **LOG_SEND=TCP**.

2. Если установлен бит 15 переменной **ModeMaskTrace** (переменная 15.16 **@Redundant** с **Параметр=20** или ключ **DISABLE_IN_TRACE=<число HEX>**), в режиме TRACE отсылка в регистратор и добавление данных в очередь отсылки в регистратор не производится (данные в узле TRACE чистятся).

Это режим по умолчанию.

3. Для разрешения/запрета отсылки в регистратор используется ключ **LOG_SEND=ON/OFF** (в реальном времени – переменная 15.02 **@e_Logger** с **Параметр=16**).

Данные в узле не чистятся.

4. Данные отсылаются в регистратор порциями. Порция содержит **LOG_QSIZE** блоков.

Период отсылки задает ключ **LOG_SEND_TIME** (в реальном времени – переменная 15.02 **@e_Logger** с **Параметр=22**).

Если **LOG_SEND_TIME=0** (значение по умолчанию), данные в регистратор отсылаются сразу (при наличии данных и регистратора; учитывается

также задержка **LOG_SEND_SLOW**).

Если **LOG_SEND_TIME** \geq 86400с, данные отсылаются 1 раз в день во время, заданное ключом **LOG_SEND_OFFSET**.

Время отсылки всех данных запоминается.

Особенности отправки:

- отправка производится не чаще, чем 1 раз за время, заданное переменной 15.02 **@e_Logger** с **Параметр=11**;
- при отправке по UDP период не более 64с;
- если **LOG_DISCONNECT=ON**, **LOG_SEND=TCP**, и все данные отосланы, соединение разрывается.

5. На архивирование в регистратор оказывает влияние параметр **Статус** (см. **Задание параметров узла**, раздел **Регистратор**).

Если **Статус=Пассивен**:

- если порт регистратора не задан, используется predeterminedенный порт (в узле регистратора);
- используется протокол TCP (в узле регистратора и передающем узле);
- запускаются соответствующие потоки (в узле регистратора и передающем узле);
- если для передающего узла не задан период проверки наличия регистратора (**Период проверки состояния удаленных узлов** – см. **Задание параметров узла**, вкладка **Дополнительно**), этот период устанавливается равным параметру **LOG_SEND_TIME** передающего узла (+1с).

6. Если регистратор не обнаружен, передающий узел сохраняет данные в файл с именем **L<индекс узла>.bin** (при перезагрузке узла данные восстанавливаются из этого файла).

Если регистратор не обнаружен, в передающем узле генерируется ошибка соединения (TCP).

Индивидуальный архив

С помощью каналов CALL могут быть созданы индивидуальные (поканальные) архивы в памяти. Данные таких архивов могут быть запрошены у удаленного узла по любому доступному интерфейсу.

Для создания локального индивидуального архива используется канал CALL с типом вызова (23) **LArc0** или (24) **LArc1**. Канал, для которого создается архив, должен быть привязан к каналу CALL.

Аргументы 0 и 1 канала CALL используются следующим образом:

- **arg0** – для служебных целей (в профайлере в этом аргументе отображается привязка канала CALL; тип данных **arg0** должен соответствовать типу данных архивируемого канала);
- **arg1** – значение этого аргумента (тип данных – любой целочисленный) задается как число секунд. Период запроса значения привязанного канала устанавливается монитором кратным периоду пересчета канала CALL и примерно соответствующим **arg1**.

Архивируемые данные записываются в последующие аргументы канала CALL. Эти аргументы используются парами (создаются вручную или автоматически): в аргумент с четным порядковым номером записывается архивируемое значение, в последующий аргумент с нечетным порядковым номером – время этого значения. При создании вручную для аргументов с четными порядковыми номерами должен быть задан соответствующий числовой тип данных, для аргументов с нечетными порядковыми номерами – **Date_And_Time**.

Модификации индивидуальных архивов определяются значением атрибута **Параметр** канала CALL:

- **Параметр** = 0 – архив реальных значений привязанного канала (стек, последнее по времени значение записывается в первую пару аргументов);
- **Параметр** = 1...60 – циклически перезаписываемый архив реальных значений привязанного канала, усредненных по интервалу 1...60 минут;
- **Параметр** = 61...120 – циклически перезаписываемый архив реальных значений привязанного канала, усредненных по интервалу 1...60 минут, с нарастающим итогом. Если не начат новый цикл записи в архив, архивное значение представляет собой сумму предыдущего архивного значения и среднего значения привязанного канала за последний интервал. При начале нового цикла записи в архив архивное значение представляет собой среднее значение привязанного канала за последний интервал. Величина интервала усреднения (1-60) записывается в атрибут **34, FPrint**; 1 как признак вы-

числений нарастающим итогом – в атрибут **53, Update**.

Аргумент **arg1** канала CALL с усреднением должен быть равен 0, значения привязанного канала запрашиваются с периодом канала CALL, текущая сумма значений за заданный интервал записывается в атрибут **1, A**, текущее среднее – в атрибут **87, CC**.

Если аргументы канала CALL с усреднением не созданы, то в канале автоматически создается следующее количество пар аргументов для записи архивных значений и их времен:

- если интервал усреднения (**Interval**) больше или равен 10 минутам, в канале создается количество пар аргументов, необходимое для суточного архива:

$$\frac{24 * 60}{\text{Interval}}$$

- если интервал усреднения (**Interval**) меньше 10 минут, в канале создается количество пар аргументов, необходимое для 8-часового архива:

$$\frac{8 * 60}{\text{Interval}}$$

Времена для созданных таким образом пар аргументов фиксированы, первой паре соответствует начало суток (0 минут). При этом при старте монитора могут возникнуть две ситуации:

- время (**T** минут от начала суток) значения для записи в архив меньше или равно времени, соответствующему последней паре аргументов (это время равно **(N-1)*Interval**, где **N** – номер последней пары);
- **T > (N-1)*Interval**.

В первом случае первое вычисленное архивное значение и его время записываются в соответствующую пару аргументов.

Во втором случае архив закликивается необходимое число раз, и первое вычисленное архивное значение и его время записываются в соответствующую пару аргументов (номер этой пары равен остатку от деления **T** на **N**).

Если аргументы канала CALL с усреднением созданы вручную, первое вычисленное архивное значение и его время записываются в произвольную пару аргументов.

Алгоритм вычислений архивируемых данных для (23) **LArc0** и (24) **LArc1** отличается:

- для (23) **LArc0** реальное значение канала обрабатывается с заданным периодом, если с момента предыдущего запроса изменилось время значения;

- для (24) **LArc1** реальное значение канала обрабатывается с заданным периодом, если с момента предыдущего запроса значение изменилось.

Индивидуальный архив может быть отображен с помощью ГЭ/еГЭ **Тренд** (см. ГЭ 'Тренд'), а также выведен в генерируемый документ (на тренд – см. **Вставка тренда**).

Если имя канала CALL – индивидуального архива некоторого канала **ch** – начинается со знака "минус" и в канале **ch** не используется трансляция, MPB уничтожает канал CALL после переноса его аргументов в канал **ch**. Класс канала **ch** не изменяется, **ch.123=CALL.123**. Если существует несколько подобных архивов канала **ch**, уничтожается один канал CALL (с младшим ID).

Запрос удаленного индивидуального архива

Для запроса по M-LINK (в том числе через модем) или I-NET удаленных индивидуальных архивов используются каналы CALL INPUT с типами вызова (47) **RemArc0** и (48) **RemArc1** и типом пересчета **цикл IDLE** (ограничения на максимальное число передаваемых данных указаны в разделе **Особенности взаимодействия**). Сетевой протокол передачи по умолчанию – UDP.

В течение одной отработки канал **CALL.RemArc0** запрашивает несколько удаленных индивидуальных архивов, а канал **CALL.RemArc1** – один удаленный индивидуальный архив; полученные данные записываются в локальные архивы SIAD.

Конфигурация узла, содержащего канал **CALL.RemArc0** или **CALL.RemArc1**:

- необходимые архивы SIAD должны быть сконфигурированы, и их использование должно быть разрешено;
- для узла должен быть сконфигурирован соответствующий обмен;
- узел должен содержать один или несколько вспомогательных каналов типа INPUT (**loc_i**), в каждом из которых:
 - должна быть задана привязка к атрибуту **0,R** или **140,ARG00** удаленного индивидуального архива, данные которого требуется запрашивать;

Если канал CALL – индивидуальный архив некоторого канала **ch** – уничтожается монитором (см. **Индивидуальный архив**), в соответствующем **loc_i** должна быть задана привязка к **ch.R**.

- должно быть задано архивирование в один из сконфигурированных архивов SIAD.

Конфигурация канала **CALL.RemArc0** / **CALL.RemArc1**:

- тип – **INPUT**;
- тип пересчета – **цикл IDLE**;
- в ИС в канале **CALL.RemArc0** должна быть задана привязка к произвольному удаленному каналу; эта привязка определяет начальный узел, у которого канал будет запрашивать индивидуальные архивы. Для переключения канала **CALL.RemArc0** на запрос индивидуальных архивов другого узла нужно в реальном времени присвоить порядковый номер узла (**ordinal**) атрибуту 92, **I2** канала. В ИС в канале **CALL.RemArc1** должна быть задана привязка к одному из каналов **loc_i**; эта привязка определяет начальный индивидуальный архив (и одновременно узел), у которого канал будет запрашивать данные. Для переключения на запрос другого индивидуального архива (в том числе другого узла) нужно в реальном времени присвоить ID соответствующего канала **loc_i** атрибуту 89, **L0** канала **CALL.RemArc1**.

В реальном времени атрибуты 89 и 92 индицируют соответственно ID привязанного канала **loc_i** и **ordinal** запрашиваемого узла;

- аргументы **arg0** и **arg1** канала **CALL.RemArc0** / **CALL.RemArc1** должны иметь тип данных **DATE_AND_TIME**. Эти аргументы формируются в зависимости от битов 1 и 2 атрибута **Параметр** и задают временной диапазон выборки из полученных данных для дальнейшей записи в файлы:
 - если биты 1 и 2 равны 0, в **arg0** записывается время исчезновения, а в **arg1** – время появления узла (для фиксации времени используется собственный алгоритм канала);
 - если бит 2 равен 1, **arg1** может быть задан вручную;
 - если бит 1 равен 1, **arg0** может быть задан вручную;
 - если биты 1 и 2 равны 1, а **arg0** и **arg1** не заданы (**arg0** = **arg1** = 0), то **arg0** = **@Status.T_Loss**, **arg1** = **@Status.T_Detection** (см. описание переменной **@Status** в разделе **Группа СИСТЕМНЫЕ**);
 - если биты 1 и 2 равны 1, а заданные значения **arg0** и **arg1** не превышают числа секунд в году, то
arg1 = <текущее время> – <заданное значение **arg1**>
arg0 = **arg1** – <заданное значение **arg0**>
- начиная с **arg2**, аргументы **CALL.RemArc0** / **CALL.RemArc1** создаются парами. В четные аргументы (соответствующего числового типа данных) записываются запрошенные архивные значения, в нечетные (**DATE_AND_TIME**) – метки времени этих значений.

Канал **CALL.RemArc0** / **CALL.RemArc1** работает следующим образом (состояние канала в процедуре индицируют атрибуты (86, **nAttrt**) и (7,**P**):

- в начальном состоянии (0) канал проверяет наличие указанного удаленного узла. Если удаленный узел отсутствует, канал переходит в состояние 1;
- при переходе удаленного узла в состояние WORK канал переходит в состояние 2. В этом состоянии инициализируются внутренние переменные канала, и он переходит в состояние 3 (если удаленный узел «пропадает», канал переходит в состояние 0);
- в состоянии 3 при значении канала, отличном от 0, формируется запрос удаленного индивидуального архива:
 - в случае **CALL.RemArc0**, среди **loc_i** ищется канал с младшим ID (пусть его имя – **loc_A**), связанный с индивидуальным архивом указанного узла, формируется запрос этого архива, а для канала **loc_A** устанавливается признак игнорирования в последующем поиске;

Если последний символ кодировки **CALL.RemArc0** – звездочка, то **loc_A** выбирается в том случае, если первые символы кодировки удаленного индивидуального архива совпадают с кодировкой **CALL.RemArc0** (до знака звездочки).

- в случае **CALL.RemArc1** (пусть к нему привязан **loc_B**), формируется запрос того индивидуального архива, который привязан к **loc_B**.

После формирования запроса канал **CALL.RemArc0** / **CALL.RemArc1** переходит в состояние 4 (канал принимает значение 17). Если необходимые каналы среди **loc_i** не найдены, канал **CALL.RemArc0** / **CALL.RemArc1** переходит в состояние 0, и его значение обнуляется;

- в состоянии 4 в аргументы канала **CALL.RemArc0** / **CALL.RemArc1** копируются данные удаленного индивидуального архива, и из полученных данных производится выборка в диапазоне **arg0 < t < arg1**, которая записывается в тот же архив SIAD, в который архивируется **loc**. Кроме того, если бит 6 (0x40) атрибута **Параметр** канала **CALL.RemArc0** / **CALL.RemArc1** равен 1, выборка записывается в текстовый файл **loc_A.dra/loc_B.dra**.

Для отмены проверки времени при записи в SIAD и файлы *.dra нужно установить бит 7 (0x80) атрибута **Параметр** (34, **FPnt**) канала **CALL.RemArc0** / **CALL.RemArc1**.

- после получения данных в канале **CALL.RemArc0** устанавливается бит 4 (0x10) атрибута 46, **QE**, и канал переходит в состояние 3 (в случае сбоя – через промежуточное состояние 7, т.е. 4-7-3) для поиска очередного канала **loc_i**. MPB переходит к поиску следующего канала также в том случае, если в течение заданного таймаута получить архив не удалось (в этом случае признак достоверности

в канале **CALL.RemArc0** не устанавливается). Таймаут задается в файле *.cnf (ключи **ARCHT_NET** и **ARCHT_RS** – см. **Задание параметров работы мониторов**).

После получения данных обработка канала **CALL.RemArc1** завершается (канал запрашивает только один архив).

Число полученных архивов индицирует атрибут 2, **A** канала **CALL.RemArc0 / CALL.RemArc1**.

Дополнительные средства

В TRACE MODE предусмотрены дополнительные средства для архивирования и работы с архивами, эти средства описаны в следующих разделах:

- Регистратор аварийных событий;
- Канал CALL.TVC;
- Канал CALL.ChGroupReq;
- Канал CALL.Vector;
- Универсальный механизм обмена с электросчетчиками;
- Перенаправление архива в базу данных.

Перенаправление архива в базу данных

Перенаправление архива в БД конфигурируется с помощью ключей в файле *.cnf:

- **SQL_CHANNEL_NUMBER**=<номер канала для управления и диагностики> (CALL.ChGroupReq – внешняя DLL для записи, CALL.SQLQuery – через ODBC).
Ключ не требуется, если канал имеет имя @IDW;
- **SQL_NSIAD_NUMBER**=<номер архива> (3 по умолчанию);
- **SQL_QUEUE_SIZE**=<размер очереди на запись> (64000 по умолчанию);
- **SQL_AGE_PERIOD**=<количество часов назад> – нижняя временная граница выборки данных из очереди (24 по умолчанию);
- **SQL_COMMENT**=OFF/ON(по умолчанию) – не писать/писать статус работы в комментарий;
- **SQL_TIME_DISCONNECT**=<время в секундах> – таймаут разрыва соединения в отсутствие данных для записи (по умолчанию 120с);
- **SQL_TRACE_WORK**=ON(по умолчанию)/OFF – разрешение/запрет записи в режиме TRACE (если запись запрещена, данные в очередь не добавляются).
- **SQL_DELETE_BEFORE**=<T, число часов> – в БД хранятся значения за период (T_{ТЕКУЩЕЕ}-T), более ранние удаляются;
- **SQL_WORK_FLAG**= – управление режимом работы внешней DLL, обеспечивающей интерфейс к БД;
- **SQL_FLUSH_PERIOD**=<число минут> – запись в БД данных из очереди производится не чаще, чем задано этим ключом, или если число значений превышает значение ключа **SQL_FLUSH_COUNT**;
- **SQL_FLUSH_COUNT**=<число значений> – см. выше

SQL_FLUSH_PERIOD.

Реальное значение индицирует следующие состояния:

- 0 – отсоединен;
- 2 – норма;
- 4 – ошибка записи;
- 6 – ошибка подключения;
- 7 – отключен.

Использование других атрибутов:

- 90, **I0** – среднее число записей за 10с;
- 91, **I1** – число потерянных записей;
- 92, **I2** – текущий размер очереди на запись.

Диагностика в оболочке МРВ (см. **Панель МРВ**):

```
sqlw:%d,%d %d,%d %d,%d
```

- первое число – текущий счетчик записей за интервал;
- второе число – среднее число записей за 10с;
- третье число – число потерянных записей;
- четвертое число – общее число внесенных записей;
- пятое число – текущий размер очереди на запись;
- шестое число – достигнутый максимум текущего размера очереди на запись.

Если (8, **W**)=1 (отключен от источника/приемника), запись не производится.

Если (3, **C**)=1 (выключен) – реальное значение равно 7, записи нет и производится отключение.

Глава 10

Генерация документов

Использование разработанных шаблонов

Мониторы поддерживают функцию генерации файлов документов (отчетов) формата HTML по разработанным шаблонам (см. **Генерация документов (отчетов)**). Сгенерированный файл имеет имя **<имя канала вызова>.html**.

Для разработки шаблонов документов в ИС встроен соответствующий редактор (см. **Редактор шаблонов документов (отчетов)**).

Для входа в редактор аргументов шаблона документа используется команда **Аргументы** (меню **Вид**).

Вызов шаблона документа

Для вызова шаблона документа (генерации файла) используется канал CALL с типом вызова (3) **Document(Report)** (см. **Канал класса CALL** и **Атрибуты канала класса CALL**). Файл отчета генерируется при присвоении такому каналу ненулевого значения, при этом биты этого значения, установленные в 1, задают следующие опции:

- бит 0 (0x1) – вывод на принтер;
- бит 1 (0x2) – публикация на web-сервере (см. **TRACE MODE Data Center**);
- бит 4 (0x10) – отображение жирных кривых (3 px) на трендах.

После отработки значение канала автоматически сбрасывается в 0.

Биты атрибута **Параметр** канала CALL.Document задают параметры генерации:

- **Бит0=0** и **Бит1=0** – при многократной генерации новое содержимое добавляется в существующий файл в виде секции. Файл не добавляется в содержание (файл **index.html**). Заголовок секции содержит дату и время генерации, если одновременно бит 3 реального значения и бит 3 атрибута **Параметр** равны 0;
- **Бит0=1** – разрешение добавления файла в содержание;
- **Бит1=1** – разрешение перезаписи файла;
- **Бит4=1** – задание сохранения файла в папке, путь к которой задан атрибутом 80, **COMMNT**; если **Бит4=0**, файл сохраняется в папке узла;
- **Бит5=1** – задание вывода документа на принтер;

Сгенерированный документ передается по заданному

направлению только в том случае, если предварительно запущен соответствующий модуль (**сервер печати TRACE MODE 6** или **TRACE MODE Data Center**), и использование этого модуля разрешено.

- **Бит6=1** – разрешение использования **сервера печати** (см. **Сервер печати TRACE MODE 6**);
- **Бит7=1** – разрешение использования **Data Center**.

В атрибут 87, **СС** канала CALL.Document записывается время (в миллисекундах), затраченное на генерацию отчета (см. **Атрибуты каналов, отображаемые профайлером**). Пока документ генерируется, в атрибуте 87 хранится время начала генерации. В атрибут 92, **I2** записывается код ошибки (см. **Коды диагностируемых ошибок**).

Значение байта 1 (0xFF00) атрибута 2, **In** или значение атрибута 1, **A** канала задает **категорию** документа (этот параметр может использоваться для конфигурирования дерева публикатора).

Файл index.html

Файл **index.html** создается при генерации документа, для которого добавление в содержание разрешено. Файл сохраняется в той же папке, что и документ, и содержит гиперссылки на аналогичные документы.

В заголовок файла содержания выводится строка **<имя файла prj>_<ordinal>** (см. **Файлы узла, создаваемые при экспорте**).

Текст гиперссылки на документ в файле содержания соответствует титулу документа (см. **Задание свойств документа**).

Редактирование шаблонов документов

Редактор шаблонов документов (отчетов)

При разработке шаблонов документов в редакторе используются типовые операции редактирования (см. **Редактор шаблонов документов** и **Типовые операции редактирования**). В редакторе предусмотрен также набор команд, для выполнения которых используются:

- меню **Правка** (встраивается в главное меню ИС и автоматически модифицируется в зависимости от доступных в данный момент операций и положения курсора);
- контекстное меню (содержит те же команды, что и меню **Правка**);
- панели типовых инструментов (добавляются к панелям инструментов ИС, см. **Типовые инструменты редактирования**):

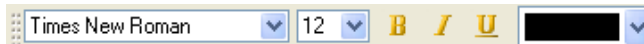
- **Правка**



- **Поиск**



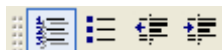
- **Шрифт**



- **Выравнивание**



- **Список**



- **Таблица**









Аналогичные команды содержит раздел **Таблица** меню **Правка** и контекстного меню, если курсор расположен в пределах таблицы:




Вставить таблицу



Вставить столбец слева (от текущего)








-  Вставить столбец справа
-  Вставить строку выше
-  Вставить строку ниже
-  Удалить столбец
-  Удалить строку
-  Удалить таблицу

Если курсор находится вне таблиц, доступна только команда  **Вставить таблицу**;

- **Объекты**



Команды этой панели идентичны командам раздела **Вставить** меню **Правка** и контекстного меню:

-  Вставить горизонтальную линию
-  Вставить картинку
-  Вставить выражение времени
-  Вставить тренд
-  Вставить круговую диаграмму
-  Вставить гистограмму
-  Вставить отчет тревог
-  Вставить документ

Раздел **Свойства** меню **Правка** и контекстного меню содержит команду **Документ** для открытия диалога редактирования свойств документа. Кроме того, при установке курсора слева от элемента шаблона, в раздел **Свойства** автоматически добавляется команда (команды), с помощью которой открывается диалог редактирования свойств этого элемента.

Для входа в редактор аргументов шаблона документа используется команда **Аргументы** (меню **Вид**).

Задание параметров редактора документов

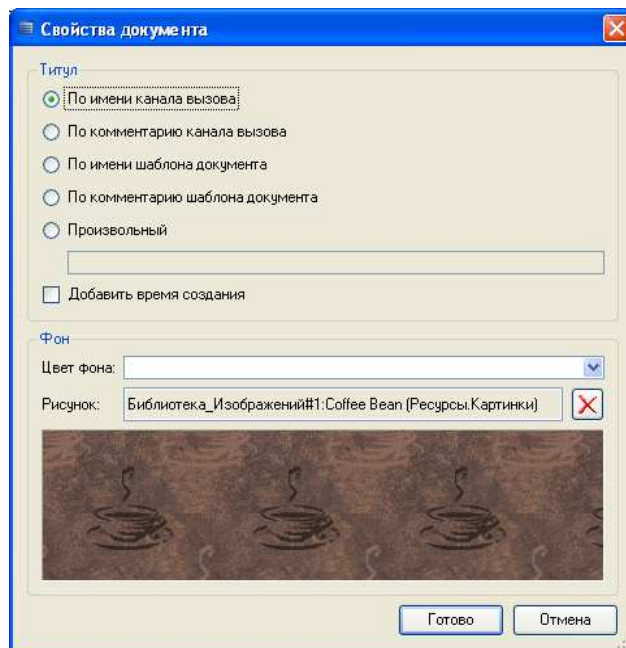
Вкладка **Редактор документов** окна **Настройки**, вызываемого из меню **Файл**, содержит диалог задания параметров редактора.

В разделе **Общие** этой вкладки задаются следующие параметры:

- **Выделять некорректные объекты** – если этот флаг установлен, в редакторе выделяются красным цветом объекты, зависящие от несуществующих аргументов.


Задание свойств документа

При выполнении команды **Документ** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется следующий диалог:



Раздел **Титул** этого диалога содержит инструменты для выбора метода формирования титула документа (тега <TITLE>):

- **По имени канала вызова**
- **По комментарию канала вызова**
- **По имени шаблона документа**
- **По комментарию шаблона документа**
- **Произвольный**
- **Добавить время создания** – если этот флаг установлен, в титул добавляется время генерации.

Раздел **Фон** диалога содержит инструменты задания фона шаблона документа (цвет или рисунок из библиотеки изображений или из файла). Установка рисунка в качестве фона имеет более высокий приоритет по отношению к цвету. При этом кнопка  в диалоге **Свойства докумен-**

та заменяется кнопкой **X** – при ее нажатии использование рисунка в качестве фона отменяется.

В нижней части диалога располагается окно просмотра цвета/рисунка, выбранного в качестве фона.

Форматирование текста

Текст в шаблон документа вводится с помощью клавиатуры. Для редактирования и форматирования текста используются типовые средства (см. **Редактор шаблонов документов (отчетов)**, а также **Типовые инструменты редактирования** и **Типовые операции редактирования**).

Форматирование списков

Кроме стандартных средств (см. **Редактор шаблонов документов (отчетов)**, а также **Типовые инструменты редактирования** и **Типовые операции редактирования**), для форматирования списков предусмотрены два диалога (**Свойства списка** и **Свойства элемента списка**), которые появляются на экране при выполнении команд **Список** и **Элемент списка** раздела **Свойства** меню **Правка** или контекстного меню.

Форматирование, заданное в диалоге **Свойства списка**, применяется ко всему списку. Форматирование, заданное в диалоге **Свойства элемента списка**, применяется к элементу списка.

Для преобразования списка в обычный текст используется команда **Уменьшить отступ**.

Каждый из диалогов включает две вкладки (**Нумерованный** и **Маркированный**).

Форматирование нумерованных списков




На вкладке **Нумерованный** задаются параметры нумерованных списков:

- при установке флага **Стиль** на вкладке доступны переключатели вида представления номера (без номера, десятичные числа, римские числа в нижнем регистре, римские числа в верхнем регистре, строчные латинские буквы, прописные латинские буквы);
- при установке флага **Шрифт** на вкладке доступны типовые инструменты форматирования номера (цвет, семейство, размер и вид начертания шрифта).

В нижней части вкладки расположено окно предварительного просмотра заданного форматирования.


Форматирование маркированных списков

На вкладке **Маркированный** задаются параметры маркированных списков.

При установке флага **Стиль** на вкладке доступны инструменты выбора картинки, используемой в качестве маркера: **по умолчанию** (в диалоге **Свойства элемента списка** этой опции нет), **без маркера, квадрат, круг, диск, произвольный**. При выборе произвольного маркера доступна кнопка . При ее нажатии на экране появляется диалог выбора картинки из библиотеки изображений или из файла. После выбора кнопка  заменяется кнопкой  – при ее нажатии использование выбранного рисунка в качестве маркера отменяется.

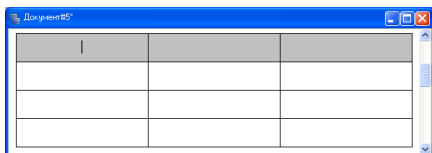
В нижней части вкладки расположено окно предварительного просмотра заданного форматирования.

Использование таблиц в шаблоне документа

При выполнении команды  **Вставить таблицу** на экране появляется диалог выбора типа таблицы (**Обычная таблица** или **Таблица архива**).

Конфигурирование обычной таблицы

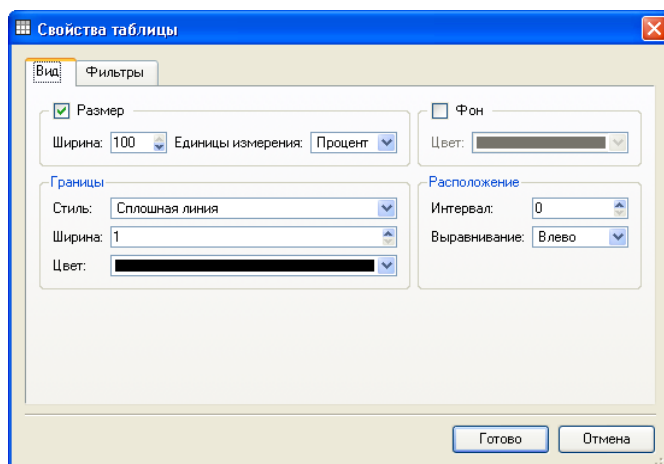
При вставке в шаблон документа обычной таблицы (см. **Использование таблиц в шаблоне документа**) на экране появляется диалог задания числа строк и столбцов создаваемой таблицы. Диалог содержит также флаг **Форматировать первую строку как заголовок таблицы**, при установке которого параметры форматирования первой строки устанавливаются по умолчанию:



При выполнении команды **Таблица** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог **Свойства таблицы**, в котором задаются параметры таблицы.

Форматирование таблицы

На вкладке **Вид** диалогового **Свойства таблицы** задается форматирование таблицы:



При установке флага **Размер** доступны инструменты задания ширины

таблицы – в процентах по отношению к ширине документа или в пикселях.

В разделе **Границы** задаются стандартные параметры границ таблицы – стиль, ширина в пикселях и цвет.

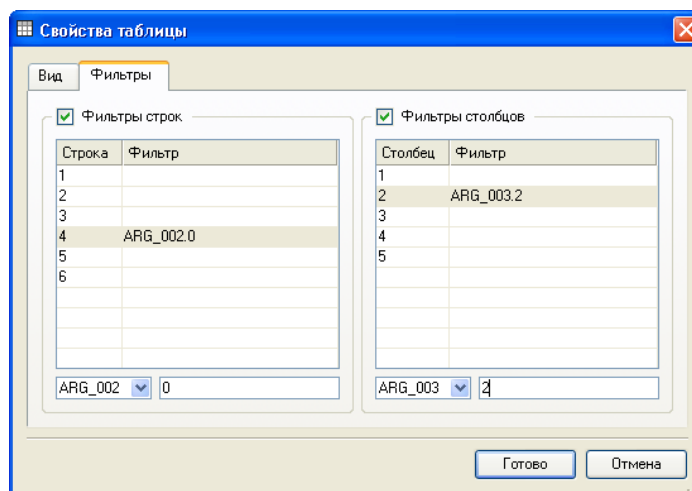
При установке флага **Фон** доступен список **Цвет**, с помощью которого задается цвет фона таблицы.

В разделе **Расположение** задается размер промежутка между ячейками таблицы (в пикселях, одновременно по горизонтали и вертикали), а также выравнивание таблицы в документе.

Заданные параметры не применяются к ячейкам таблицы, которые форматировались вручную.

Задание фильтров в таблице

На вкладке **Фильтры** диалога **Свойства таблицы** задаются фильтры строк/столбцов:

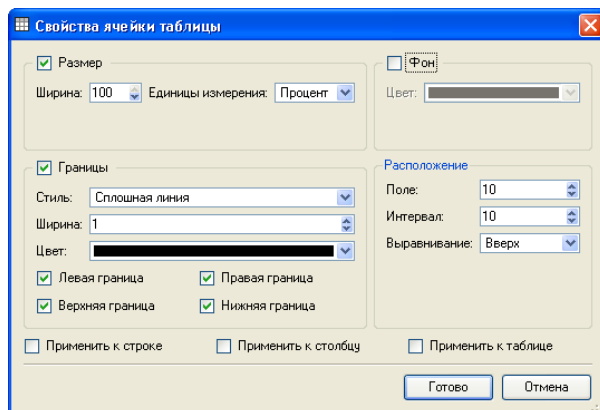


Для задания фильтра строки/столбца нужно выполнить следующие действия:

- установить соответствующий флаг разрешения использования фильтров (**Фильтры строк** / **Фильтры столбцов**);
- нажатием ЛК выделить строку/столбец в списке строк/столбцов;
- выбрать аргумент шаблона документа в нижнем списке;
- в поле справа от списка аргументов ввести номер атрибута. Строка/столбец выводится в документ, если в момент его генерации указанный атрибут канала, привязанного к выбранному аргументу, отличен от 0.

Форматирование ячеек таблицы

При выполнении команды **Ячейка таблицы** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется следующий диалог:



При установке флага **Размер** доступны инструменты задания ширины ячейки – в процентах по отношению к ширине таблицы или в пикселях.

При установке флага **Фон** доступен список **Цвет**, с помощью которого задается цвет фона ячейки таблицы.

При установке флага **Границы** доступны инструменты задания стандартных параметров границ ячейки таблицы (стиля, ширины в пикселях и цвета), а также флаги использования (отображения) этих границ.

В разделе **Расположение** задаются поля ячейки (в пикселях, параметр **Поле** задает одновременно все четыре поля), интервал после абзаца (в пикселях), а также выравнивание текста в ячейке.

С помощью флагов в нижней части диалога можно применить заданное форматирование к строке, столбцу или всей таблице. Если эти флаги не установлены, форматирование применяется к ячейке.

Конфигурирование таблицы архивных значений

При вставке в шаблон документа таблицы архивных значений (см. **Использование таблиц в шаблоне документа**) на экране появляется диалог задания числа столбцов.

При размещении таблицы в шаблоне она по умолчанию имеет вертикальный вид (назначение первого столбца – отображение времени – фиксировано):

| Время | | |
|---------|--|--|
| 0:00:00 | | |

Для вывода в столбец архивных значений (из SIAD) в нем нужно разместить одно из следующих выражений (см. **Вставка значения переменной**):

@<имя аргумента>.<номер атрибута>

@<имя аргумента>.<короткое имя атрибута>

@ (<имя аргумента>.<номер атрибута>,<формат>)

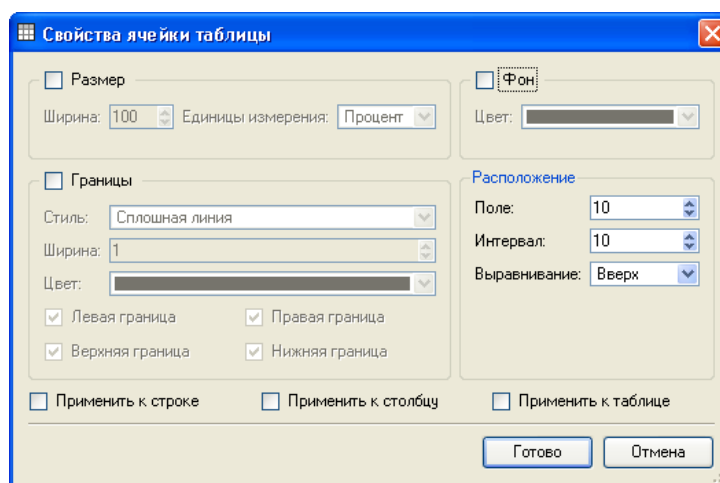
@ (<имя аргумента>.<короткое имя атрибута>,<формат>)

В сгенерированном документе число архивных значений в таблице (число ее строк) будет соответствовать разбиению временного диапазона, заданному при конфигурировании таблицы (см. ниже **Задание параметров выборки**).

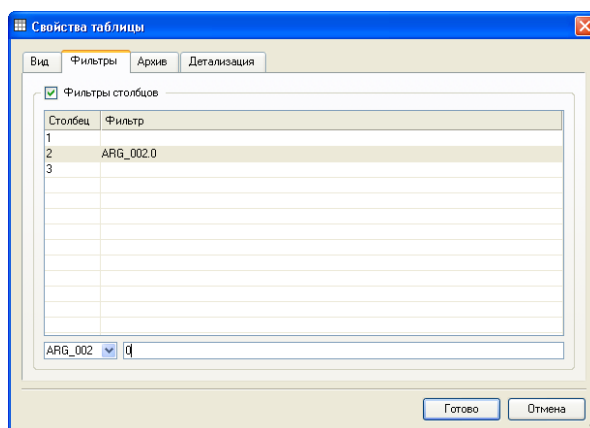
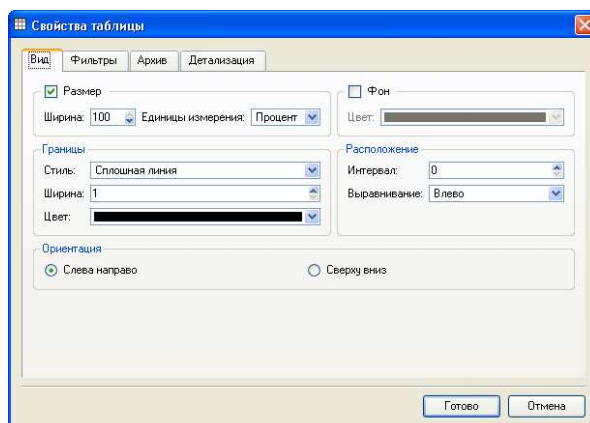
Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия MPB с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа**.

Общее форматирование таблицы

При выполнении команды **Ячейка таблицы** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог форматирования ячейки таблицы:



При выполнении команды **Таблица** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог, на вкладках **Вид** и **Фильтры** которого задаются соответственно форматирование таблицы и фильтры столбцов:

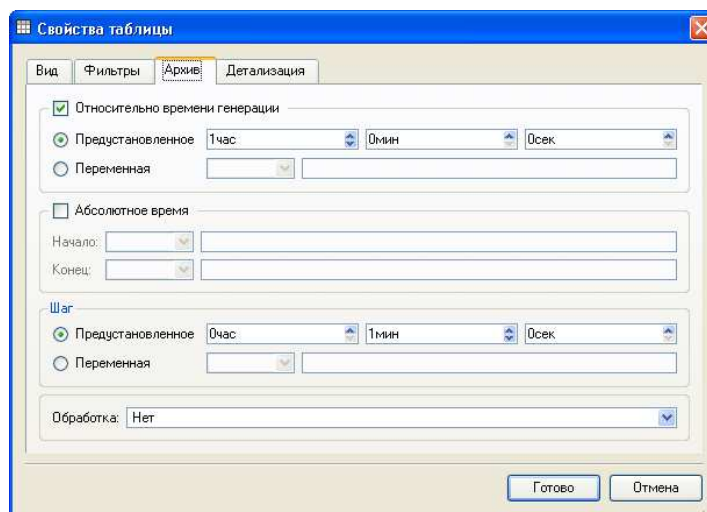


Параметры форматирования и фильтров, показанные на рисунках, аналогичны соответствующим параметрам обычной таблицы (см. **Конфигурирование обычной таблицы**). В отличие от обычной таблицы, вкладка **Вид** диалога свойств архивной таблицы содержит дополнительные флаги в разделе **Ориентация**:

- **Слева направо** – при установке этого флага таблица ориентирована горизонтально; данные, извлеченные из архива, размещаются в таблице по возрастанию времени слева направо;
- **Сверху вниз** – при установке этого флага таблица ориентирована вертикально (значение по умолчанию); данные, извлеченные из архива, размещаются в таблице по возрастанию времени сверху вниз.

Задание параметров выборки

Временной диапазон выборки из архива и его разбиение на интервалы, а также вид обработки данных задаются на вкладке **Архив** диалога **Свойства таблицы**:

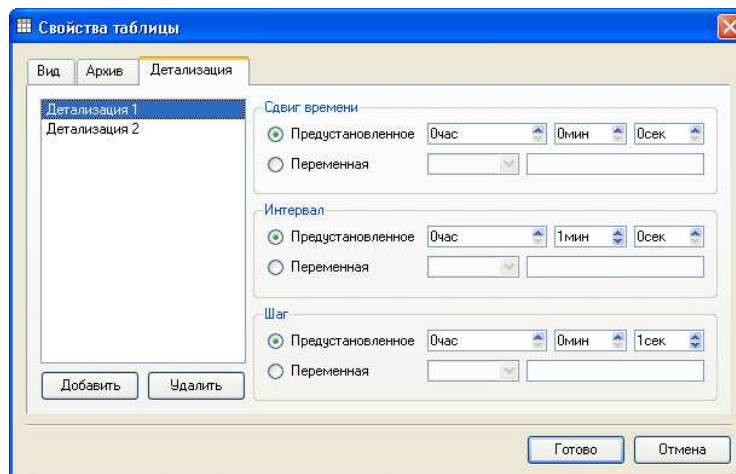


Параметры, задаваемые в разделах **Относительно времени генерации** и **Абсолютное время**, аналогичны параметрам задания временного диапазона выборки из архива для тренда (см. **Вставка тренда**).

В разделе **Шаг** задается шаг выборки значений из заданного временного диапазона:

- **Предустановленное** – при выборе этой опции шаг задается как абсолютное время;
- **Переменная** – при выборе этой опции шаг задается значением атрибута канала, привязанного к выбранному аргументу. Аргумент выбирается в списке, атрибут указывается в поле справа.

Детализации выборки из архива могут быть заданы на вкладке **Детализация** диалога **Свойства таблицы**:



Для добавления детализации нужно нажать кнопку **Добавить**, для удаления детализации, выделенной в списке слева, нужно нажать кнопку **Удалить**.

Каждая детализация разбивает временной поддиапазон от

$T_{\text{генерации}} - \text{Сдвиг времени}$

до

$T_{\text{генерации}} - \text{Сдвиг времени} + \text{Интервал}$

с шагом **Шаг**.

Параметры **Сдвиг времени**, **Интервал** и **Шаг** могут быть заданы абсолютно (**Предустановленное**) или определяться значением атрибута канала, привязанного к выбранному аргументу (**Переменная**).

Если задано большое число разбиений временного диапазона выборки из архива, для генерации документа потребуется значительное время.

Вид обработки данных, извлеченных из архива, задается с помощью списка **Обработка** вкладки **Архив**:

- **Нет** – без обработки;
- **Разность** – в таблицу заносится разность текущего и предыдущего извлеченных значений;
- **Всего** – в таблице создается дополнительный столбец/строка (в зависимости от ориентации), в который записываются суммы извлеченных значений соответственно по столбцам/строкам.

Вставка объектов в шаблон документа

Вставка значения переменной

В шаблоне документа могут быть использованы следующие выражения:

@<имя аргумента>

@<имя аргумента>.<номер атрибута>

@<имя аргумента>.<короткое имя атрибута>

@<имя аргумента>.<SubNum>

В первом случае в генерируемый документ выводится значение аргумента.

Выражение @<имя аргумента> создается автоматически при перетаскивании аргумента из редактора аргументов в шаблон документа (drag-and-drop).

С помощью выражений 2 и 3 в генерируемый документ можно вывести значение произвольного атрибута канала, привязанного к аргументу.

Выражение 4 задает одну из предопределенных функций вывода (см. **Номер SubNum**).

Форматированный вывод числовых значений задается с помощью следующих выражений:

@ (<имя аргумента> ,<формат>)

@ (<имя аргумента> .<номер атрибута> ,<формат>)

@ (<имя аргумента> .<короткое имя атрибута> ,<формат>)

где <формат> – Си-формат числа (см. **Формат Си вывода чисел**).


При синтаксической ошибке выражение подчеркивается красной чертой.

Вставка горизонтальной линии

По команде  **Вставить горизонтальную линию** в шаблон документа вставляется горизонтальная линия заданной по умолчанию толщины.


При выполнении команды **Горизонтальная линия** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог задания толщины линии.

Вставка рисунка

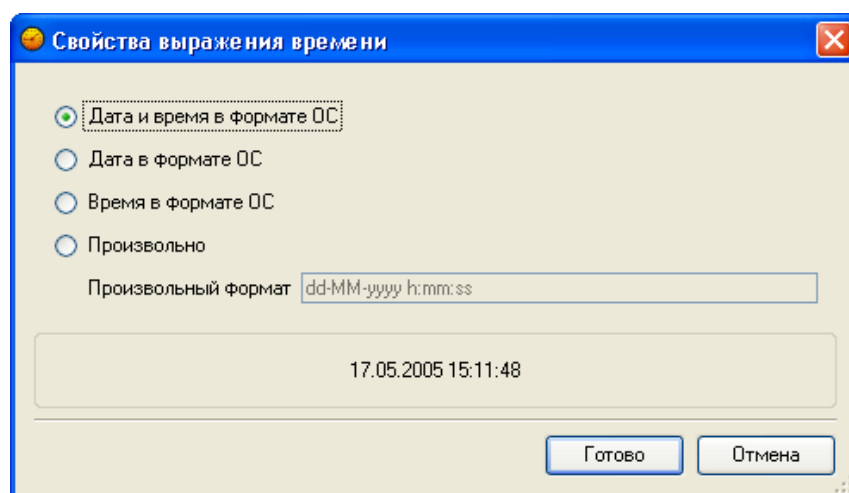
По команде  **Вставить картинку** на экране появляется диалог выбора статической картинке из библиотеки изображений или из файла.

Параметры этого объекта не редактируются.

Вставка выражения времени

Чтобы документ содержал значение времени генерации, в шаблон документа нужно вставить соответствующий объект по команде  **Вставить выражение времени**.

При выполнении команды **Выражение времени** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется следующий диалог:




В этом диалоге задается один из следующих форматов времени:

- дата и время в формате, соответствующем региональным настройкам ОС (значение по умолчанию);
- дата в формате, соответствующем региональным настройкам ОС;
- время в формате, соответствующем региональным настройкам ОС;
- произвольно – при выборе этой опции формат времени задается в текстовом поле (описание форматов отображается во всплывающей подсказке поля).

В нижней части диалога отображается текущее время в выбранном формате.

Вставка тренда


Для отображения архивных данных в виде тренда, в шаблон документа нужно вставить соответствующий объект по команде  **Вставить тренд**.

При выполнении команды **Тренд** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог конфигурирования тренда, содержащий несколько вкладок.

Вкладка 'Аргументы'

Эта вкладка предназначена для задания кривых тренда (отображаются в таблице в верхней части вкладки).

Для добавления кривой нужно выполнить следующие действия:

- нажать кнопку **Добавить**;
- в разделе **Имя** выбрать аргумент и атрибут канала, привязанного к этому аргументу (по умолчанию – 127, NAME);
- в разделе **Значение** выбрать аргумент и атрибут канала, привязанного к этому аргументу (по умолчанию – 0, R). Для вывода данных индивидуального архива к аргументу должен быть привязан соответствующий канал CALL (см. **Индивидуальный архив и MPV как клиент сервера OPC HDA**);
- в разделе **Цвет** выбрать цвет для кривой;
- при необходимости, в разделе **Фильтр** выбрать аргумент и атрибут канала, привязанного к этому аргументу. Кривая выводится на тренд, если в момент генерации документа указанный атрибут отличен от 0. Для удаления условия вывода кривой нужно нажать кнопку ;
- в разделе **Тип** выбрать тип кривой – **аналоговый** или один из **дискретных** (аналогичных интерпретациям дискретных кривых ГЭ 'Тренд'). В зависимости от типа, кривая отображается на соответствующей панели тренда (верхняя панель – аналоговая, под ней расположена дискретная панель);
- в разделе **Масштаб** задать диапазон по оси Y (только для аналоговых кривых):
 - **Абсолютный** – при выборе этой опции диапазон ординат устанавливается автоматически от 0 до максимального выводимого значения;
 - **Относительный** – при выборе этой опции диапазон ординат устанавливается автоматически от минимального выводимого значения до максимального;
 - **Произвольный** – при выборе этой опции диапазон ординат задается вручную в соответствующих полях.

Во всех случаях заданный диапазон будет автоматически расширен, если отображаемые значения выходят за его пределы.

Для добавления кривых можно перетащить выделенную группу аргументов на тренд, и далее сконфигурировать кривые.

Для удаления кривой, выделенной в таблице, нужно нажать кнопку **Удалить**. Для перемещения кривой на одну позицию вверх/вниз в таблице нужно нажать кнопку **Вверх/Вниз**.

Вкладка ‘Архив’

На этой вкладке конфигурируется выборка архивных данных:

Особенности отображения и ошибки описаны в разделах **Особенности взаимодействия MPB с документом/экраном** и **Ошибки выборки данных по запросу экрана/документа**.

Временной интервал выборки (**T_FROM**, **T_TO**) может быть задан относительно времени генерации документа (раздел **Относительно времени генерации**) или абсолютно (раздел **Абсолютно**).

Раздел **Относительно времени генерации** содержит следующие инструменты:

- **Предустановленное** – при задании некоторого значения **T** в этом разделе $T_FROM = T_{generation} - T$, $T_TO = T_{generation}$;
- **Переменная** – при выборе этого раздела $T_FROM = T_{generation} - T$, $T_TO = T_{generation}$, где **T** равно значению атрибута канала, привязанного к выбранному аргументу (значение атрибута интерпретируется как число секунд). Аргумент выбирается в списке, атрибут указывается в поле справа.

По умолчанию, MPB дополнительно ищет первое архивное значение левее заданного **T_FROM** и также передает его в документ. Этот поиск можно отменить с помощью ключа **HNRDSIAD=OFF** в файле *.cnf.

Раздел **Абсолютно** содержит следующие инструменты:

- **Начало** – привязка **T_FROM** к атрибуту/аргументу канала, привязанного к выбранному аргументу;
- **Конец** – привязка **T_TO** к атрибуту/аргументу канала, привязанного к выбранному аргументу.

Если флаг **Использовать быструю выборку** установлен, будет ис-

пользоваться алгоритм быстрой выборки из архива.

Вкладка ‘Вид’

На этой вкладке конфигурируется вид тренда.

Вкладка содержит следующие инструменты:

- раздел **Тип** – инструменты задания вида кривых на аналоговой панели;
- раздел **Размер** – инструменты **Ширина** и **Высота** задания размера тренда в пикселях;
- **Цвет фона** – при установке этого флага доступно задание цвета фона тренда;
- **Цвет фона панелей тренда** – при установке этого флага доступно задание цвета фона панелей тренда;
- раздел **Рамка тренда** – инструменты задания цвета и толщины рамки тренда;
- раздел **Цвета статусов** – инструменты задания цветов статусов 0...7 для отображения кривых с типом **статус** на дискретной панели.

Вкладка ‘Оси’

На этой вкладке конфигурируются параметры осей тренда.

Вкладка содержит следующие инструменты:

- раздел **Ось значений**:
 - **Разбиение** – число делений оси;
 - **Период подписи** – период подписей на оси (в делениях);
 - **Одна ось на аналоговой панели** – если этот флаг установлен, аналоговая панель содержит единственную ось значений, в противном случае для каждой кривой отображается своя ось;
- раздел **Ось времени**:
 - **Разбиение** – число делений оси;
 - **Период подписи** – период подписей на оси (в делениях);
 - **Формат времени** – формат времени в подписи (описание форматов отображается во всплывающей подсказке поля);
 - **Формат даты** – формат даты в подписи (описание форматов отображается во всплывающей подсказке поля);
 - **Дата в подписи** – отображение даты в подписи (**Всегда** или **Только при смене даты**);
 - **Увеличение диапазона времени** – увеличение диапа-

зона времени в случае необходимости (**Не использовать** или **С точностью до деления**);

- раздел **Шрифт** – инструменты задания типовых параметров шрифта, используемого для отображения значений осей.


Вкладка ‘Колонтитулы’

На этой вкладке задаются типовые параметры колонтитулов, отображаемых на тренде.

Вкладка ‘Легенда’

На этой вкладке задаются типовые параметры легенды, отображаемой на тренде. В легенду выводятся имена кривых и их цвет.

Вставка круговой диаграммы

Для отображения в документе значений переменных в виде круговой диаграммы, в шаблон документа нужно вставить соответствующий объект по команде  **Вставить круговую диаграмму**.

При выполнении команды **Круговая диаграмма** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог конфигурирования диаграммы, содержащий несколько вкладок.

Параметры, задаваемые на вкладках **Аргументы**, **Вид** и **Колонтитулы**, аналогичны параметрам, задаваемым на одноименных вкладках диалога конфигурирования тренда (см. **Вставка тренда**).

Вкладка ‘Легенда’

На этой вкладке задаются типовые параметры легенды, отображаемой на диаграмме.

Если флаг **Показать значения** установлен, а флаг **Показать в процентах** не установлен, в легенду выводятся значения переменных.

Если установлены оба флага, в легенду выводятся значения переменных в процентах.

Вкладка ‘Подписи’

На этой вкладке задаются параметры подписей к секторам диаграммы.

Для конфигурирования подписей нужно установить флаг **Подписи**, при этом доступны следующие инструменты:

- **Показать подписи** – если этот флаг установлен, подписи к сек-

торам отображаются на диаграмме;

- **Показать значения** – аналог одноименного флага легенды;
- **Показать в процентах** – аналог одноименного флага легенды;
- **Показать ключ легенды** – если этот флаг установлен, в подписи отображается цвет сектора.

В разделе **Шрифт** задаются типовые параметры шрифта, используемого для подписей.

Вставка гистограммы

Для отображения в документе значений переменных в виде гистограммы, в шаблон документа нужно вставить соответствующий объект по команде



Вставить гистограмму.

При выполнении команды **Гистограмма** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог конфигурирования гистограммы, содержащий несколько вкладок.

Параметры, задаваемые на вкладках **Аргументы**, **Вид** и **Колонтитулы**, аналогичны параметрам, задаваемым на одноименных вкладках диалога конфигурирования тренда (см. **Вставка тренда**).

Параметры, задаваемые на вкладках **Легенда** и **Подписи**, аналогичны параметрам, задаваемым на одноименных вкладках диалога конфигурирования круговой диаграммы (см. **Вставка круговой диаграммы**).

Вкладка 'Ось'

На этой вкладке конфигурируются параметры оси гистограммы.

Если флаг **Ось** не установлен, на гистограмме не отображается ось, подписи и сетка. При установке флага доступны следующие инструменты:

- **Показать значения** – если этот флаг установлен, слева от штрихов оси выводятся их значения;
- **Показать главную сетку** – если этот флаг установлен, отображаются линии сетки;
- **Показать дополнительную сетку** – если этот флаг установлен, отображаются линии дополнительной сетки. Отображение дополнительной сетки возможно только в том случае, если отображается главная сетка.

В разделе **Масштаб** задается диапазон по оси Y (см. **Вставка тренда**).

Вставка отчета тревог

Для отображения в документе данных отчета тревог (ОТ, см. **Отчет тревог узла**), в шаблон документа нужно вставить соответствующий объект

по команде  **Вставить отчет тревог**.

Таблица содержит столбцы **Время** (данные полей **Date** и **Time** ОТ – см. **Формат строки ОТ**), **Категория**, **Имя**, **Кодировка**, **Текст** (соответственно данные полей **Category**, **Name**, **Coding** и **Text** ОТ) и **Квитирование** (информации о квитировании сообщения ОТ).

При выполнении команды **Отчет тревог** раздела **Свойства** меню **Правка** или контекстного меню на экране появляется диалог конфигурирования таблицы.

На вкладках **Границы** и **Заголовок** диалога задаются стандартные параметры форматирования границ и заголовка таблицы.

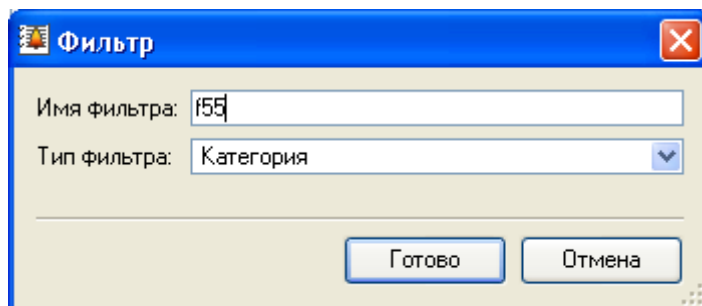
На вкладке **Вид** задаются параметры форматирования для каждой категории сообщений (см. **Редактор словарей сообщений**).

Фон, заданный в верхней части вкладки, применяется к тем категориям сообщений, для которых не задан собственный фон.

Вывод заголовка и столбцов таблицы в документ, а также выравнивание текста в заголовке и столбцах задаются на вкладке **Выравнивание и видимость**.

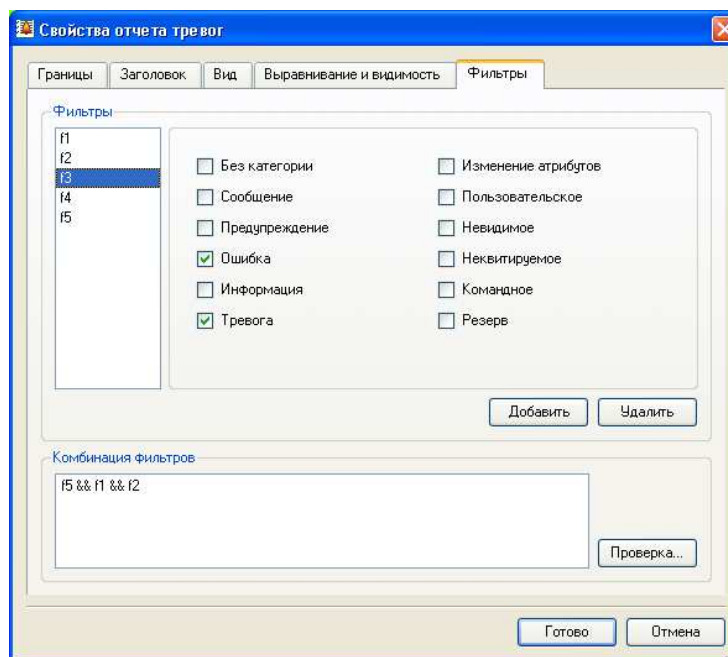
Фильтрация сообщений ОТ для вывода в документ задается на вкладке **Фильтры**, которая содержит два раздела – **Фильтры** (конфигурирование фильтров) и **Комбинация фильтров** (применение фильтров).

Для управления списком фильтров используются кнопки **Добавить** и **Удалить** раздела **Фильтры**. При добавлении фильтра на экране появляется следующий диалог:

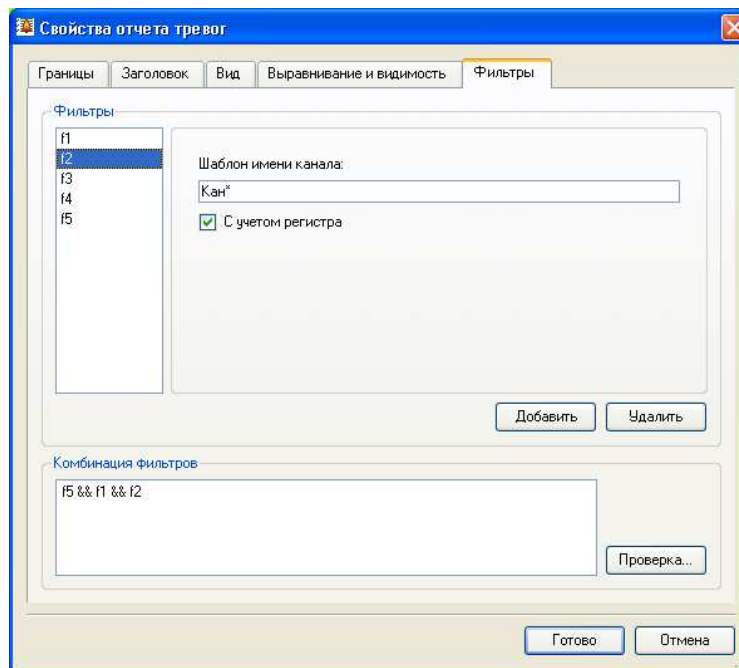


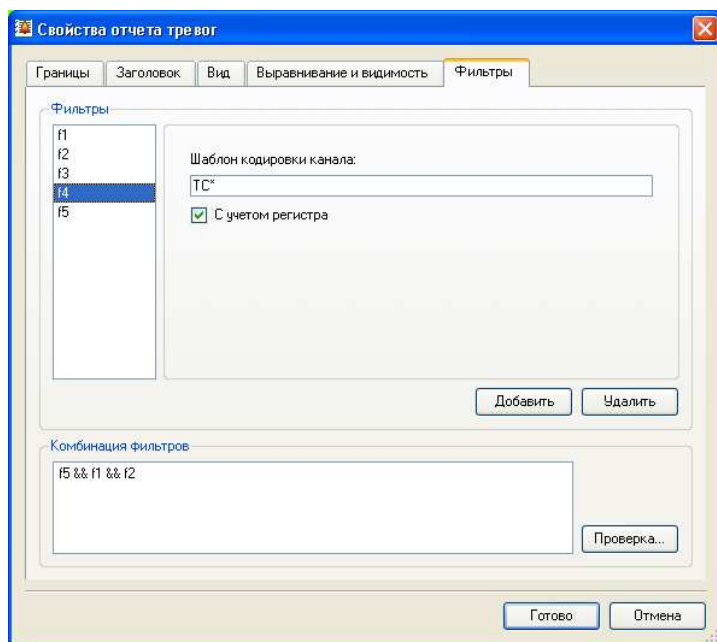
В этом диалоге нужно задать имя (должно начинаться с буквы, может содержать цифры и символы подчеркивания) и выбрать тип фильтра (по категории сообщений, по имени канала, по кодировке канала, по квитированию и по времени). В зависимости от выбранного типа в разделе **Фильтры** отображаются соответствующие инструменты конфигурирования фильтра.

Конфигурирование фильтра по категории сообщений (выбранные категории выводятся в документ):

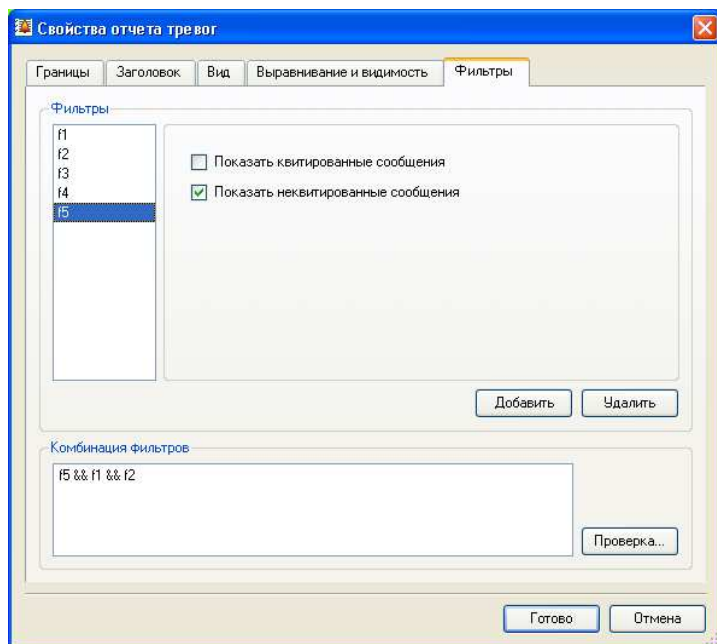


Конфигурирование фильтров по имени и кодировке канала (в обоих случаях при задании шаблона могут использоваться стандартные знаки замены символов):

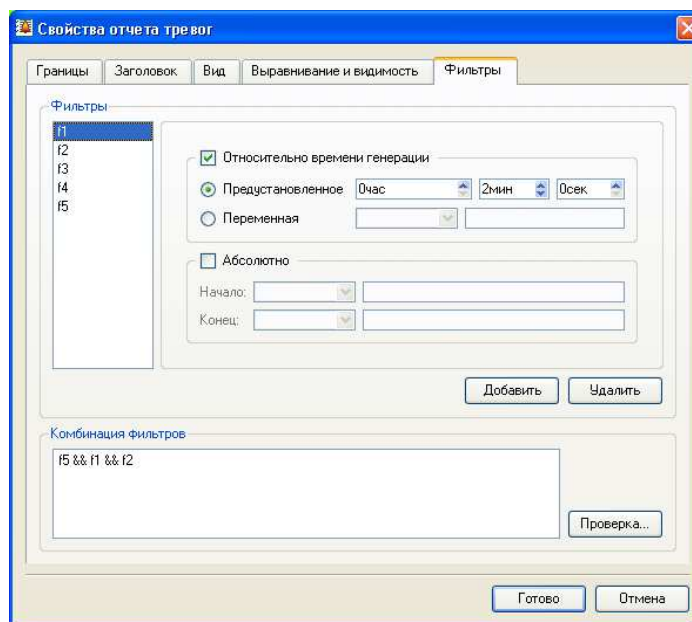




Конфигурирование фильтра по квитированию:



Конфигурирование фильтра по времени идентично заданию диапазона выборки из архива для тренда (см. Вкладка 'Архив' в разделе Вставка тренда):




Временная фильтрация производится без учета перехода на зимнес/летнее время.

Для применения сконфигурированных фильтров нужно задать выражение их логического комбинирования в разделе **Комбинация фильтров**. В выражении используются имена фильтров, операторы (&& – AND, || – OR, ! – NOT) и круглые скобки.

Для проверки заданного выражения нужно нажать кнопку **Проверка**. Если выражение корректно, на экране появится соответствующее сообщение; в противном случае – список ошибок.

Вставка документа

Возможность вставки документа в документ позволяет генерировать отчеты (далее – главные отчеты), содержащие данные нескольких отчетов, сгенерированных ранее, а также данные произвольных HTML-файлов.

По команде  **Вставить документ** на экране появляется диалог выбора типа вложенного документа. Доступные типы:

- **Документ MPB** – отчет, сгенерированный ранее. В этом случае в диалоге выбирается аргумент (к аргументу должен быть привязан канал CALL.Document);
- **Документ из файла** – в этом случае в диалоге выбирается/задается произвольный HTML-файл. Если указан несуществующий

щий файл, в главный отчет вставляется строка пробелов.

В отличие от генерации обычного документа, в ходе которой данные записываются непосредственно на диск, при генерации главного отчета каждый вложенный документ сначала полностью загружается в память. Поэтому, если вложенный документ имеет слишком большой размер, памяти может не хватить для выполнения операции.

Особенности взаимодействия МРВ с документом/экраном

В данном разделе приведены особенности взаимодействия МРВ с документом/экраном.

Ошибки взаимодействия МРВ с документом/экраном приведены в разделе **Ошибки выборки данных по запросу экрана/документа**.

Особенности извлечения данных монитором по запросу документа/экрана

При запросе тренда документ/экран создает буфер для данных. Если количество данных, извлеченных монитором по запросу, превышает размер этого буфера, монитор проводит режекцию извлеченных данных по некоторому алгоритму (настройка этого алгоритма недоступна для пользователя).

МРВ может обращаться за данными к удаленному узлу.

Аргументы документа/экрана могут быть не привязаны или привязаны к каналам, которые архивируются в разные архивы.

Скорость выборки данных из одного архива выше, чем из нескольких.

К аргументу документа/экрана может быть привязан удаленный канал (в этом случае создается локальный СНС-канал 71.3 – см. **Связь через аргументы**), а также локальный канал, привязанный к удаленному каналу.

МРВ обращается за данными к удаленному узлу в следующих случаях:

- на локальном узле архив не сконфигурирован, а локальный канал создан с помощью перетаскивания удаленного канала (см. **Копирование и вставка объекта структуры**), у которого установлен флаг архивирования (в это случае в создаваемом локальном канале флаг архивирования устанавливается автоматически);
- если есть хотя бы один аргумент, к которому привязан удаленный канал.

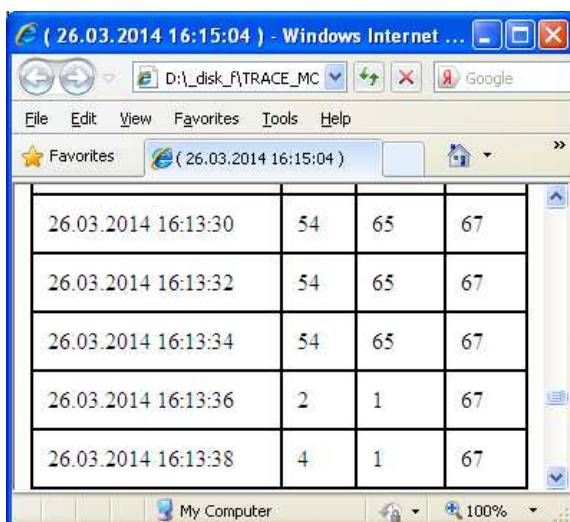
Особенности отображения данных в документе/графике

См. также **Номер SubNum**.

В документе и графике используются следующие обозначения:

- ? – недостоверность;
- ... – нет данных. Такое обозначение в графике после значения индицирует то, что значение не изменилось (в отличие от графики, в документе в этом случае всегда само значение дублируется):

| Время | saw | ТС5 |
|-------------------|-----|-----|
| 26/03/14 16:09:00 | 26 | 58 |
| 26/03/14 16:10:00 | 54 | 65 |
| 26/03/14 16:11:00 | ... | ... |
| 26/03/14 16:12:00 | ... | ... |
| 26/03/14 16:13:00 | ... | ... |
| 26/03/14 16:14:00 | 26 | 7 |
| 26/03/14 16:15:00 | 85 | 21 |
| 26/03/14 16:16:00 | ... | ... |



| | | | |
|---------------------|----|----|----|
| 26.03.2014 16:13:30 | 54 | 65 | 67 |
| 26.03.2014 16:13:32 | 54 | 65 | 67 |
| 26.03.2014 16:13:34 | 54 | 65 | 67 |
| 26.03.2014 16:13:36 | 2 | 1 | 67 |
| 26.03.2014 16:13:38 | 4 | 1 | 67 |

Если один из столбов таблицы срезов привязан к аргументу OUTPUT, не имеющему привязки, то в этот столбец выводится сумма значений срезов (сумма по строке).

Левый столбец таблицы архивных значений содержит время значения (см. ГЭ 'Архивная таблица' и Конфигурирование таблицы архивных значений).

Чтобы этот столбец отображал временной интервал, используется следующая конфигурация:

- в документе задается **ARG_<NNN>.244**, в экране – **ARG_<NNN>**;
- к **ARG_<NNN>** привязывается **call.244**, где **call** – сам канал вызова документа/экрана.

В графике правое время интервала может не отображаться.

Вывод значения времени конфигурируется в графике следующим образом:

- **Формат = Generic;**
- расшифровка формата – см. **Формат Си вывода даты и времени;**
- аргумент должен иметь временной тип данных.

Пример

| Текст [<4> ARG_004] <текст> | |
|-----------------------------|-------------------|
| Вид индикации | Значение |
| Привязка | <4> ARG_004 |
| Формат | Generic |
| Generic | %d-%B-%Y %H:%M:%S |

26-Март-2014 08:05:20

Вывод значения времени конфигурируется в документе следующим образом:

@(<аргумент>, %T <Формат Си вывода даты и времени>)

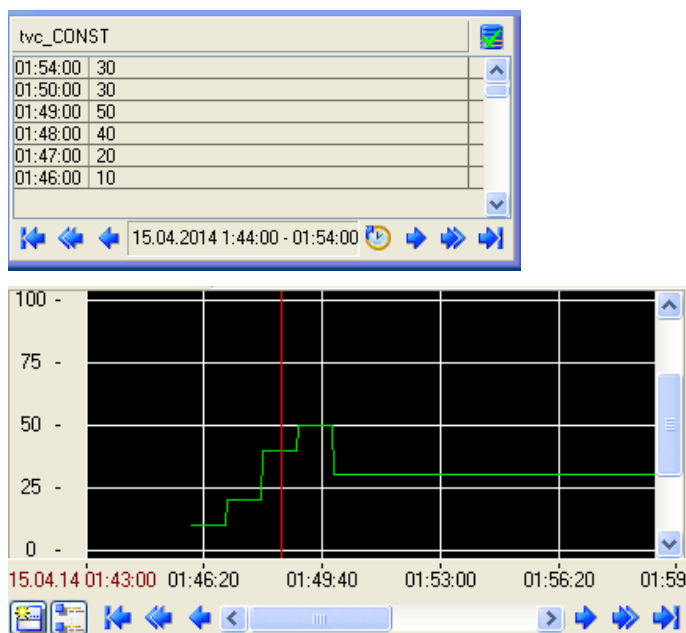
Пример

@(ARG_001, %T %d %B %Y)

В документ будет выведено:

26 Март 2014

Если к столбцу таблицы (см. **ГЭ 'Архивная таблица 2'**) или кривой архивного тренда привязан аргумент экрана/документа, к которому привязан атрибут **0,R** канала **CALL.TVC** с одной кривой (см. **Канал CALL.TVC, Содержимое аргументов = ...**), кривая отображается в таблице (на тренде).



К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан атрибут 0/2 канала CALL.Vector, CALL.ChGroupReq или CALL.AS_DATA (**Содержимое аргументов = trend**).

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан атрибут 0, 140-186 канала CALL.Vector, CALL.ChGroupReq или CALL.AS_DATA (**Содержимое аргументов = snar**). Значение отображается в одной строчке:

- атрибут 0 – выводится сумма аргументов;
- атрибут 140-186 – выводится заданный аргумент.

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан **Канал класса СОБЫТИЕ**.

Если стек разрешен, в ячейке отображаются все события на данный момент времени, начало события отображается красным (только в документе).

Таблица срезов (графика и документ).

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан другой аргумент этого же экрана/документа:

- DATE_AND_TIME – время данных другого аргумента;
- DATE/TIME – время данных другого аргумента (необходимо для столбца задать формат времени).

К столбцу таблицы может быть привязан аргумент экрана/документа без привязки:

- BOOL – пустой столбец;
- UCHAR/SCHAR – приблизительное время в мс чтения среза;
- UINT/SINT – номер строки;
- DATE_AND_TIME – двойное время (10:00-11:00);
- DATE/TIME – время среза (необходимо для столбца задать формат времени).
- FLOAT/DOUBLE_FLOAT – для вычисления сумм (число аргументов, найденных с начала таблицы):
- 1 – INPUT – SummaInput; OUTPUT – SummaOutput;
- 2 – INPUT – (SummaOutput-SummaOutput); OUTPUT – $100 * (\text{SummaOutput} - \text{SummaOutput}) / \text{SummaOutput}$;
- 3 – INPUT – $100 * (\text{SummaOutput} - \text{SummaOutput}) / \text{SummaOutput}$

Вычисляются две суммы – отдельно для входных и выходных аргументов.

К столбцу таблицы может быть привязан аргумент FLOAT экрана/документа, к которому привязан атрибут 9, Q канала CALL.ChGroupReq (**Содержимое аргументов = trend**).

Для вычисления сумм число аргументов канала должно быть равно числу строк в таблице, а аргументы равны числу найденных с начала таблицы:

- 1 – INPUT – SummaInput; OUTPUT – SummaOutput;
- 2 – INPUT – (SummaOutput-SummaOutput); OUTPUT – $100 * (\text{SummaOutput} - \text{SummaOutput}) / \text{SummaOutput}$;
- 3 – INPUT – $100 * (\text{SummaOutput} - \text{SummaOutput}) / \text{SummaOutput}$.

К столбцу таблицы срезов может быть привязан аргумент экрана/документа, к которому привязан атрибут 140-186 канала CALL.ROOT. В этом случае используется канал, который привязан к соответствующему аргументу CALL.ROOT.

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан атрибут 124 канала CALL.TVC. В этом случае выводятся все кривые.

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан атрибут 124 канала CALL.ROOT. В этом случае выводятся все значения всех каналов CALL, которые привязаны к аргументам канала CALL.ROOT.

К столбцу таблицы может быть привязан аргумент экрана/документа, к которому привязан один из следующих атрибутов канала **CGR_main** (см. **Универсальный механизм обмена с электросчетчиками**):

- 235 – выводится усредненная мощность;
- 239 – выводится энергия за день (сумма всех тарифов).

Использование встроенных шаблонов

При генерации документов могут использоваться шаблоны, встроенные в МРВ. С помощью встроенных шаблонов выполняются функции, описанные в следующих разделах:

- Вставка значения переменной;
- Канал CALL.Vector;
- Номер SubNum;
- Выборка и обработка данных SIAD;
- Обмен по OPC;
- Универсальный механизм обмена с электросчетчиками;
- Профайлеры;
- Атрибуты каналов, отображаемые профайлером;
- Канал CALL.TVC;
- Канал класса ПОЛЬЗОВАТЕЛЬ;
- Отчеты АСКУЭ.

Глава 11

**Разработка
драйверов.
Интерфейс ТСОМ**

Драйверы обмена с контроллерами

TRACE MODE поддерживает обмен данными с наиболее распространенными контроллерами. PC-based контроллеры, работающие под управлением Micro RTM, используют для этого собственные протоколы TRACE MODE. Для остальных поддерживаемых контроллеров часть протоколов встроена в исполнительные модули TRACE MODE, а другая часть поставляется опционально в виде драйверов, оформленных как динамически загружаемые библиотеки.

Для обмена данными с устройствами, поддержка которых не реализована в TRACE MODE, надо разработать драйвер. Драйвер представляет собой обычную динамически загружаемую библиотеку (DLL), которая должна экспортировать определенный набор функций. Существует несколько типов драйверов (t11, t12, t13), отличающихся набором экспортируемых функций. MPB в процессе работы в определенной последовательности вызывает из драйвера определенные функции. Через аргументы функций передаются адреса различных переменных и буферов, заполняя которые драйвер в итоге передает значение переменных контроллера в MPB. Для разработки драйверов следует использовать MS Visual C++.

Обмен с использованием внешних протоколов не поддерживается в Micro RTM.

Ниже приводятся описания функций и фрагменты кода для MS Visual C++ (версии 5,6, .NET).

Драйвер t13

Поддержка обмена с устройствами через драйвер t13 сохранена в TRACE MODE 6 только для совместимости с версией 5, эту технологию не следует использовать для разработки новых драйверов.

Этот драйвер – драйвер с фиксированным подключением – может быть только один, он оформляется в виде библиотеки **t13.dll**. Библиотека должна быть расположена в директории, содержащей исполнительный файл MPB.

Драйвер вызывается каналами **Fast_R/W** или **R/W** (здесь и далее под такими каналами понимаются числовые каналы, связанные с соответствующими источниками/приемниками – в данном случае с переменными **Fast_R/W** и **R/W** группы **RWH_Driver** – см. Группа ‘Платы ввода-вывода’).

Первые вызывают драйвер при каждом их пересчете. При этом в зависимости от типа переменной идет обращение к функциям драйвера **t13_fread** или **t13_fwrite**. Если возвращаемое функцией значение отличается от 0, каналу устанавливается признак аппаратной недостоверности.

Каналы **R/W** вызывают драйвер при условии наличия в базе канала **R/W_Control** и отличия от 0 результата логического умножения его значения и настройки **IO** канала **R/W**. MPB запускает в отдельном потоке обработку сразу всех каналов **R/W**, удовлетворяющих описанному условию, независимо от их периодов. Для каждого из них вызываются функции **t13_read** или **t13_write** (в зависимости от типа переменной). Каждая из них обрамляется вызовами **t13_open** и **t13_close**. Поток обработки каналов **R/W** работает параллельно с пересчетом базы каналов, но имеет более низкий приоритет.

Описание функций в драйвере выглядит следующим образом:

```
typedef union
{
    unsigned char c[6];
    unsigned short int i[3];
}
IA;
__declspec(dllexport) void t13_start();
__declspec(dllexport) void t13_stop();
__declspec(dllexport) void t13_open();
__declspec(dllexport) void t13_close();
__declspec(dllexport) int t13_write(char *name, IA &ia, float v);
__declspec(dllexport) int t13_read(char *name, IA &ia, float &v);
```

```

__declspec(dllexport) int t13_fwrite(char *name, IA &ia, float
v);
__declspec(dllexport) int t13_fread(char *name, IA &ia, float
&v);

```

где

name – имя канала;

ia – удаленный адрес, шестибайтовое число, составленное из настроек переменной **I0** (слово), **I1**(слово), **C2** (байт) и **C3** (байт);

v – значение канала. МРВ передает драйверу реальное значение канала.

Ниже приведен простейший драйвер. Он записывает все выходные параметры на диск **E:** в файл с именем **aaa**, а входным параметрам присваивает значение текущего системного времени в секундах.

```

FILE *fj;
void t13_start()
{
}
void t13_stop()
{
}
void t13_open()
{
    fj=fopen("e:\\aaa", "a");
}
void t13_close()
{
    fclose(fj);
}
int t13_write(char *name, IA &ia, float v)
{
    fprintf(fj, "%s %g\n", name, v);
    return(0);
}
int t13_read(char *name, IA &ia, float &v)
{
    v=(time(NULL) & 0x00ff);
    return(0);
}
int t13_fwrite(char *name, IA &ia, float v)
{
    return(0);
}
int t13_fread(char *name, IA &ia, float &v)
{
    return(0);
}

```

Редактор параметров переменных **Fast_R/W**, **R/W**, **R/W_Control** показан на рисунке:

Подтип и дополнение к подтипу каналов t13:

- **Fast_R/W** – 0,10;
- **R/W** – 0,9;
- **R/W_Control** – 0,8.

t13.dll 6.09

Канал (0,9) **R/W** (медленный) воспринимает атрибуты так же, как канал **Fast_R/W** (быстрый): раньше параметры **C2** и **C3** из редактора источника/приемника записывались в атрибуты (92, **I2**) и (93, **I3**), теперь – в атрибуты (95, **C2**) и (96, **C3**), а параметры **I0** и **I1** из редактора источника/приемника – в атрибуты (90, **I0**) и (90, **I1**) канала.

Поддержан механизм **request/response/DataReady** (см. описание атрибутов (52, **FS**) и (120, **ACK**) в разделе **Атрибуты каналов, отображаемые профайлером**).

Введен механизм чтения/записи в **CALL.ChGroupReq**. При вызове атрибут (98, **C5**) принимает значение номера аргумента, возвращаемое значение интерпретируется через тип аргумента.

Для канала (0,8) управления медленными каналами (0,9) имеет значение атрибут (90, **I0**):

- **I0=0** – для задания маски;
- **I0=1** – изменение всех (0,9).C4, равных (0,8).C4, на реальное зна-

чение;

- **IO=2** – изменение всех (0,9).I1, равных (0,8).I1, на реальное значение.

Если (0,8) является каналом CALL.ChGroupReq, действия происходят над каналами, привязанными к аргументам, – без проверки равенства атрибутов и реальном значении канала больше 0. Доступен случай 3, где (0,9).C4=(0,8).C4 и (0,9).I1=(0,8).I1.

Функции t13.dll изменены.

Драйверы t11 и t12

Для упрощения разработки драйверов обмена по последовательным портам MPV реализует самостоятельно все функции обмена с портами (с некоторыми ограничениями). Пользовательский драйвер t11 должен только сформировать посылаемые сообщения и расшифровать ответ. Такой драйвер оформляется в виде библиотеки **t11s<n>.dll**. Значение **n** должно соответствовать номеру, зарезервированному для пользовательских драйверов t11 (см. файл **%TRACE MODE% \ Drivers_with_Setup \ drivers.txt**).

Если обмен данными с контроллером осуществляется не по стандартным последовательным интерфейсам, или для связи с ним имеются промежуточные средства, то кроме модуля, описывающего протокол обмена, необходимо разработать также модуль, описывающий интерфейс. Описание протокола должно быть реализовано в виде модуля **t12s<n>.dll**, описание интерфейса – в виде модуля **media<n>.dll**. Значение **n** должно совпадать у **t12s<n>.dll** и **media<n>.dll** и, кроме того, соответствовать номеру, зарезервированному для пользовательских драйверов t12 (см. файл **drivers.txt**).

Мониторы TRACE MODE 6 поддерживают два интерфейса взаимодействия с драйверами t11 и t12 – **TCOM5** и **TCOM6**. Драйверы, разработанные в соответствии с интерфейсом **TCOM5**, совместимы с версией 5 TRACE MODE.

Интерфейс **TCOM6** представляет собой расширенную версию интерфейса **TCOM5**. При использовании **TCOM6** доступно использование шестого байта удаленного адреса и передача строкового параметра, заданного в поле **Дополнительно** редактора пользовательского драйвера (см. **Удаленный адрес и разновидности драйверов**). Драйверы **TCOM6** могут быть подключены к TRACE MODE, начиная с версии 6.02.1.

MPV определяет интерфейс драйвера по количеству экспортируемых им функций. При использовании **TCOM6** библиотека драйвера (как t11, так и t12) должна экспортировать 7 функций. Функция, экспортируемая под номером 7 (MPV вызывает функции внешних драйверов по номеру), никогда не вызывается и служит только для определения интерфейса, поэтому может иметь любое имя.

Если драйвер экспортирует меньше 7 функций (5 или 6 для t11 и 6 для t12), MPV использует интерфейс **TCOM5**.

Начиная с релиза 6.07, настоятельно рекомендуется использовать только интерфейс **TCOM6**.

Библиотеки драйверов t11 и t12 должны быть размещены в директории,

содержащей исполнительный модуль MPB.

MPB поставляется с подключенным набором драйверов – они перечислены в файле **drivers.txt**. Канал вызова драйвера имеет номер дополнения к подтипу (см. **Подтипы каналов**), который соответствует номеру драйвера, указанному в файле **drivers.txt**.

Здесь и далее под каналами вызова драйвера понимаются каналы, связанные с соответствующими переменными слоя **Источники/Приемники** (см. **Назначение групп источников (приемников)**).

Удаленный адрес и разновидности драйверов

При обработке канала MPB вызывает драйвер, соответствующий его дополнению к подтипу. При этом драйверу через структуру **IA** (см. заголовочный файл) передается **удаленный адрес**. В нем указываются данные, которые надо запросить у контроллера, или переменная контроллера, в которую надо переслать значение канала.

Удаленный адрес включает в себя шесть байт. Его можно разбить на несколько полей (до шести). Каждое поле может включать в себя один или два байта. Значения полей формируются атрибутами переменной, с которой связан канал вызова драйвера. Возможные сочетания разрядности атрибутов образуют 4 разновидности драйверов t11 и t12 (отличия драйверов **ТСОМ5** и **ТСОМ6** описаны в разделе **Драйверы t11 и t12**):

| Вид драйвера | Атрибут 1 | Атрибут 2 | Атрибут 3 | Атрибут 4 | Атрибут 5 | Атрибут 6 |
|--------------|-----------|-----------|-----------|-----------|-----------|-----------|
| 1 | BYTE | BYTE | WORD | WORD | - | - |
| 2 | BYTE | BYTE | WORD | BYTE | BYTE | - |
| 3 | BYTE | BYTE | BYTE | BYTE | WORD | - |
| 4 | BYTE | BYTE | BYTE | BYTE | BYTE | BYTE |

Наименования разработанных драйверов t11 и t12 в соответствии с **%TRACE MODE% \ Drivers_with_Setup \ drivers.txt** должны быть следующими:

| Вид драйвера | t11 | t12 |
|--------------|------------|-----------------------|
| 1 | t11s27.dll | t12s4.dll, media4.dll |
| 2 | t11s28.dll | t12s5.dll, media5.dll |
| 3 | t11s29.dll | t12s6.dll, media6.dll |
| 4 | t11s30.dll | t12s7.dll, media7.dll |

К системе можно подключить только по одному пользовательскому драйверу каждого вида.

В редакторе переменной разрядность атрибутов указана:

Два младших байта структуры **IA** (**IA.c[0]** и **IA.c[1]**) влияют также на формирование блоковых (групповых запросов). Каналы, у которых совпадают одновременно тип, подтип, дополнение к подтипу и два младших байта удаленного адреса, группируются в один блок (группу). Если хотя бы один параметр не совпадает, канал не попадает в данный блок. Это нужно учитывать при разработке структуры удаленного адреса для конкретного контроллера.

Модифицирование редактора пользовательского драйвера

Редактор пользовательского драйвера может быть модифицирован с помощью файла, располагаемого в директории ИС:

- **t11s27.frm...t11s30.frm** (соответственно для драйвера **t11** типа 1...4);
- **t12s4.frm...t12s7.frm** (соответственно для драйвера **t12** типа 1...4).

Файл ***.frm** имеет текстовый формат (UTF-8) и состоит из строк описания имен атрибутов и способов их задания. Строка имеет следующий формат:

```
nField = "fieldName", fieldInputType;
```

где

- **nField** – адрес атрибута в массиве однобайтных параметров в структуре удаленного адреса, начиная с 0 (0...5), или специальный адрес 7;
- **fieldName** – имя атрибута,
- **fieldInputType** – способ задания значения атрибута:
 - 0 – ввод 1-байтового числа в формате DEC;
 - 2 – ввод 2-байтового числа в формате DEC;
 - 1 – ввод 1-байтового числа в формате HEX,

- 3 – ввод 2-байтового числа в формате HEX,
- 4 – выбор значения из списка,
- 5 – ввод строки (только для специального адреса 7).

Список значений для выбора из раскрывающегося меню задается после указания способа задания (4) в следующем формате:

("type1"=1, "type2"=2, "type5"=5)

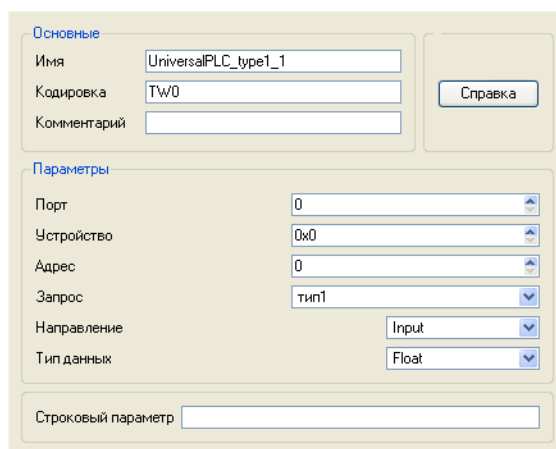
Элементы списка разделяются запятой, в кавычках указывается строка для отображения, а число задает значение, записываемое в структуру удаленного адреса. Если число не указано, он равно предыдущему числу, увеличенному на 1. Если число не задано для первого элемента списка, оно принимается равным 0.

Специальное значение **nField** = 7 соответствует строковому атрибуту (в отсутствие файла ***frm** этот атрибут имеет имя **Дополнительно**).

В качестве примера разместим в директории ИС следующий файл **t11s27.frm**:

```
0 = "Порт", 0;
1 = "Устройство", 1;
2 = "Адрес", 2;
4 = "Запрос", 4 ("тип1"=1, "тип2", "тип5"=5);
7 = "Строковый параметр", 5;
```

Редактор драйвера **t11** типа 1 примет следующий вид (ср. с редактором по умолчанию, показанным выше):



TCOM5. Драйвер t11

TCOM5. Функции драйвера t11, вызываемые MPB

Все перечисленные ниже функции (за исключением **zReadAny_xxx**) обязательно должны быть определены в драйвере.

TCOM5 Prepare_xxx. Первое обращение к драйверу

Эта функция вызывается при инициализации работы MPB с драйвером. В ней можно прописать операции установки начальных условий для обмена данными, а также полную инициализацию драйвера. В частности, из этой функции можно передать MPB указание сформировать блоковые запросы (не более 128 каналов).

```
int Prepare_xxx(int type, char *str, int
&type_cnv, int &q_out);
```

где

type – номер драйвера;

str – передаваемая драйвером строка, содержащая название протокола (не более 31 байта). Профайлер пишет эту строку в свой протокол;

type_cnv – признак формирования блоковых запросов. Эта константа задается при разработке драйвера; при вызове функции передается MPB:

0 – формировать;

не 0 – не формировать;

q_out – ограничение на величину буферов для приема и передачи данных в контроллер (передается в драйвер, но может быть изменено). Максимальное значение – 2048 байт.

Возвращаемое функцией значение, равное 0, говорит о нормальном ее выполнении. Отличие возвращаемого значения от 0 воспринимается как ошибочное завершение. В этом случае каналы с соответствующим дополнением к подтипу отключаются.

TCOM5 zCompare_xxx. Функция сравнения

Эта функция используется при формировании и расшифровке блоковых запросов. Она определяет принадлежность каналов к блокам (группам).

```
int zCompare_xxx(IA &ia, IA &ia1, int &count);
```

где

ia – удаленный адрес первого канала;

ia1 – удаленный адрес сравниваемого канала;

count – число каналов, уже попавших в текущий блок. Этот параметр формируется МРВ и передается в драйвер. По нему можно внести ограничение на добавление новых элементов в блоковый запрос. Для алгоритма **DATA11** (при расшифровке ответа) этот параметр равен индексу элемента массива структур **RSDATA**, содержащего значение для записи в канал. Этот массив заполняется функцией **Get_xxx**.

В процессе формирования блоковых запросов функция всегда вызывается для двух *различных* каналов. При этом **count** ≥ 1 . При разборе блокового запроса при первом вызове функции для данного блока аргументы **ia** и **ia1** оба относятся к первому каналу блока, при этом параметр **count** равен нулю. Если возвращаемое данной функцией значение больше 0, канал принадлежит к данному запросу; если при этом осуществляется обработка группового запроса по алгоритму **BLOCKDATA11**, возвращаемое значение должно быть на 1 больше индекса размещения данных в массиве структур **RSDATA**.

Если возвращаемое данной функцией значение равно 0, то анализируемый канал не попадает в данный запрос.

TCOM5 Set_xxx. Функция формирования сообщения

Эта функция вызывается МРВ при пересчете значений канала, связанного с драйвером.

```
void Set_xxx(IA &ia, unsigned int &max_send, unsigned int &max_rec, int &q_rec, RSDATA *p, char *sbuf);
```

где

ia – удаленный адрес канала (передается в драйвер);

max_send – количество байтов для отправки (формируется в драйвере);

max_rec – количество байтов в ответе (формируется в драйвере);

q_rec – число посылаемых или запрашиваемых значений (передается в драйвер);

p – указатель на массив структур **RSDATA**, содержащий данные и их форматы. Этот параметр требуется, если канал имеет тип OUTPUT, – в этом случае элемент 0 массива содержит значение для

записи в контроллер.

Если канал типа **OUTPUT**, то

```
p[0].F.fmt[3]=1;
```

для данных в формате **FLOAT**:

```
p[0].V.v – посылаемое значение,
```

```
p[0].F.fmt[0]=0;
```

для данных в формате **HEX**:

```
p[0].V.i[0] – посылаемое значение,
```

```
p[0].F.fmt[0]=0x40.
```

Тип значения (**HEX** или **FLOAT**), посылаемого из канала в устройство, не связан с классом канала, который пользователь указал в редакторе базы каналов. **p[0].F.fmt[0]** нужен только для корректной интерпретации **p[0].V** внутри драйвера.

Если канал типа **INPUT**, то

```
p[0].F.fmt[3]=0.
```

sbuf – буфер для отправки (формируется в драйвере). Память под этот буфер выделяет МРВ, драйверу требуется только заполнить его нужными значениями. Длина буфера передается драйверу при вызове функции **Prepare_xxx** через ее аргумент **q_out**.

Данная функция вызывается для формирования сообщений, посылаемых по последовательному интерфейсу в контроллер. Сформированное сообщение должно быть размещено в буфере **sbuf**.

С помощью функции **Set_xxx** можно установить значение каналу, обратившемуся к драйверу. Если эта функция возвращает **max_send=max_rec**=[младший байт буфера **sbuf**]=0, то МРВ присваивает каналу значение, взятое из первого элемента массива структур **RSDATA**.

TCOM5 zReadAny_xxx. Функция приема произвольного числа символов

Данная функция предназначена для организации приема с заранее не известным числом символов.

```
int zReadAny_xxx(IA &ia, unsigned int &count_rec, unsigned int &all_c, char *rbuf);
```

где

ia – удаленный адрес канала;

count_rec – указывает, сколько байтов еще принять;

all_c – указывает, сколько байтов принято всего;

rbuf – буфер приема.

Если драйвер поддерживает эту функцию, то в протокол профайлера пишется строка

RS: Loaded Protocol (5.11) <название протокола>

Функция вызывается МРВ после функции **Set_xxx**. Если **zReadAny_xxx** возвращает 0, то считается, что все байты от устройства приняты и далее будет вызвана функция **Check_xxx**. Если **zReadAny_xxx** возвращает не 0, то МРВ примет еще **count_rec** байтов от устройства и опять вызовет **zReadAny_xxx**. Таким образом можно получить от устройства сообщение, длина которого заранее (т.е., на момент вызова **Set_xxx**) не известна.

TCOM5 Check_xxx. Функция проверки

Эта функция вызывается после приема ответа от контроллера перед функцией расшифровки полученных данных. Она проверяет корректность полученного ответа. Она имеет следующий формат:

```
int Check_xxx(IA &ia, unsigned int &count_rec, unsigned int &max_rec, unsigned int &max_send, char *rbuf);
```

где

ia – удаленный адрес канала (передается в драйвер);

count_rec – число принятых байтов (передается в драйвер);

max_rec – количество байтов в ответе (передается драйверу и формируется в нем);

max_send – количество байтов для отправки (передается драйверу и формируется в нем);

rbuf – буфер, содержащий принятое сообщение (передается драйверу и формируется в нем).

Если возвращаемое функцией значение равно 0, то принятые данные считаются корректными и осуществляется вызов функции расшифровки принятого сообщения.

При отличии возвращаемого функцией значения от 0 анализируется величина параметра **max_rec**. Если он больше 0, то осуществляется прием дополнительных байтов. Количество этих байтов равно значению параметра **max_rec**.

Если возвращаемое значение параметра **max_rec** равно 0, то каналу, для которого осуществлялся вызов драйвера, устанавливается признак аппа-

ратной недостоверности. Если при этом значение параметра **max_send** больше 0, то осуществляется посылка по последовательному порту строки из буфера **rbuf**. Количество посылаемых байтов равно величине параметра **max_send**.

TCOM5 Get_xxx. Функция расшифровки сообщения

Эта функция вызывается для расшифровки полученного от контроллера ответа. Она имеет следующий формат:

```
int Get_xxx(IA &ia, unsigned int &count_rec, int
&q_rec, RSDATA *p, char *rbuf, int &type_cnv);
```

где

- ia** – удаленный адрес канала (передается в драйвер);
- count_rec** – число принятых символов (передается в драйвер);
- q_rec** – число посылаемых или запрашиваемых значений (передается в драйвер, может быть изменено);
- p** – указатель на массив структур **RSDATA**, содержащий данные и их форматы;
- rbuf** – буфер, содержащий принятые символы (передается в драйвер);
- type_cnv** – этот параметр формируется в драйвере, его значение соответствует типу разборки данных (см. ниже).

Значения, полученные от контроллера, должны быть присвоены элементам массива **p** следующим образом: **p[i].V.v = <значение>** (даже если значение целое).

Возвращаемое значение указывает время ожидания до следующего запроса (в миллисекундах) по данному порту. Если функция **Check_xxx** возвращает значение, отличное от 0, перехода к **Get_xxx** не произойдет, и повторный запрос уйдет без задержки.

TCOM5. Заголовок драйвера t11

Ниже приведен стандартный заголовок драйвера t11 TCOM5.

```
typedef union
{
    unsigned char c[6];
    unsigned short int i[3];
}
IA;
typedef union
{
    unsigned char c[4];
    unsigned short int i[2];
```

```

        unsigned long l;
        float f;
    }
        FOUR_BYTE;
typedef union
{
    unsigned char c[2];
    unsigned short int i;
}
        TWO_BYTE;
typedef struct
{
    union
    {
        float v;
        unsigned char c[4];
        unsigned short int i[2];
        int d;
    }
    V;
    union
    {
        unsigned char fmt[4];
        unsigned short int ind[2];
    }
    F;
}
        RSDATA;
#define ERR_RT_FILE      1 // ошибка открытия файла
#define ERR_RT_SEEK     2 // ошибка поиска файла
#define ERR_RT_WRITE    3 // ошибка записи файла
#define ERR_RT_READ     4 // ошибка чтения файла
#define ERR_RT_MEM      5 // недостаточно памяти
#define ERR_RT_LIST     6 // ошибка создания списка
#define ERR_RT_FORMAT   7 // ошибка в формате
#define ERR_RT_COUNT    8 // неправильный счетчик
#define ERR_RT_TIMEOUT  9 // обнаружен таймаут
#define ERR_RT_RESP     10 // неправильный ответ
#define ERR_RT_FUNC     11 // код ошибки API WIN32
#define ERR_RT_NOTFOUND 12 // запрос неопределенного канала
#define ERR_RT_CODE     13 // код ошибки контроллера
#define ERR_RT_FSC      14 // неправильная контрольная сумма
__declspec(dllexport) int Prepare_xxx(int type,
    char *str,
    int &type_cnv,
    int &q_out);
__declspec(dllexport) void Set_xxx(IA &ia,
    unsigned int &max_send,
    unsigned int &max_rec,
    int &q_rec,
    RSDATA *p,
    char *sbuf);
__declspec(dllexport) int Get_xxx(IA &ia,
    unsigned int &count_rec,
    int &q_rec,
    RSDATA *p,
    char *rbuf,
    int &type_cnv);
__declspec(dllexport) int Check_xxx(IA &ia,
    unsigned int &count_rec,

```

```

        unsigned int &max_rec,
        unsigned int &max_send,
        char *rbuf);
__declspec(dllexport) int zCompare_xxx(IA &ia0,
IA &ia1,
int &count);

```

TCOM5. Примеры драйверов t11

Драйвер для контроллера Ш711/1. TCOM5

```

int first[16]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
int Prepare_xxx(int type,char *str,int &type_cnv,int &q_out)
{
    strcpy(str,"SH-711-60");
    return(0);
}
void Set_xxx(IA &ia, unsigned int &max_send, unsigned int &max_rec, int
&q_rec, RSDATA *p,char *sbuf)
{
    if (first[ia.c[0]]==0)
    {
        sbuf[0]=0x7c;
        max_send=1;
        max_rec=0;
        first[ia.c[0]]=1;
    }
    else
    {
        sbuf[0]=0x67; //0x72 - 80
        max_send=1;
        max_rec=11+14*64;
    }
    return;
}
int Get_xxx(IA &ia,unsigned int &count_rec,int &q_rec,RSDATA *p,char
*rbuf,int &type_cnv)
{
    int i,h;
    q_rec=64;
    for (i=0;i<q_rec;i++)
        p[i].F.fmt[1]=1;
        i=11;
        while(i<count_rec)
        {
            h=rbuf[i]-31+((rbuf[i+1]-35)/20)*15; // ch from S711
            i+=2;
            if (h<q_rec)
            {
                if ( (rbuf[i+1]<'0') || (rbuf[i+1]>'9') )
                    p[h].F.fmt[1]=1;
                else
                {
                    p[h].F.fmt[1]=0;
                    rbuf[i+7]=0;
                    p[h].V.v=(float)atof(rbuf+i);
                }
            }
        }
}

```

```

    }
    i+=12;
    }
    type_cnv=3;
    return(500);
}
int Check_xxx(IA &ia,unsigned int &count_rec,unsigned int
&max_rec,unsigned int &max_send,char *rbuf)
{
    if (count_rec>11)
        return(0);
    else
    {
        max_rec=0;
        max_send=0;
        first[ia.c[0]]=0;
        return(ERR_RT_RESP);
    }
}
int zCompare_xxx(IA &ia0,IA &ia1,int &count)
{
    return(ia1.i[1]+1);
}

```

Драйвер для контроллера OMRON по протоколу HOSTLINK. TCOM5

```

int Prepare_xxx(int type,char *str,int &type_cnv,int &q_out)
{
    strcpy(str,"Omron");
    type_cnv=0;
    return(0);
}
void Set_xxx(IA &ia,unsigned int &max_send,unsigned int &max_rec,int
&q_rec,RSDATA *p,char *sbuf)
{
    unsigned short int err;
    int i;
    sbuf[0]='@';
    max_send=13;
    sprintf(sbuf+1,"%.2d",ia.c[1]); //units
    sprintf(sbuf+5,"%.4d",ia.i[1]);
    switch(ia.c[4])
    {
        case 0: sbuf[4]='R'; break; //IR
        case 1: sbuf[4]='H'; break; //HR
    }
    if (p[0].F.fmt[3]==1) // write
    {
        sbuf[3]='W';
        if (p[0].F.fmt[0]==0)
            sprintf(sbuf+9,"%x", (unsigned short
            int)p[0].V.v);
        else
            sprintf(sbuf+9,"%x",p[0].V.i[0]);
        max_rec=11;
    }
}

```

```

    }
    else // read
    {
        sbuf[3]='R';
        sprintf(sbuf+9,"%d",q_rec+1);
        max_rec=11+((q_rec+1)<<2);
    }
    err=0;
    for (i=0;i<max_send;i++)
        err ^=sbuf[i];
        sprintf(sbuf+max_send,"%d",err);
        max_send+=2;
        sbuf[max_send]='*';
        max_send++;
        sbuf[max_send]=13;
        max_send++;
        return;
    }
int Get_xxx(IA &ia,unsigned int &count_rec,int &q_rec,RSDATA *p,char
*rbuf,int &type_cnv)
{
    int i,j;
    char good;
    if ( p[0].F.fmt[3]==1) // out
    {
        p[0].F.fmt[1]=0;
        return(0);
    }
    i=7;q_rec=-1;
    while(i<count_rec)
    {
        good=rbuf[i+5]; rbuf[i+5]=0;
        sscanf(rbuf+i,"%x",&j);
        q_rec++;
        p[q_rec].V.v=(float)j;
        i+=4;
        rbuf[i]=good;
    }
    type_cnv=0;
    return(0);
}
int Check_xxx(IA &ia,unsigned int &count_rec,unsigned int
&max_rec,unsigned int &max_send,char *rbuf)
{
    if ( count_rec < max_rec )
    {
        max_send=max_rec=0;
        return(ERR_RT_TIMEOUT);
    }
    else
        if ( (rbuf[5]!=48) || (rbuf[6]!=48) )
        {
            max_send=max_rec=0;
            return(ERR_RT_RESP);
        }
        return(0);
}
int zCompare_xxx(IA &ia0,IA &ia1,int &count)
{
    if ( (count < 28) && (ia0.c[4]==ia1.c[4]) && (ia0.i[1]+count

```

```
== ial.i[1]) )  
    return(1);  
else  
    return(0);  
}
```


TCOM6. Драйвер t11

TCOM6. Функции драйвера t11, вызываемые MPB

В данном разделе описаны отличия функций **TCOM6** от одноименных функций **TCOM5**. Общие параметры функций описаны в разделе **TCOM5**.
Функции драйвера t11, вызываемые MPB.

TCOM6 Prepare_xxx. Первое обращение к драйверу

```
int Prepare_xxx(int type, char *str, int
&type_cnv, int &q_out);
```

Данная функция идентична функции **Prepare_xxx TCOM5**.

TCOM6 zCompare_xxx. Функция сравнения

```
int zCompare_xxx(int runtime, IA &ia0, IA &ia1,
int &count, void *ext_data0, void *ext_data1);
```

Параметры:

- **runtime** – 0, если функция вызывается при старте MPB на этапе формирования блоковых запросов; не 0 при разборе блоковых запросов;
- **ia0** – удаленный адрес первого канала;
- **ia1** – удаленный адрес сравниваемого канала;
- **count** – число каналов, уже попавших в текущий блок;
- **ext_data0** – дополнительная информация для первого канала;
- **ext_data1** – дополнительная информация для сравниваемого канала.

Параметр **ext_data** является указателем на строку, заданную пользователем в поле **Дополнительно** редактора, т.е.

```
const char* szExtString0 = *(char**)ext_data0;
```

Параметры **ia0**, **ia1**, **count** имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 Set_xxx. Функция формирования сообщения

```
void Set_xxx(IA &ia, unsigned int &max_send, unsigned int &max_rec, int &q_rec, RSDATA *p, char
```

```
*sbuf, void *ext_data);
```

Параметры:

- **ia** – удаленный адрес канала;
- **max_send** – количество байтов для отправки;
- **max_rec** – количество байтов в ответе;
- **q_rec** – число посылаемых или запрашиваемых значений;
- **p** – указатель на массив структур RSDATA, содержащий данные и их форматы. Этот параметр требуется, если канал имеет тип OUTPUT, – в этом случае элемент 0 массива содержит значение для записи в контроллер;
- **sbuf** – буфер для отправки;
- **ext_data** – дополнительная информация для канала.

Если канал имеет тип OUTPUT, элемент 0 массива **p** содержит значение для записи в контроллер. Формат значения определяется флагом **p[0].F.fmt[0]**:

- **p[0].F.fmt[0]=0** для данных в формате FLOAT, тогда **p[0].V.v** содержит посылаемое значение,
- **p[0].F.fmt[0]=0x40** для 16-разрядных целочисленных данных; **p[0].V.i[0]** – посылаемое значение,
- **p[0].F.fmt[0]=0x80** для 32-разрядных целочисленных данных; **p[0].V.d** – посылаемое значение.

Параметр **ext_data** является указателем на строку, заданную пользователем в поле **Дополнительно** редактора, т.е.

```
const char* szExtString = *(char**)ext_data;
```

Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 zReadAny_xxx. Функция приема произвольного числа символов

Данная функция идентична функции **zReadAny_xxx TCOM5**.

TCOM6 Check_xxx. Функция проверки

```
int Check_xxx(IA &ia, unsigned int &count_rec, unsigned int &max_rec, unsigned int &max_send, char *rbuf, void *ext_data, int q_rec);
```

Параметры:

- **ia** – удаленный адрес канала;

- **count_rec** – число принятых байтов;
- **max_rec** – количество байтов в ответе;
- **max_send** – количество байтов для отправки;
- **rbuf** – буфер, содержащий принятое сообщение;
- **ext_data** – дополнительная информация для канала;
- **q_rec** – число посылаемых или запрашиваемых значений.

Параметры **ext_data** и **q_rec** аналогичны одноименным параметрам функции **Set_xxx()** TCOM6. Остальные параметры имеют такое же назначение, как и в интерфейсе TCOM5.

TCOM6 Get_xxx. Функция расшифровки сообщения

```
int Get_xxx(IA &ia, unsigned int &count_rec, int
&q_rec, RSDATA *p, char *rbuf, int &type_cnv, void
*ext_data);
```

Параметры:

- **ia** – удаленный адрес канала;
- **count_rec** – число принятых символов;
- **q_rec** – число посылаемых или запрашиваемых значений;
- **p** – указатель на массив структур RSDATA, содержащий данные и их форматы;
- **rbuf** – буфер, содержащий принятые символы;
- **type_cnv** – тип разборки данных;
- **ext_data** – дополнительная информация для канала.

Параметр **ext_data** аналогичен одноименному параметру функции **Set_xxx()** TCOM6. Остальные параметры имеют такое же назначение, как и в интерфейсе TCOM5.

TCOM6. Функция, указывающая MPB использовать интерфейс TCOM6

```
void zzTM6Stub();
```

Функция не имеет параметров, служит только для указания использования интерфейса TCOM6 и никогда не вызывается.

TCOM6. Заголовок драйвера t11

Ниже приведен заголовок драйвера t11 TCOM6.

```
typedef union
{
```

```
    unsigned char c[6];
    unsigned short int i[3];
} IA;
typedef union
{
    unsigned char c[4];
    unsigned short int i[2];
    unsigned long l;
    float f;
} FOUR_BYTE;
typedef union
{
    unsigned char c[2];
    unsigned short int i;
} TWO_BYTE;
typedef struct
{
    union
    {
        float v;
        unsigned char c[4];
        unsigned short int i[2];
        unsigned long l;
        int d;
    } V;
    union
    {
        unsigned char fmt[4];
        unsigned short int ind[2];
    } F;
} RSDATA;
#define ERR_RT_FILE      1 // ошибка открытия файла
#define ERR_RT_SEEK     2 // ошибка поиска файла
#define ERR_RT_WRITE    3 // ошибка записи файла
#define ERR_RT_READ     4 // ошибка чтения файла
#define ERR_RT_MEM      5 // недостаточно памяти
#define ERR_RT_LIST     6 // ошибка создания списка
#define ERR_RT_FORMAT   7 // ошибка в формате
#define ERR_RT_COUNT    8 // неправильный счетчик
#define ERR_RT_TIMEOUT  9 // обнаружен таймаут
#define ERR_RT_RESP     10 // неправильный ответ
#define ERR_RT_FUNC     11 // код ошибки, возвращаемый API WIN32
#define ERR_RT_NOTFOUND 12 // запрос неопределенного канала
#define ERR_RT_CODE     13 // код ошибки посланный контроллером
#define ERR_RT_FSC      14 // неправильная контрольная сумма

int __declspec(dllexport) Prepare_xxx(int type, char *str, int
&blockQueryFlag, int &nBuffersSize);
```

```
int __declspec(dllexport) zReadAny_xxx(IA &ia, unsigned int &more_rec,
unsigned int &all_rec, char *rbuf);

void __declspec(dllexport) Set_xxx(IA &ia, unsigned int &max_send, un-
signed int &max_rec, int &q_rec, RSDATA *p, char *sbuf, void *ext_data);

int __declspec(dllexport) Check_xxx(IA &ia, unsigned int &count_rec, un-
signed int &max_rec, unsigned int &max_send, char *rbuf, void *ext_data,
int q_rec);

int __declspec(dllexport) Get_xxx(IA &ia, unsigned int &count_rec, int
&q_rec, RSDATA *p, char *rbuf, int &type_cnv, void *ext_data);

int __declspec(dllexport) zCompare_xxx(int runtime, IA &ia0, IA &ia1,
int &count, void *ext_data0, void *ext_data1);

void __declspec(dllexport) zzTM6Stub();
```

Алгоритм взаимодействия с драйвером t11

Алгоритм вызова драйвера t11

1. Инициализация

При старте МРВ загружает драйверы исходя из наличия каналов с соответствующими дополнениями к подтипу.

Для загруженного драйвера вызывается функция **Prepare_xxx** с соответствующим аргументом **type**.

Если задано, что блоковые запросы для данного драйвера не формируются (аргумент **type_cnv** функции **Prepare_xxx** отличен от нуля), загружается следующий драйвер.

2. Установка признака принадлежности канала к блоковому запросу

Если задано, что блоковые запросы формируются (аргумент **type_cnv** функции **Prepare_xxx** равен нулю), МРВ с помощью функции **zCompare_xxx** устанавливает каналам признаки принадлежности к таким запросам.

Вначале МРВ выполняет проход по базе с ее начала и находит канал с дополнением, соответствующим загруженному драйверу (канал-образец). Этот канал становится первым каналом блока и в дальнейшем инициирует блоковый запрос.

Далее МРВ производит сравнение всех каналов базы с каналом-образцом с помощью функции **zCompare_xxx** (эта функция вызывается для каждого анализируемого канала). В блок группируются те каналы, у которых тип, подтип, дополнение к подтипу и два младших байта удаленного адреса совпадают с аналогичными параметрами канала-образца, и **zCompare_xxx** вернула не 0. Таким образом, разработчик драйвера с помощью задания структуры удаленного адреса и алгоритма функции **zCompare_xxx** может запрограммировать критерии формирования блоковых запросов.

Если анализируемый канал попадает в блок, ему устанавливается соответствующий признак (этот признак не доступен пользователю), **zCompare_xxx** возвращает ненулевое значение.

3. Формирование и посылка сообщения

После формирования блоков МРВ находит канал-инициатор блокового запроса и вызывает функцию **Set_xxx** с необходимыми параметрами:

- **ia** – удаленный адрес канала-инициатора запроса;
- **q_rec** – число передаваемых или принимаемых значений (опрашиваемых регистров);
- адрес буфера **sbuf**. Память под буфер выделяет МРВ.

Функция **Set_xxx** формирует сообщение-строку в требуемом для используемого протокола формате, и помещает это сообщение в буфер **sbuf**.

Если значение параметра **max_send**, переданное функцией **Set_xxx**, больше 0, то МРВ посылает содержимое буфера **sbuf** в порт. Длина посылаемого сообщения в байтах равна значению **max_send**.

Если при передаче был сбой, то такт обмена завершается. При этом увеличивается счетчик ошибок, но для канала ничего не меняется.

4. Прием ответа

*5. Вызов функции **zReadAny_xxx***

Если МРВ определяет, что драйвер поддерживает функцию **zReadAny_xxx** (эта функция может отсутствовать в драйвере), она будет вызываться до тех пор, пока не вернет 0 (это означает, что все байты от устройства приняты, и далее будет вызвана функция **Check_xxx**). Если **zReadAny_xxx** возвращает не 0, то МРВ примет еще **count_rec** байтов от устройства и опять вызовет **zReadAny_xxx**. Таким способом можно получить от устройства сообщение, длина которого заранее (т.е., на момент вызова **Set_xxx**) не известна.

6. Проверка корректности полученного ответа

На этом этапе вызывается функция **Check_xxx**. Если возвращаемое ей значение равно 0, то принятые данные считаются корректными и вызывается функция **Get_xxx** для расшифровки принятого сообщения.

При отличии от 0 значения, возвращаемого функцией **Check_xxx**, анализируется величина параметра **max_rec**. Если он больше 0, то осуществляется прием дополнительных байтов. Количество этих байтов равно значению параметра **max_rec**.

Если возвращаемое значение параметра **max_rec** равно 0, то каналу, для которого осуществлялся вызов драйвера, устанавливается признак аппаратной недостоверности. Если при этом значение параметра **max_send** больше 0, то осуществляется посылка по последовательному порту строки из буфера **rbuf**. Количество посылаемых байтов равно величине параметра **max_send**.

7. Расшифровка полученного ответа

На этом этапе осуществляется вызов функции **Get_xxx**.

Для каналов OUTPUT характеристика завершения передачи данных описывается в элементе **p[0].F.fmt[1]** массива структур **RSDATA**:

- 0 – нормальное завершение передачи данных;
- 1 – сбой при передаче. Выставить каналу флаг аппаратной недостоверности;
- 2 – сбой при передаче. Выставить каналу флаг аппаратной недостоверности и флаг повторной передачи.

Для каналов INPUT характеристика завершения приема данных передается в MPB с помощью параметра **type_cnv** функции **Get_xxx**. Его значения передают в MPB следующую информацию:

- 99 – получены некорректные данные. Выставить каналу флаг аппаратной недостоверности;
- 0 или 1 – получены корректные данные. Использовать алгоритм обработки запросов **DATA11** (см. ниже);
- 2 или 3 – получены корректные данные. Использовать алгоритм обработки запросов **BLOCKDATA11** (см. ниже);
- 8,10 – получены корректные значения для каналов блочного запроса (8 – запись полученных значений в архив запрещена, 10 – разрешена). Использовать алгоритм обработки данных **DATA11**. Элементы **p[i]** (где $i=2k$, $k=0.1,2...$) содержат полученные значения, элементы **p[i+1]** – соответствующие этим значениям метки времени:

p[i+1].V.d – число секунд с 1 января 1970 года

p[i+1].F.ind[0] – число миллисекунд

- 12 – получен блок корректных данных для одного канала для записи в архив. Использовать алгоритм обработки данных **DATA11**. Элементы **p[i]** (где $i=2k$, $k=0.1,2...$) содержат полученные значения, элементы **p[i+1]** – соответствующие этим значениям метки времени:

p[i+1].V.d – число секунд с 1 января 1970 года

p[i+1].F.ind[0] – число миллисекунд

- 9,11 – то же, что 8, 10, но использовать алгоритм обработки данных **BLOCKDATA11**. 9 – запись полученного значения канала в архив запрещена, 11 – разрешена.

- 13 – то же, что 12, но использовать алгоритм обработки данных **BLOCKDATA11**.

Если в массиве **p** содержатся метки времени (**type_cnv** больше 3) значение **q_rec** необходимо увеличить в два раза, так как для передачи каждого значения канала и метки времени для архива нужно два элемента мас-

сива структур **RSDATA** ($q_rec = 2 * q_rec$).

Если получено значение **type_cnv**, равное 12 или 13, МРВ просматривает все каналы базы, совпадающие с каналом, инициировавшим запрос, по типу, подтипу и дополнению к подтипу, и вызывает драйвер с функцией **zCompare_xxx** с **count** = -1. На первом же канале, по которому драйвер ответил положительно, из буфера **RSDATA** считываются записи в количестве **q_rec** и записываются в архив с индексом этого канала. Объем настроек, используемых при этом функцией **zCompare_xxx** (включая номер RS и адрес устройства) определяется драйвером. В канале, инициировавшем запрос, записывается значение **q_rec**.

Если младший бит первого байта параметра **type_cnv** равен 1 (0x0100), то следующий вызов драйвера выполняется для того же самого канала (переход к обработке очередного канала не производится). Это свойство используется при разработке многопроходных драйверов.

8. Перейти к пункту 2

Алгоритм обработки данных **BLOCKDATA11**

Этот алгоритм используется, если количество и характер данных, передаваемых контроллером в ответе, заранее не известны. Примером является Ш711/1, текст драйвера для которого приведен ниже.

При разборе ответа от контроллера по данному алгоритму МРВ выполняет следующие операции.

1. Проход по базе с ее начала и выбор каналов с тем же подтипом, дополнением к подтипу и двумя младшими байтами удаленного адреса (**ia.i[0] == ia1.i[0]**). Количество вызываемых каналов определяется значением параметра **q_rec**, возвращаемым **Get_xxx**.
2. Для каждого из найденных каналов вызывается функция **zCompare_xxx**. В ней надо проанализировать принадлежность выбранного канала к разбираемому блоковому запросу. Если данные для анализируемого канала в запросе не содержатся, то функция должна вернуть значение 0. В противном случае возвращаемая величина должна быть на 1 больше индекса элемента массива структур **RSDATA**, где размещено значение для записи в канал.
3. По каждому из каналов, значения которых формируются данным запросом, анализируется элемент **p[ind].F.fmt[1]** массива **p** структур **RSDATA**, где **ind** – уменьшенное на 1 значение, возвращенное по данному каналу функцией **zCompare_xxx**. Значение этого элемента определяет следующие действия:

- 0 – присвоить входу канала значение элемента **p[ind].V.v** массива структур **RSDATA**;

1 – сформировать для канала признак аппаратной недостоверности;

Выход из алгоритма осуществляется либо после прохода по всем каналам, либо при равенстве количества выбранных каналов значению **q_rec**.

Алгоритм обработки данных *DATA11*

Данный алгоритм следует использовать при единичных запросах или при блоковых, если заранее известен порядок размещения значений, которые будут переданы устройством в ответе.

При разборе ответа от контроллера по данному алгоритму МРВ выполняет следующие операции:

1. Осуществляет проход по базе с ее начала. (Повторный проход выполняется с места окончания предыдущего блокового запроса). При этом выбираются каналы с тем же подтипом, дополнением к подтипу и двумя младшими байтами удаленного адреса (**ia.i[0] == ia1.i[0]**). Количество вызываемых каналов определяется значением параметра **q_rec**.

2. Если найден канал, удовлетворяющий описанным условиям, то для него вызывается функция **zCompare_xxx**. В эту функцию передаются удаленные адреса канала-инициатора запроса и найденного канала. В ней надо проанализировать принадлежность выбранного канала к разбираемому блоковому запросу.

Если данные для анализируемого канала в запросе не содержатся, то функция должна вернуть значение 0. В противном случае возвращаемая величина должна быть больше 0, а значение параметра **count** будет равно индексу элемента в массиве структур **RSDATA**, содержащего значение для записи в канал.

3. Для найденного канала анализируется элемент **p[ind].F.fmt[1]** массива структур **RSDATA**. Значение этого элемента определяет следующие действия:

0 – присвоить входу канала значение элемента **p[ind].V.v** массива структур **RSDATA**;

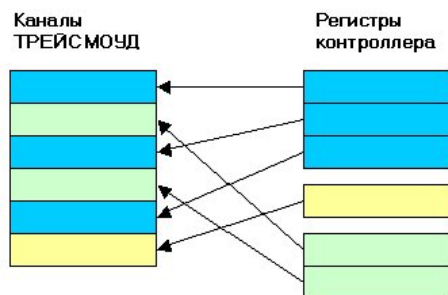
1 – сформировать для канала признак аппаратной недостоверности.

Выход из алгоритма осуществляется либо после прохода по всем каналам, либо при равенстве количества выбранных каналов значению **q_rec**.

Алгоритм формирования блоковых запросов

Блоковые запросы позволяют МРВ за один сеанс обмена данными с контроллером получить значения для нескольких каналов. Такие запросы применяются только для чтения данных из контроллера, т.е. для каналов INPUT. Рассмотрим ситуацию, когда в контроллере есть несколько регистров (ячеек памяти), которым мы хотим сопоставить столько же каналов

TRACE MODE. Пусть контроллер позволяет одной командой считать значения всех интересующих нас регистров. Обычно такие команды есть для регистров, расположенных по последовательным адресам в памяти контроллера. В команде тогда задается адрес первого регистра и количество регистров. Блоковый запрос позволяет драйверу использовать такие команды. В драйверах t12 всегда используются блочные запросы.



Для реализации блочных запросов МРВ должен до начала пересчета базы каналов «знать», какие каналы образуют группы. Таких групп может быть произвольное число. Группа также может состоять и из единственного канала. На представленном рисунке можно видеть три группы каналов и соответствующие блоки регистров, обозначенные разными цветами. Все каналы одной группы, естественно, имеют одинаковый тип (INPUT), подтип и дополнение к подтипу. Кроме того, у них совпадают два младших байта удаленного адреса. Об этом надо помнить при разработке структуры удаленного адреса (настроек канала).

При пересчете базы каналов функции **Set**, **Check** и **Get** будут вызываться по одному разу для каждой группы каналов. В функции **Set** известно число регистров, которые нужно считать из контроллера – оно равно **q_rec** (см. объявление функции). Поэтому **Set** может сформировать команду для контроллера на чтение этого числа регистров. Удаленный адрес (**ia**), передаваемый в эту функцию – это удаленный адрес первого канала из группы.

Функции **Check** нет необходимости знать число запрашиваемых значений. Она должна просто проверить правильность формата пришедшего ответа (например, она может посчитать контрольную сумму).

От функции **Get** требуется расшифровать ответ контроллера и поместить полученные значения в массив структур **RSDATA**. Каждому значению соответствует один элемент массива. В функцию передается указатель на этот массив. Также функция должна через аргумент **q_rec** (передаваемый по ссылке) сообщить МРВ количество значений, полученных в ответе. Кроме того, функция должна сообщить через аргумент **type_cnv** (также передаваемый по ссылке), какой алгоритм применять для разбора значений, переданных в массиве структур **RSDATA**. Существует два алгоритма – **DATA** и **BLOCKDATA**.

Функция **zCompare** вызывается для каждого канала в группе после функции **Get**. Точнее, она вызывается для всех каналов, у которых совпадают тип, подтип, дополнение к подтипу и два младших байта удаленного адреса. Если канал не принадлежит данному запросу, функция должна вернуть 0. Если канал принадлежит данному запросу, то возвращаемое значение зависит от того, какой алгоритм используется – **DATA** или **BLOCKDATA**. В случае использования алгоритма **DATA** достаточно вернуть не ноль. В случае алгоритма **BLOCKDATA** нужно вернуть число, на 1 большее, чем индекс элемента массива структур **RSDATA**, в котором находится значение канала (помещенное туда функцией **Get**).

Ограничения драйвера t11

Некоторые протоколы обмена данными с контроллером предусматривают переключение скорости обмена в течение одного сеанса обмена. Драйвер t11 не позволяет реализовать такие протоколы, т.к. МРВ в этом случае сам управляет чтением и записью в последовательный порт, и не предусмотрен способ изменения из драйвера каких бы то ни было настроек порта.

TCOM5. Драйвер t12

TCOM5. Функции модуля описания протокола

MPB вызывает из модуля **t12s<N>.dll** шесть функций.

TCOM5 Open_yyy. Функция инициализации

Эта функция вызывается при инициализации работы MPB с драйвером. В ней можно прописать операции для установки начальных условий для обмена данными. В отличие от функции **Prepare_xxx** драйвера t11, функция **Open_yyy** всегда формирует блоковые запросы.

```
int Open_yyy(int &media, char *str, int &q_in, int &q_out, int &q_data);
```

где

media – номер драйвера;

str – строка, содержащая название протокола (не более 31 байта). Передается из драйвера MPB. Профайлер пишет эту строку в свой протокол;

q_in – размер буфера **rbuf** для приема данных;

q_out – размер буфера **sbuf** для передачи данных;

q_data – максимальное количество структур **RSDATA** (т.е., максимальное количество каналов в групповом запросе; указатель **p** на массив структур **RSDATA**, передаваемый в функции **Set_yyy** и **Get_yyy**, будет указывать на массив из **q_data** элементов.).

Возвращаемое функцией значение, равное 0, свидетельствует о нормальном ее выполнении. Отличие возвращаемого значения от 0 воспринимается как ошибочное завершение. В этом случае каналы с соответствующим дополнением к подтипу отключаются.

TCOM5 zCompare_yyy. Функция сравнения

Эта функция используется при формировании и расшифровке блоковых запросов.

```
int zCompare_yyy(IA &ia0, IA &ia1, int &count);
```

где

ia0 – удаленный адрес первого канала;

ia1 – удаленный адрес сравниваемого канала;

count – число каналов, уже находящихся в блоке, минус один при формировании блоковых запросов или смещение канала относительно начала блока при расшифровке запроса. Этот параметр формируется МРВ и передается в драйвер. По нему можно внести ограничение на добавление новых элементов в блоковый запрос. Для алгоритма **DATA12** этот параметр равен индексу элемента в структуре **RSDATA**, содержащего значение для записи в канал.

Если возвращаемое данной функцией значение больше 0, то канал принадлежит к данному запросу; если при этом осуществляется обработка группового запроса по алгоритму **BLOCKDATA12**, возвращаемое значение на 1 больше индекса размещения данных в структуре **RSDATA**,

Если возвращаемое данной функцией значение равно 0, то анализируемый канал не попадает в данный запрос.

TCOM5 Set_yyy. Функция формирования посылаемого сообщения

Эта функция вызывается МРВ при пересчете значений канала, связанного с драйвером.

```
int Set_yyy(IA &ia, int &max_send, int &max_rec,
RSDATA *p, char *sbuf);
```

где

ia – удаленный адрес канала (передается в драйвер);

max_send – количество байтов для отправки (формируется в драйвере);

max_rec – количество байтов в ответе (формируется в драйвере);

p – указатель на массив структур **RSDATA**, содержащий данные и их форматы. Этот параметр требуется, если канал имеет тип OUTPUT, – в этом случае элемент 0 массива содержит значение для записи в контроллер.

Если канал типа OUTPUT, то

p[0].F.fmt[3]=1;

для данных в формате FLOAT:

p[0].V.v – посылаемое значение,

p[0].F.fmt[0]=0;

для данных в формате HEX:

p[0].V.i[0] – посылаемое значение,

p[0].F.fmt[0]=0x40.

Тип значения (HEX или FLOAT), посылаемого из канала в устройство, не связан с типом канала, который пользователь указал в редакторе базы каналов. **p[0].F.fmt[0]** нужен только для корректной интерпретации **p[0].V** внутри драйвера.

Если канал типа **INPUT**, то

p[0].F.fmt[3]=0.

sbuf – буфер для отправки (формируется в драйвере).

Данная функция вызывается для формирования сообщений, посылаемых с использованием соответствующего носителя в устройство. Сформированное сообщение должно быть размещено в буфере **sbuf**.

Отличное от 0 значение, возвращаемое функцией, свидетельствует об ошибке ее выполнения.

С помощью функции **Set_yyy** можно установить значение каналу, обратившемуся к драйверу. Если эта функция возвращает **max_send=max_rec**=[младший байт буфера **sbuf**]=0, то МРВ присваивает каналу значение, взятое из первого элемента массива структур **RSDATA**.

TCOM5 Check_yyy. Функция проверки

Эта функция вызывается после приема ответа от контроллера перед функцией расшифровки полученных данных. Ее назначение - проверить корректность полученного ответа. Она имеет следующий формат:

```
int Check_yyy(IA &ia, int &count_rec, int
&max_rec, char *rbuf);
```

где

ia – удаленный адрес канала (передается в драйвер);

count_rec – число принятых байтов (передается в драйвер);

max_rec – количество байтов в ответе (передается драйверу и формируется в нем);

rbuf – буфер, содержащий принятое сообщение (передается драйверу и формируется в нем).

Если возвращаемое функцией значение равно 0, то принятые данные считаются корректными и осуществляется вызов функции расшифровки принятого сообщения (**Get_yyy**).

TCOM5 Get_yyy. Функция расшифровки принятого сообщения

Эта функция вызывается для расшифровки полученного ответа от устройства. Она имеет следующий формат:

```
int Get_yyy(IA &ia, int &count_rec, int &q_rec,
RSDATA *p, char *rbuf, int &type_cnv);
```

где

ia – удаленный адрес канала (передается в драйвер);

count_rec – число принятых символов (передается в драйвер);

q_rec – число посылаемых или запрашиваемых значений (передается в драйвер, может быть изменено);

p – указатель на массив структур **RSDATA**, содержащий данные и их форматы;

rbuf – буфер, содержащий принятое сообщение (передается в драйвер);

type_cnv – тип разборки принятых данных, формируется в драйвере (см. ниже):

Возвращаемое данной функцией значение, большее 0, означает ошибку в данных. При этом соответствующему каналу устанавливается признак аппаратной недостоверности.

TCOM5 Close_yyy. Функция завершения обмена

Эта функция не имеет параметров. Она вызывается при завершении работы MPB и имеет следующий формат:

```
int Close_yyy();
```

TCOM5. Функции модуля описания интерфейса

MPB вызывает из модуля **media<n>.dll** шесть функций.

TCOM5 OpenMedia. Функция инициализации

Эта функция вызывается при инициализации работы MPB с драйвером. В ней можно прописать операции для установки начальных условий для обмена данными.

```
int OpenMedia(long &hndl);
```


где

hndl – любое отличное от 0 значение – признак интерфейса. Это значение передается в МРВ и затем всегда используется для передачи драйверу в качестве параметра при обращении к данному интерфейсу

TCOM5 CloseMedia. Функция завершения работы

Данная функция вызывается при завершении работы МРВ после закрытия протоколов (вызовы функций **Close_yyy** по всем протоколам).

```
int CloseMedia(long &hndl);
```

где

hndl – признак интерфейса. Это значение передается в МРВ функцией **OpenMedia** при открытии обмена по данному интерфейсу.

TCOM5 StartMedia. Функция запуска интерфейса

Эта функция вызывается перед началом работы МРВ в реальном времени после вызова функции **OpenMedia**.

```
int StartMedia(long &hndl);
```

где

hndl – признак интерфейса. Это значение передается в МРВ функцией **OpenMedia** при открытии обмена по данному интерфейсу.

TCOM5 StopMedia. Функция остановки интерфейса

Эта функция вызывается перед завершением работы МРВ в реальном времени (до закрытия протоколов). Она имеет следующий формат:

```
int StopMedia(long &hndl);
```

где

hndl – признак интерфейса. Это значение передается в МРВ функцией **OpenMedia** при открытии обмена по данному интерфейсу.

TCOM5 ReadMedia. Функция чтения ответа

Эта функция вызывается для запроса данных у устройств. Она имеет следующий формат:

```
int ReadMedia(long &hdl, char *rbuf, int in_count,
int &actual_count);
```

где

hdl – признак интерфейса. Это значение передается в MPV функцией **OpenMedia** при открытии обмена по данному интерфейсу;

rbuf – буфер для размещения ответа от внешнего устройства (формируется в драйвере);

in_count – количество символов, которые должны были присутствовать в ответе (передается в драйвер);

actual_count – реальное количество принятых символов (формируется в драйвере).

TCOM5 WriteMedia. Функция отправки запроса/данных

Эта функция вызывается для передачи данных в устройство. Она имеет следующий формат:

```
int WriteMedia(long &hdl, char *sbuf, int
out_count, int &actual_count);
```

где

hdl – признак интерфейса. Это значение передается в MPV функцией **OpenMedia** при открытии обмена по данному интерфейсу;

sbuf – буфер, содержащий сообщение для отправки в контроллер (передается в драйвер);

out_count – количество символов, которые должны быть переданы в сообщении (передается в драйвер);

actual_count – реальное количество переданных символов (формируется в драйвере).

TCOM5. Пример драйвера t12

```
int count=0;
void rrr(char *);
void rrr(char *str)
{
    FILE *fx;
    fx=fopen("e:\\bbb", "a");
    fprintf(fx, "%s %d\n", str, count);
    fclose(fx);
}
int OpenMedia(long &hdl)
{
```

```
        hndl=1;
        rrr("OpenMedia");
        return(0);
    }
int CloseMedia(long &hndl)
{
    rrr("CloseMedia");
    return(0);
}
int StartMedia(long &hndl)
{
    rrr("StartMedia");
    return(0);
}
int StopMedia(long &hndl)
{
    rrr("StopMedia");
    return(0);
}
int WriteMedia(long &hndl, char *sbuf, int out_count, int &actual_count)
{
    rrr("WRITE");
    rrr(sbuf);
    return(0);
}
int ReadMedia(long &hndl, char *rbuf, int in_count, int &actual_count)
{
    rrr("READ");
    count++;
    return(0);
}
.....
int Set_yyy(IA &ia,int &max_send,int &max_rec,RSDATA *p,char *sbuf)
{
    sprintf(sbuf,"%0.2d %0.8d\n", (int) ia.c[2],time(NULL) );
    max_send=strlen(sbuf);
    max_rec=10;
    return(0);
}
int Get_yyy(IA &ia,int &count_rec,int &q_rec,RSDATA *p,char *rbuf,int
&type_cnv)
{
    q_rec=1;
    type_cnv=0;
    p[0].V.v=ia.c[2];
    p[0].F.fmt[0]=p[0].F.fmt[1]=0;
    return(0);
}
int Check_yyy(IA &ia,int &count_rec,int &max_rec,char *rbuf)
{
    return(0);
}
int zCompare_yyy(IA &ia0,IA &ia1,int &count)
{
    if (ia0.c[2] == ia1.c[2])
        return(1);
    else return(0);
}
int Open_yyy(int &media,char *str,int &q_in,int &q_out,int &q_data)
{

```

```
        q_data=128;
        media=0;
        q_in=1024;
        q_out=1024;
        return(0);
    }
int Close_yyy()
{
    return(0);
}
```

TCOM6. Драйвер t12

TCOM6. Функции модуля описания интерфейса

В данном разделе описаны отличия функций **TCOM6** от одноименных функций **TCOM5**. Общие параметры функций описаны в разделе **TCOM5. Функции модуля описания интерфейса**.

TCOM6. Функции инициализации, запуска интерфейса, остановки интерфейса и завершения работы

Эти функции идентичны одноименным функциям **TCOM5**:

```
int OpenMedia(long &hndl);
int StartMedia(long &hndl);
int StopMedia(long &hndl);
int CloseMedia(long &hndl);
```

TCOM6 WriteMedia. Функция отправки запроса/данных

```
int WriteMedia(long &hndl, char *sbuf, int
out_count, int &actual_count, IA &ia, void
*ext_data, DEF_MAINFLAG flags);
```

Параметры:

- **hndl** – признак интерфейса;
- **sbuf** – буфер;
- **out_count** – количество символов, которые должны быть переданы в сообщении;
- **actual_count** – реальное количество переданных символов;
- **ia** – удаленный адрес канала;
- **ext_data** – дополнительная информация для канала;
- **flags** – битовая структура, набор значений основных атрибутов канала.

Параметр **ext_data** является указателем на строку, заданную пользователем в поле **Дополнительно**, т.е.

```
const char* szExtString = *(char**)ext_data;
```

Значения полей битовой структуры **DEF_MAINFLAG** описаны в заголовочном файле (см. **TCOM6. Заголовок драйвера t12**).

Остальные параметры имеют такое же назначение, как и в интерфейсе

TCOM5.***TCOM6 ReadMedia. Функция чтения ответа***

```
int ReadMedia(long &hdl, char *rbuf, int in_count,
int &actual_count, IA &ia, void *ext_data,
DEF_MAINFLAG flags);
```

Параметры:

- **hdl** – признак интерфейса;
- **sbuf** – буфер;
- **in_count** – количество символов, которые должны были присутствовать в ответе;
- **actual_count** – реальное количество принятых символов;
- **ia** – удаленный адрес канала;
- **ext_data** – дополнительная информация для канала;
- **flags** – битовая структура, набор значений основных атрибутов канала.

Параметры **ext_data** и **flags** аналогичны одноименным параметрам функции **WriteMedia()** **TCOM6**. Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6. Функции модуля описания протокола

В данном разделе описаны отличия функций **TCOM6** от одноименных функций **TCOM5**. Общие параметры функций описаны в разделе **TCOM5. Функции модуля описания протокола**.

TCOM6 Open_yyy. Функция инициализации

```
int Open_yyy(int &media, char *str, int &q_in, int
&q_out, int &q_data);
```

Данная функция идентична функции **Open_yyy** **TCOM5**.

TCOM6 zCompare_yyy. Функция сравнения

```
int zCompare_yyy(IA &ia0, IA &ia1, int &count, void
*ext_data0, void *ext_data1, DEF_MAINFLAG flags0,
DEF_MAINFLAG flags1);
```

Параметры:

- **ia0** – удаленный адрес первого канала;

- **ia1** – удаленный адрес сравниваемого канала;
- **count** – число каналов в блоке;
- **ext_data** – дополнительная информация для первого канала;
- **ext_data1** – дополнительная информация для сравниваемого канала;
- **flags0** – битовая структура, набор значений основных атрибутов первого канала;
- **flags1** – битовая структура, набор значений основных атрибутов сравниваемого канала.

Параметр **ext_data** является указателем на строку, заданную пользователем в поле **Дополнительно**, т.е.

```
const char* szExtString0 = *(char**)ext_data0;
```

Значения полей битовой структуры **DEF_MAINFLAG** описаны в заголовочном файле (см. **TCOM6. Заголовок драйвера t12**). Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 Set_yyy. Функция формирования посылаемого сообщения

```
int Set_yyy(IA &ia, int &max_send, int &max_rec,
RSDATA *p, char *sbuf, void *ext_data, DEF_MAINFLAG
flags, int &q_rec);
```

Параметры:

- **ia** – удаленный адрес канала;
- **max_send** – количество байтов для отправки;
- **max_rec** – количество байтов в ответе;
- **p** – указатель на массив структур RSDATA, содержащий данные и их форматы;
- **sbuf** – буфер для отправки;
- **ext_data** – дополнительная информация для канала;
- **flags** – битовая структура, набор значений основных атрибутов канала;
- **q_rec** – число посылаемых или запрашиваемых значений (передается в драйвер).

Если канал имеет тип OUTPUT, элемент 0 массива **p** содержит значение для записи в контроллер. Формат значения определяется флагом **p[0].F.fmt[0]**:

- **p[0].F.fmt[0]=0** для данных в формате FLOAT; тогда **p[0].V.v** содержит посылаемое значение,
- **p[0].F.fmt[0]=0x40** для 16-разрядных целочисленных данных,

p[0].V.i[0] – посылаемое значение,

- **p[0].F.fmt[0]=0x80** для 32-разрядных целочисленных данных, **p[0].V.d** – посылаемое значение.

Параметр **ext_data** является указателем на строку, заданную пользователем в поле **Дополнительно**, т.е.

```
const char* szExtString = *(char**)ext_data;
```

Значения полей битовой структуры **DEF_MAINFLAG** описаны в заголовочном файле (см. **TCOM6. Заголовок драйвера t12**). Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 Check_yyy. Функция проверки

```
int Check_yyy(IA &ia, int &count_rec, int &max_rec,
char *rbuf, void *ext_data, DEF_MAINFLAG flags, int
q_rec);
```

Параметры:

- **ia** – удаленный адрес канала;
- **count_rec** – число принятых байтов;
- **max_rec** – количество байтов в ответе;
- **rbuf** – буфер, содержащий принятое сообщение;
- **ext_data** – дополнительная информация для канала;
- **flags** – битовая структура, набор значений основных атрибутов канала;
- **q_rec** – число посылаемых или запрашиваемых значений.

Параметры **ext_data**, **flags** и **q_rec** аналогичны одноименным параметрам функции **Set_yyy()** **TCOM6**. Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 Get_yyy. Функция расшифровки принятого сообщения

```
int Get_yyy(IA &ia, int &count_rec, int &q_rec,
RSDATA *p, char *rbuf, int &type_cnv, void
*ext_data, DEF_MAINFLAG flags);
```

Параметры:

- **ia** – удаленный адрес канала;
- **count_rec** – число принятых символов;
- **q_rec** – число посылаемых или запрашиваемых значений;
- **p** – указатель на массив структур **RSDATA**, содержащий данные и

их форматы;

- **rbuf** – буфер, содержащий принятое сообщение;
- **type_cnv** – тип разборки принятых данных;
- **ext_data** – дополнительная информация для канала;
- **flags** – битовая структура, набор значений основных атрибутов канала.

Параметры **ext_data** и **flags** аналогичны одноименным параметрам функции **Set_yyy() TCOM6**. Остальные параметры имеют такое же назначение, как и в интерфейсе **TCOM5**.

TCOM6 Close_yyy. Функция завершения обмена

Данная функция идентична функции **Close_yyy TCOM5**:

```
int Close_yyy();
```

TCOM6. Функция-признак интерфейса TCOM6

```
void zzTM6Stub();
```

Функция не имеет параметров, служит только для указания МРВ использовать интерфейс **TCOM6** и никогда не вызывается.

TCOM6. Заголовок драйвера t12

```
typedef union
{
    unsigned char c[6];
    unsigned short int i[3];
} IA;
typedef union
{
    unsigned char c[4];
    unsigned short int i[2];
    unsigned long l;
    float f;
} FOUR_BYTE;
typedef union
{
    unsigned char c[2];
    unsigned short int i;
} TWO_BYTE;
typedef struct
{
    union
    {
```

```

        float v;
        unsigned char c[4];
        unsigned short int i[2];
        unsigned long l;
        int d;
    } V;
union
{
    unsigned char fmt[4];
    unsigned short int ind[2];
} F;
} RSDATA;
typedef union
{
    unsigned long lflag;
    unsigned short int flag[2]; // 0 bits 1-
struct
{
    unsigned SC : 1; // ON/OFF
    unsigned WC : 1; // connect/disconnect
    unsigned FA : 1; // hardware failure
    unsigned IO : 1; // INPUT/OUTPUT
    unsigned HF : 1; // FLOAT/HEX
    unsigned interval : 3; // interval
    unsigned FS : 1; // program invalidity
    unsigned bIO : 1; // use in net but in work re-changes
                        // type io
    unsigned bSV : 1; //use in net but set in request and
                        // reset in response
    unsigned b11 : 1; // if set in hex print in int
    unsigned LENB0 : 1; // init ai and di
    unsigned LENB1 : 1; // 0 - aperture 1 - smooth
    unsigned LENB2 : 1;
    unsigned LENB3 : 1; // port.data - char * presents

    unsigned frqc3 : 7; //frq.c[3],q_rec
    unsigned d_bit : 1; // for debugging
    unsigned frqc2 : 6; //frq.c[2]
    unsigned b14 : 1; // value or FA changes
    unsigned b15 : 1; // for show

}
    BIT;
}
DEF_MAINFLAG;
#define ERR_RT_FILE      1 // ошибка открытия файла
#define ERR_RT_SEEK     2 // ошибка поиска файла
#define ERR_RT_WRITE    3 // ошибка записи файла
#define ERR_RT_READ     4 // ошибка чтения файла

```

```
#define ERR_RT_MEM      5 // недостаточно памяти
#define ERR_RT_LIST    6 // ошибка создания списка
#define ERR_RT_FORMAT  7 // ошибка в формате
#define ERR_RT_COUNT   8 // неправильный счетчик
#define ERR_RT_TIMEOUT 9 // обнаружен таймаут
#define ERR_RT_RESP    10 // неправильный ответ
#define ERR_RT_FUNC    11 // код ошибки, возвращаемый API WIN32
#define ERR_RT_NOTFOUND 12 // запрос неопределенного канала
#define ERR_RT_CODE    13 // код ошибки, посланный контроллером
#define ERR_RT_FSC     14 // неправильная контрольная сумма

// Функции библиотеки описания интерфейса (media)

int __declspec(dllexport) OpenMedia(long &hdl);
int __declspec(dllexport) StartMedia(long &hdl);
int __declspec(dllexport) StopMedia(long &hdl);
int __declspec(dllexport) CloseMedia(long &hdl);

int __declspec(dllexport) WriteMedia(long &hdl, char *sbuf, int
out_count, int &actual_count, IA &ia, void *ext_data, DEF_MAINFLAG
flags);

int __declspec(dllexport) ReadMedia(long &hdl, char *rbuf, int
in_count, int &actual_count, IA &ia, void *ext_data, DEF_MAINFLAG
flags);

// Функции библиотеки описания протокола

int __declspec(dllexport) Open_yyy(int &media, char *str, int
&nRecvBufferSize, int &nSendBufferSize, int &nMaxBlockQuery);

int __declspec(dllexport) Close_yyy();

int __declspec(dllexport) Set_yyy(IA &ia, int &max_send, int &max_rec,
RSDATA *p, char *sbuf, void *ext_data, DEF_MAINFLAG flags, int &q_rec);

int __declspec(dllexport) Check_yyy(IA &ia, int &count_rec, int
&max_rec, char *rbuf, void *ext_data, DEF_MAINFLAG flags, int q_rec);

int __declspec(dllexport) Get_yyy(IA &ia, int &count_rec, int &q_rec,
RSDATA *p, char *rbuf, int &type_cnv, void *ext_data, DEF_MAINFLAG
flags);

int __declspec(dllexport) zCompare_yyy(IA &ia0, IA &ia1, int &count,
void *ext_data0, void *ext_data1, DEF_MAINFLAG flags0, DEF_MAINFLAG
flags1);

void __declspec(dllexport) zzTM6Stub();
```

Алгоритм взаимодействия с драйвером t12

Инициализация

На этом этапе МРВ инициализирует протоколы исходя из наличия каналов с соответствующими дополнениями к подтипу. Затем для этих протоколов вызываются функции **Open_yyy**. Далее инициализируются все интерфейсы, которые используются включенными протоколами. Для этого вызываются функции **OpenMedia** для каждого из интерфейсов.

Построение блоковых запросов

По каждому из используемых протоколов строятся блоковые запросы. Алгоритм построения этих запросов приведен выше в разделе, посвященном описанию разработки драйверов t11.

Работа в реальном времени

1. Поиск канала, для которого требуется обмен

2. Вызов функции формирования строки послылки. Set_yyy

Если значение параметра **max_send**, переданное функцией **Set_yyy**, больше 0, МРВ осуществляет вызов функции **WriteMedia** для послылки по соответствующему интерфейсу содержимого буфера **sbuf**. Длина послылаемой строки в байтах равна значению **max_send**.

Если функции **Set_yyy** или **WriteMedia** возвращают значения, отличные от 0, для канала устанавливается признак аппаратной недостоверности.

3. Прием строки ответа. ReadMedia

На этом этапе вызывается функция чтения **ReadMedia** для соответствующего интерфейса. Если эта функция возвращает значение, отличное от 0, каналу устанавливается признак аппаратной недостоверности.

4. Проверка корректности полученного ответа

На этом этапе вызывается функция **Check_yyy**.

Если возвращаемое функцией значение равно 0, то принятые данные считаются корректными и осуществляется вызов функции **Get_yyy** для расшифровки принятой строки. Если возвращаемое значение больше 0, каналу устанавливается признак аппаратной недостоверности.

5. Расшифровка полученного ответа

На этом этапе осуществляется вызов функции **Get_yyy**.

Характеристика завершения передачи или приема данных описывается в элементе **p[0].F.fmt[1]** массива структур **RSDATA**:

- 0 – нормальное завершение передачи данных;
- 1 – сбой при передаче. Выставить каналу флаг аппаратной недостоверности;
- 2 – сбой при передаче. Выставить каналу флаг аппаратной недостоверности и флаг повторной передачи.

Элементу **p[0].F.fmt[0]** необходимо присвоить 0, иначе МРВ не обновит значение канала.

Для каналов INPUT при помощи аргумента **type_cnv** функции **Get_yyy** в МРВ передается следующая информация:

- 0 – использовать алгоритм обработки запросов **DATA12**;
- 1 – использовать алгоритм обработки запросов **BLOCKDATA12**;
- 8,10 – получены корректные значения для каналов блочного запроса (8 – запись полученных значений в архив запрещена, 10 – разрешена). Использовать алгоритм обработки данных **DATA12**. Элементы **p[i]** (где $i=2k$, $k=0.1,2...$) содержат полученные значения, элементы **p[i+1]** – соответствующие этим значениям метки времени:

p[i+1].V.d – число секунд с 1 января 1970 года

p[i+1].F.ind[0] – число миллисекунд

- 12 – получен блок корректных данных для одного канала для записи в архив. Использовать алгоритм обработки данных **DATA12**. Элементы **p[i]** (где $i=2k$, $k=0.1,2...$) содержат полученные значения, элементы **p[i+1]** – соответствующие этим значениям метки времени:

p[i+1].V.d – число секунд с 1 января 1970 года

p[i+1].F.ind[0] – число миллисекунд

- 9,11 – то же, что 8, 10, но использовать алгоритм обработки данных **BLOCKDATA12**. 9 – запись полученного значения канала в архив запрещена, 11 – разрешена.

- 13 – то же, что 12, но использовать алгоритм обработки данных **BLOCKDATA12**.

Если в массиве **p** содержатся метки времени (**type_cnv** больше 1) значение **q_rec** необходимо увеличить в два раза, так как для передачи каждого значения канала и метки времени для архива нужно два элемента мас-

сива структур RSDATA ($q_rec = 2 * q_rec$).

Если получено значение **type_cnv**, равное 12 или 13, MPB просматривает все каналы базы, совпадающие с каналом, инициировавшим запрос, по типу, подтипу и дополнению к подтипу, и вызывает драйвер с функцией **zCompare_yyy** с **count** = -1. На первом же канале, по которому драйвер ответил положительно, из буфера **RSDATA** считываются записи в количестве **q_rec** (выдано функцией **Get_yyy**) и записываются в архив с индексом этого канала. Объем настроек, используемых при этом функцией **zCompare_yyy** (включая номер **RS** и адрес устройства) определяется драйвером. В канале, инициировавшем запрос, записывается значение **q_rec**.

Алгоритмы обработки данных **DATA12** и **BLOCKDATA12** совпадают с аналогичными алгоритмами для обработки данных по последовательным интерфейсам (**DATA11** и **BLOCKDATA11**), описание которых приведено выше в разделе, посвященном разработке драйверов **t11**. Отметим, что для алгоритмов обработки данных **DATA12** и **BLOCKDATA12** при проходе по базе выбираются каналы, имеющие указанные тип, подтип, дополнение к подтипу и два младшие байта удаленного адреса.

6. Перейти к пункту 1.

Завершение работы драйверов

При остановке монитора реального времени сначала по всем интерфейсам вызываются функции **StopMedia**. Затем по каждому используемому протоколу вызывается функция **Close_yyy**. После этого по каждому интерфейсу вызывается функция **CloseMedia**.

Использование для разработки драйверов других компиляторов

Разработка драйверов t11 и t12 возможна с использованием практически любого компилятора, который может корректно создавать WIN32 DLL библиотеки. При этом нужно учитывать следующие моменты:

- **Соглашение о вызове.** Функции, экспортируемые драйвером, должны быть объявлены как `__cdecl`.
- Функции в DLL описания протокола драйвера должны иметь совершенно определенные порядковые номера (ordinals). При этом их имена могут быть произвольными. Драйверы t11 и t12 используют для однотипных функций разные порядковые номера (ordinals).

Пример DEF-файла для типа 11:

```
; type 11
EXPORTS
Check_xxx @1
Get_xxx @2
Prepare_xxx @3
Set_xxx @4
zCompare_xxx @5
zReadAny_xxx @6
DllMain @7
```

Пример DEF-файла для типа 12:

```
; type 12
EXPORTS
Check_yyy @1
Close_yyy @2
Get_yyy @3
Open_yyy @4
Set_yyy @5
zCompare_yyy @6
DllMain @7
```

Таким образом, экспортируемые функции должны быть расположены в алфавитном порядке.

- Компилятор Borland C++ Builder 6.0 неправильно обрабатывает DEF-файлы. Добиться присвоения нужных ordinals удастся только в том случае, если определения функций расположены в алфавитном порядке в исходном файле.

- DLL описания интерфейса (**media<n>.dll**) также экспортирует функции по номерам, в алфавитном порядке.

Пример исходного текста для модуля описания носителя на Borland C++ Builder

```
//////////////////////////////////
#include <windows.h>
#include <stdio.h>

int __export CloseMedia(long &hndl)
{return 0;}

int __export OpenMedia(long &hndl)
{
    hndl = 1;
    return 0;
}

int __export ReadMedia(long &hndl, char *rbuf, int
in_count, int &actual_count)
{return 0;}

int __export StartMedia(long &hndl)
{return 0;}

int __export StopMedia(long &hndl)
{return 0;}

int __export WriteMedia(long &hndl, char *sbuf, int
out_count, int &actual_count)
{return 0;}

//////////////////////////////////
#pragma argsused
BOOL WINAPI DllMain(HINSTANCE hinstDLL, DWORD fwdrea-
son, LPVOID lpvReserved)
{return TRUE;}
//////////////////////////////////
```


Драйверы обмена с УСО в WINDOWS

Драйвер обмена с платами оформляется как DLL-модуль. Он должен иметь имя **rwh.dll** и находиться в директории, содержащей исполняемый файл MPB. Желательно, чтобы обращение к платам из драйвера было выделено в отдельный поток.

Каналы для вызова драйвера RWH

Вызов драйвера обмена с УСО осуществляется каналами (см. **Подтип 1**, **Подтип 2**) **AI_RWH** (чтение аналоговых данных), **AO_RWH** (запись аналоговых данных), **DI_RWH** (чтение дискретных данных) и дополнением **DO_RWH** (запись дискретных данных), а также **RWH** (OUTPUT – чтение данных, INPUT – запись данных).

Вызовы драйвера с помощью канала **RWH** будут работать только в том случае, если в базе имеется хотя бы один канал, инициализирующий драйвер (**AI_RWH**, **AO_RWH**, **DI_RWH** или **DO_RWH**).

Настройки этих каналов используются для формирования значения удаленного адреса, передаваемого драйверу при его вызове для запроса и передачи данных. При этом первая в списке настройка формирует значение двух младших байтов удаленного адреса. Остальные настройки формируют значения следующих по порядку байтов.

Удаленный адрес может быть сформирован также в самом драйвере.

Функции драйвера

Драйвер должен содержать набор функций, которые MPB вызывает при запуске, остановке, инициализации обмена и непосредственно для обмена данными. Ниже приводится описание этих функций.

Первый вызов драйвера

Эта функция не имеет параметров и вызывается при запуске монитора реального времени. Она имеет следующий формат:

```
void rwh_start()
```

Последний вызов драйвера

Эта функция не имеет параметров. Она вызывается при завершении работы монитора реального времени.

```
void rwh_stop();
```

Инициализация аналоговых сигналов

Эта функция вызывается для каждого канала **AI_RWH** или **AO_RWH** при запуске монитора реального времени. Она имеет следующий формат:

```
void rwh_aio_init(IA &ia);
```

где

ia – удаленный адрес канала.

Инициализация дискретных сигналов

Эта функция вызывается при запуске монитора реального времени для каждого канала **DI_RWH** или **DO_RWH**, у которого значение старшего байта удаленного адреса (атрибут **СОСТОЯНИЕ**) отлично от 0. Функция имеет следующий формат:

```
void rwh_dio_init(IA &ia);
```

где

ia – удаленный адрес канала.

Формирование аналоговых сигналов

Эта функция вызывается для каждого канала **AO_RWH** при его пересчете. Она имеет следующий формат:

```
int rwh_aio_write(IA &ia, unsigned short &v);
```

где

ia – удаленный адрес канала;

v – выходное значение канала.

Если возвращаемое данной функцией значение равно 0, то операция формирования сигнала считается завершенной успешно; если 1 – неуспешной. В последнем случае каналу устанавливается признак аппаратной недостоверности.

Опрос аналоговых сигналов

Эта функция вызывается для каждого канала **AI_RWH** при его пересчете. Она имеет следующий формат:

```
int rwh_aio_read(IA &ia, unsigned short &v);
```

где

ia – удаленный адрес канала;

v – значение, передаваемое на вход канала.

Если возвращаемое данной функцией значение равно 0, операция считывания сигнала считается завершенной успешно, а полученное число интерпретируется как 16-разрядное со знаком.

Если возвращаемое данной функцией значение равно 2, операция считывания сигнала считается завершенной успешно, а полученное число интерпретируется как 16-разрядное без знака.

Если возвращаемое данной функцией значение равно 1, операция считывания сигнала считается неуспешной. В этом случае каналу устанавливается признак аппаратной недостоверности.

Формирование дискретных сигналов

Эта функция вызывается для каждого канала **DO_RWH** при его пересчете. Она имеет следующий формат:

```
int rwh_dio_write(IA &ia, unsigned short &v);
```

где

ia – удаленный адрес канала;

v – выходное значение канала.

Если возвращаемое данной функцией значение равно 0, то операция формирования сигнала считается завершенной успешно, а при 1 – неуспешной. В последнем случае каналу устанавливается признак аппаратной недостоверности.

Опрос дискретных сигналов

Эта функция вызывается для каждого канала **DI_RWH** при его пересчете. Она имеет следующий формат:

```
int rwh_dio_read(IA &ia, unsigned short &v);
```

где

ia – удаленный адрес канала;

v – значение, передаваемое на вход канала.

Если возвращаемое данной функцией значение равно 0, то операция считывания сигнала считается завершенной успешно, а при 1 – неуспешной. В последнем случае каналу устанавливается признак аппаратной неадекватности.

Опрос атрибутов каналов

Эта функция вызывается при пересчете канала **RWH** типа INPUT. Она имеет следующий формат:

```
int rwh_atr_read(IA &ia, float &v);
```

где

ia – удаленный адрес канала. Этот параметр может быть задан настройками канала, вызывающего драйвер, а также сформирован в самом драйвере;

v – значение атрибута, передаваемое драйверу.

Если драйвер возвращает значение 1, то из параметра **ia** считываются номера канала и атрибута, и значение указанного атрибута присваивается параметру **v** для передачи драйверу. После этого драйвер вызывается снова.

Если драйвер возвращает значение 0, то выполняется переход к обработке следующего канала в базе.

Формирование атрибутов каналов

Эта функция вызывается при пересчете канала **RWH** типа OUTPUT. Она имеет следующий формат:

```
int rwh_atr_write(IA &ia, float &v);
```

где

ia – удаленный адрес канала. Этот параметр может быть задан настройками канала, вызывающего драйвер, а также сформирован в самом драйвере;

v – сформированное в драйвере значение, присваиваемое атрибуту.

Если драйвер возвращает значение 1, то из параметра **ia** считываются номера канала и атрибута, и значение параметра **v**, сформированное в драйвере, присваивается указанному атрибуту. После этого драйвер вызывается снова.

Если драйвер возвращает значение 0, значение параметра **v** присваивается указанному атрибуту, а затем выполняется переход к обработке следующего канала в базе.

Шаблон драйвера для WINDOWS

Текст драйвера

```
#define RWHIS
#include "rwh.h"
void rwh_start()
{
}
void rwh_stop()
{
}
void rwh_aio_init(IA &ia)
{
}
void rwh_dio_init(IA &ia)
{
}
int rwh_aio_write(IA &ia,unsigned short v)
{
    return(0);
}
int rwh_aio_read(IA &ia,unsigned short &v)
{
    v=0;
    return(0);
}
int rwh_dio_write(IA &ia,unsigned short v)
{
    return(0);
}
int rwh_dio_read(IA &ia,unsigned short &v)
{
    return(0);
}
int rwh_atr_read(IA &ia,float &v)
{
    return(0);
}
int rwh_atr_write(IA &ia,float &v)
{
    return(0);
}
```

Стандартный заголовок

```
#ifndef RWHIS
typedef union
{
    unsigned char c[6];
    unsigned short int i[3];
}
IA;
    __declspec(dllexport) void rwh_start();
    __declspec(dllexport) void rwh_stop();
    __declspec(dllexport) void rwh_aio_init(IA
&ia);
    __declspec(dllexport) void rwh_dio_init(IA
&ia);
    __declspec(dllexport) int rwh_aio_write(IA
&ia,unsigned short v);
    __declspec(dllexport) int rwh_aio_read(IA
&ia,unsigned short &v);
    __declspec(dllexport) int rwh_dio_write(IA
&ia,unsigned short v);
    __declspec(dllexport) int rwh_dio_read(IA
&ia,unsigned short &v);
    __declspec(dllexport) int rwh_atr_read(IA
&ia,float &v);
    __declspec(dllexport) int rwh_atr_write(IA
&ia,float &v);
#else
    __declspec(dllimport) void rwh_start();
    __declspec(dllimport) void rwh_stop();
    __declspec(dllimport) void rwh_aio_init(IA
&ia);
    __declspec(dllimport) void rwh_dio_init(IA
&ia);
    __declspec(dllimport) int rwh_aio_write(IA
&ia,unsigned short v);
    __declspec(dllimport) int rwh_aio_read(IA
&ia,unsigned short &v);
    __declspec(dllimport) int rwh_dio_write(IA
&ia,unsigned short v);
    __declspec(dllimport) int rwh_dio_read(IA
&ia,unsigned short &v);
    __declspec(dllimport) int rwh_atr_read(IA
&ia,float &v);
    __declspec(dllimport) int rwh_atr_write(IA
&ia,float &v);
#endif
```

Глава 12

T-FACTORY

Каналы T-FACTORY

Каналы T-FACTORY значительно отличаются от каналов других классов:

- атрибуты, общие для каналов всех классов (см. **Общие атрибуты каналов**), в каналах T-FACTORY могут иметь совершенно иное функциональное назначение;
- каналы T-FACTORY имеют очень большое количество специфических атрибутов.

При конфигурировании каналов T-FACTORY задаются ссылки на каналы CALL. Эти каналы могут иметь аргументы с виртуальной привязкой (см. **Привязка аргументов вручную**) – такие аргументы будут привязаны монитором к атрибутам канала T-FACTORY (по порядку).

Канал класса M-РЕСУРС

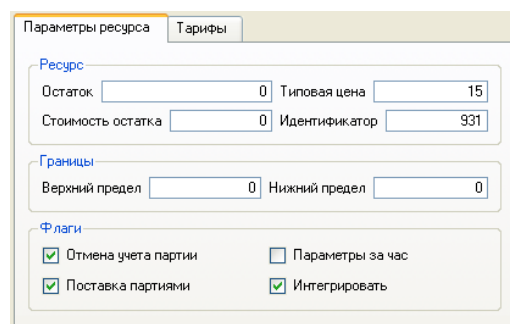
Канал класса **М-Ресурс** предназначен для учета любого вида материального ресурса в физическом и стоимостном выражении, а также брака.

Канал **М-Ресурс** может быть неинтегрирующим и интегрирующим, вследствие чего при учете, например, электроэнергии может работать как с текущими значениями потребленной энергии, так и с текущими значениями потребляемой мощности.

Редактор канала M-РЕСУРС

На вкладках редактора канала **М-Ресурс** задаются начальные значения некоторых его атрибутов (см. **Атрибуты канала M-РЕСУРС**).

Вкладка 'Параметры ресурса'



| Параметры ресурса | |
|---|---|
| Ресурс | |
| Остаток | 0 |
| Типовая цена | 15 |
| Стоимость остатка | 0 |
| Идентификатор | 931 |
| Границы | |
| Верхний предел | 0 |
| Нижний предел | 0 |
| Флаги | |
| <input checked="" type="checkbox"/> Отмена учета партии | <input type="checkbox"/> Параметры за час |
| <input checked="" type="checkbox"/> Поставка партиями | <input checked="" type="checkbox"/> Интегрировать |

В разделе **Ресурс** этой вкладки задаются начальные значения следующих атрибутов:

- **Остаток** – остаток ресурса (количество), **(000) R**;
- **Стоимость остатка** – стоимость остатка ресурса, **(030) COST_is**.

При старте монитора **(026) Price_In = COST_is / R**;

- **Типовая цена** – типовая цена ресурса, **(027) Price_Def**;
- **Идентификатор** – идентификатор (номер) ресурса, **(033) ResID**.

В разделе **Границы** задаются начальные значения следующих атрибутов:

- **Верхний предел** – максимально допустимое количество ресурса, **(028) HA**;
- **Нижний предел** – минимально допустимое количество ресурса, **(029) LA**.

В разделе **Флаги** задаются начальные значения следующих атрибутов:

- **Отмена учета партий** – флаг, при установке которого **(084) Qio=1**, и в канале не ведется учет партий ресурса;
- **Поставка партиями** – флаг, при установке которого **(040) IntDif=1**, и значение атрибута **In** канала интерпретируется как объем поступившего ресурса в случае неинтегрирующего канала или объем ресурса, пришедший за последний такт пересчета, в случае интегрирующего канала. Если **IntDif=1**, на следующем такте пересчета **In** обнуляется. Если флаг не установлен, **IntDif=0**, и значение атрибута **In** канала интерпретируется как текущее значение некоторого нарастающего счетчика прихода, и для обработки в канале вычисляется величина $\Delta = In_{cur} - In_{prev}$, где **In_{cur}** – текущее значение входа, **In_{prev}** – значение входа на предыдущем такте пересчета. Если **IntDif=0**, **HA=9999**, **99999** или **999999** и **In_{cur} < In_{prev}**, то $\Delta = HA + In_{cur} - In_{prev} + 1$; при этом если $\Delta > LA$, то **(007) P=1**, в противном случае **P=0**;
- **Параметры на час** – флаг, при установке которого **(085) dTOff=1**, и в интегрирующем канале изменяется алгоритм интегрирования;
- **Интегрировать** – флаг, при установке которого **(053) multdT=1**, и канал интегрирует входной сигнал.

Вкладка ‘Тарифы’

На этой вкладке задаются начальные значения следующих атрибутов:

- **Индекс тарифа** – индекс тарифной сетки, **(055) tarif**. Если **Индекс тарифа = N**, **tarif = N-1**;
- **01-24** – почасовая тарифная сетка, **01** соответствует стоимости единицы потребленной электроэнергии в интервале 00-01ч; **(150-173) t0-t23**;

- **Принять тариф** – флаг загрузки тарифной сетки монитором: 1 – загружать и использовать; 0 – не загружать. Если этот флаг установлен, на вход канала подается текущее значение потребленной электроэнергии или текущее значение потребляемой мощности, и стоимостные атрибуты канала рассчитываются исходя из тарифной сетки. В реальном времени тарифы (атрибуты **(150-173) t0-t23**) могут быть изменены – для их принятия нужно подать ненулевое число в атрибут **(120) АСК**.

Вид вкладки показан на рисунке:

Флаги канала М-РЕСУРС

Сочетания установленных флагов канала **М-Ресурс** задают следующее назначение канала:

| Отмена учета партий | Поставка партиями | Интегрировать | Назначение канала |
|---------------------|-------------------|---------------|---|
| + | + | + | Интегрирующий |
| + | + | | Неинтегрирующий, стек партий не ведется |
| | + | | Неинтегрирующий, стек партий ведется |

Флаг **Параметры на час** изменяет формулу интегрирования в канале (см. **Атрибуты канала М-РЕСУРС**).

Флаг **Принять тариф** влияет на расчет стоимостных атрибутов канала.

Атрибуты канала М-РЕСУРС

При описании атрибутов канала **М-Ресурс** указаны соответствующие поля редактора (см. **Редактор канала М-РЕСУРС**):

- **(000) R** – остаток ресурса (количество). Начальное значение этого атрибута задается на вкладке **Параметры ресурса** редактора (поле **Остаток**). В дальнейшем значение атрибута вычисляется автоматически:
 - в неинтегрирующем канале: $R=R+In-Q$;
 - в интегрирующем канале с **(085) dTOff=0**: $R=R+In*T-Q$ (T – период пересчета канала в секундах, перевод периода в секунды выполняется автоматически);
 - в интегрирующем канале с **(085) dTOff=1**: $R=R+In*T/3600-Q$;

Приведенные формулы справедливы для такта пересчета, на котором в **Q** подано значение (в том числе и равное предыдущему), в противном случае эту величину из формул нужно убрать;

- **(001) A** – не используется;
- **(002) In** – в этот атрибут подается количество пришедшего ресурса (положительное число) в случае неинтегрирующего канала или величина производной количества – в случае интегрирующего канала;
- **(003-008)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**).

Атрибут **(006, D) Тенденция** не вычисляется.

Атрибут **(007, P)** указывает интервал, в котором находится значение атрибута **R**:

- $P=1$, если $R \geq HA$;
- $P=0$, если $LA < R \leq HA$;
- $P=2$, если $R \leq LA$;
- **(009) Q** – в этот атрибут подается положительное значение, интерпретируемое как запрос на выдачу ресурса (количество). Если $Q \leq R$, запрос выполняется (уменьшается остаток, его стоимость и т.д.), в противном случае запрос не выполняется (параметры ресурса не изменяются);
- **(010) In2** – этот атрибут хранит значение прихода ресурса;
- **(026) Price_In** – в этот атрибут подается значение, соответствующее цене поступившего ресурса. Если эта цена задана, стоимость поступившего ресурса рассчитывается по ней (если в канале не используется тарифная сетка);
- **(027) Price_Def** – типовая цена ресурса (**Параметры ресурса / Типовая цена**). Если не задана цена **Price_In**, стоимость поступившего ресурса рассчитывается по **Price_Def** (если в канале не используется тарифная сетка);
- **(028) HA** – максимально допустимое количество ресурса (**Параметры ресурса / Верхний предел**). Если $R > HA$, соответству-

ющее сообщение заносится в отчет тревог;

Сообщения по каналу М-РЕСУРС заносятся в ОТ также в некоторых других случаях (см. **Сообщения по каналу М-РЕСУРС**).

- **(029) LA** – минимально допустимое количество ресурса (**Параметры ресурса / Нижний предел**). Если $R < LA$, соответствующее сообщение заносится в отчет тревог;
- **(030) COST_is** – стоимость остатка ресурса. Начальное значение этого атрибута задается на вкладке **Параметры ресурса** редактора (поле **Стоимость остатка**), в дальнейшем вычисляется автоматически;
- **(031) PH_is** $\equiv R$;
- **(032) Cost_Out** – стоимость выданного количества ресурса, вычисляется автоматически при выполнении запроса на выдачу:
 - в канале без учета партий (атрибут **084, Qio** равен 1) выдача ресурса производится по средней цене, т.е. **Cost_Out** $= (Cost_is/R)*Q$;
 - в канале с учетом партий (**Qio** = 0) запрошенное количество ресурса формируется из партий по алгоритму FIFO по цене партий;
- **(033) ResID** – идентификатор (номер) ресурса (**Параметры ресурса / Идентификатор**);
- **(035) Q_of_in** – автоматически вычисляемый атрибут. Если значение атрибута (**084, Qio**) равно 0 (т.е. учет партий ведется), **Q_of_in** равно числу оставшихся в наличии партий ресурса (другими словами, **Q_of_in** индицирует текущую глубину стека FIFO, в котором хранятся данные о партиях);
- **(036) in_id** – в этот атрибут подается значение, соответствующее ID поставщика ресурса;
- **(037) out_id** – автоматически вычисляемый атрибут. Если атрибут **084, Qio** равен 0 (учет партий ведется), при выдаче ресурса **out_id** индицирует ID поставщика последней партии, которая задействована для формирования запрошенного количества ресурса;
- **(038, 039)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(040) IntDif** – флаг **Поставка партиями** (см. **Редактор канала М-РЕСУРС**);
- **(041-049)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(051) bIO** – не используется;
- **(052) FS** – не используется;

- **(053) multdT** – флаг **Интегрировать** (см. **Редактор канала М-РЕСУРС**);
- **(054) in_POS** – вход; значение этого атрибута задает позицию (начиная с 0) в стеке FIFO, в котором хранятся данные об оставшихся партиях ресурса (стек инициализируется в канале, если значение атрибута **084, Qio** равно 0). Данные из указанной с помощью **in_POS** позиции стека записываются в атрибуты **128, i_ph** (объем партии); **129, i_price** (цена ресурса в партии); **130, i_time** (время прихода партии). В атрибут **131, o_ph** данная информация записывается в комбинированном виде:

```

i_ph : 100
i_price : 15
i_time : 29.11.2004 17:15:45
o_ph : 29.11.2004 17:15:45 100 15

```

Данные о партиях (кроме ID поставщика) записываются в автоматически создаваемые аргументы канала:

```

ArgSize : 24
A000 : 29.11.2004 17:24:20
A001 : 100 (Q)
A002 : 10 (Price)
A003 : 01.01.1970 3:00:33
A004 : 29.11.2004 17:24:28
A005 : 100 (Q)
A006 : 20 (Price)
A007 : 01.01.1970 3:00:33
A008 : 29.11.2004 17:24:34
A009 : 100 (Q)
A010 : 30 (Price)
A011 : 01.01.1970 3:00:33
A012 : 29.11.2004 17:24:42
A013 : 100 (Q)
A014 : 40 (Price)
A015 : 01.01.1970 3:00:33
RInd : 35

```

Если **(040) IntDif=0** и **(084) Qio=0**, стек хранит архив значений канала;

- **(055) tarif** – индекс тарифной сетки (**Тарифы / Индекс тарифа**), натуральное число. Если в редакторе **Индекс тарифа = N**, в мониторе **tarif = N-1**;
- **(056-061, 078-083)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(084) Qio** – флаг **Отмена учета партий** (см. **Редактор канала М-РЕСУРС**);
- **(085) dTOff** – флаг **Параметры на час** (см. **Редактор канала М-РЕСУРС**);
- **(086-098)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(099) ID_RES_L** – этот атрибут индицирует имя (во внутреннем

представлении – ID) канала М-РЕСУРС, который привязан к данному каналу;

- **(100) sPh_in** – суммарный приход ресурса (количество), автоматически вычисляемый атрибут;
- **(101) sCost_in** – суммарная стоимость поступившего ресурса, автоматически вычисляемый атрибут.
Если в канале не используется тарифная сетка, стоимость поступившего ресурса рассчитывается по **Price_In** или **Price_Def**, в противном случае – по соответствующему тарифу;
- **(102) sPh_out** – общее количество выданного ресурса, автоматически вычисляемый атрибут;
- **(103) sCost_out** – общая стоимость выданного ресурса, автоматически вычисляемый атрибут;
- **(118-119)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(120) АСК** – чтобы принять изменения, внесенные в тарифную сетку в реальном времени, в этот атрибут нужно подать ненулевое значение;
- **(123-127)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(128) i_ph** – в этот атрибут из стека выводится объем партии, указанной атрибутом **(054) in_POS**;
- **(129) i_price** – в этот атрибут из стека выводится цена ресурса в партии, указанной атрибутом **(054) in_POS**;
- **(130) i_time** – в этот атрибут из стека выводится время прихода партии, указанной атрибутом **(054) in_POS**;
- **(131) o_ph** – в этот атрибут из стека в комбинированном виде выводится информация о партии, указанной атрибутом **(054) in_POS**;
- **(132) o_cost** – зарезервирован;
- **(133) o_time** – зарезервирован;
- **(150-173) t0-t23** – почасовая тарифная сетка (**Тарифы / 01-24**). **t0** соответствует тарифу за промежуток 00-01ч.

В профайлере канал **М-Ресурс** индицируется как **C8_M-Resource** (атрибут 126, **TsT**).

Обобщение данных по ресурсу

Если в узле создано несколько каналов М-РЕСУРС (пусть они имеют имена **res<N>**), учитывающих ресурс с одним и тем же ID, обобщенную информацию по ресурсу можно получить с помощью дополнительного канала М-РЕСУРС (пусть он имеет имя **sum**) со следующими начальными

ми настройками:

- **(027) Price_Def = (030) COST_is = -1**
- **(033) ResID = ID**, где **ID** – идентификатор ресурса, по которому производится обобщение данных.

Канал-сумматор М-РЕСУРС имеет другой номер подтипа (см. **Подтипы каналов**) и работает по следующему алгоритму (вне зависимости от флагов):

$$\text{sum.R} = \text{sum.Ph_is} = \sum \text{res}_i . \text{R}$$

$$\text{sum.sPh_in} = \sum \text{res}_i . \text{sPh_in}$$

$$\text{sum.sCost_in} = \sum \text{res}_i . \text{sCost_in}$$

$$\text{sum.sPh_out} = \sum \text{res}_i . \text{sPh_out}$$

$$\text{sum.sCost_out} = \sum \text{res}_i . \text{sCost_out}$$

$$\text{sum.Price_Def} = \frac{\text{sum.sCost_out}}{\text{sum.sPh_out}}$$

$$\text{sum.Price_in} = \frac{\text{sum.sCost_in} - \text{sum.sCost_out}}{\text{sum.R}}$$

Аналитический учет потребления ресурса

Для аналитического учета потребления ресурса нужно создать для каждого потребителя канал М-РЕСУРС (например, **unit1**, **unit2**, **unit3**) и привязать его к каналу М-РЕСУРС (например, **spirit**), учитывающему ресурс.

В этом случае запрос на выдачу ресурса подается в атрибут **(2, In)** канала **unit<N>**. Если запрошенное количество имеется, запрос выполняется. Если запрошенного количества ресурса нет в наличии, каналу **unit<N>** устанавливается признак аппаратной недостоверности.

Канал CALL с типом вызова MRESOURCE_1

Канал класса CALL с типом вызова MRESOURCE_1 (тип такого канала не имеет значения) используется для управления несколькими каналами класса М-РЕСУРС. Для управления используются аргументы канала CALL, создаваемые группами по 4.

Для прихода первый аргумент группы должен иметь тип OUTPUT (в приведенных ниже номерах аргументов **k=0,1...**):

- **arg<4k>** – привязка канала М-РЕСУРС;

- **arg<4k+1>** – количество приходуемого ресурса (передается в атрибут **In** привязанного канала М-РЕСУРС);
- **arg<4k+2>** – цена приходуемого ресурса (передается в атрибут **Price_In** привязанного канала М-РЕСУРС);
- **arg<4k+3>** – ID поставщика (передается в атрибут **in_id** привязанного канала М-РЕСУРС).

Для расхода первый аргумент группы должен иметь тип INPUT:

- **arg<4k>** – привязка канала М-РЕСУРС;
- **arg<4k+1>** – количество запрашиваемого ресурса (передается в атрибут **Q** привязанного канала М-РЕСУРС);
- **arg<4k+2>** – при выполнении запроса в этот аргумент записывается цена ресурса (рассчитывается в привязанном канале М-РЕСУРС);
- **arg<4k+3>** – при выполнении запроса в этот аргумент записывается значение атрибута **out_id** привязанного канала М-РЕСУРС.

Если хотя бы один запрос не может быть выполнен, каналу CALL устанавливается признак аппаратной недостоверности.

Значение канала CALL задает число операций прихода/расхода (при каждой отработке канала CALL выполняется одна операция по всем привязанным каналам М-РЕСУРС).

Канал класса ЕДИНИЦА ОБОРУДОВАНИЯ

Канал класса **ЕДИНИЦА ОБОРУДОВАНИЯ** предназначен для учета оборудования и вычисления ряда его характеристик в процессе эксплуатации.

Канал **ЕДИНИЦА ОБОРУДОВАНИЯ**, имя которого совпадает с именем группы, в которую он входит, выполняет особые функции:

- если для канала заданы паспортное имя и номер паспорта, он играет роль сборочной единицы – по отношению к такому каналу все другие каналы **ЕДИНИЦА ОБОРУДОВАНИЯ** группы являются составными частями;
- если для канала не заданы паспортное имя и номер паспорта, в его атрибуты 27, 28, 20, 26, 36, 65, 21, 37, 17, 23, 29, 32, 30 и 31 записываются суммы значений тех же атрибутов других каналов **ЕДИНИЦА ОБОРУДОВАНИЯ** группы.

Редактор канала ЕДИНИЦА ОБОРУДОВАНИЯ

В разделе **Параметры**, на вкладке **Параметры объекта** и вкладках **Сервис** редактора канала **ЕДИНИЦА ОБОРУДОВАНИЯ** задаются начальные значения некоторых атрибутов канала (см. **Атрибуты канала**

ЕДИНИЦА ОБОРУДОВАНИЯ).

Раздел 'Параметры'

The screenshot shows a web-based form titled "Параметры" (Parameters). It contains several sections with input fields and dropdown menus:

- Параметры (Parameters):**
 - Паспортное имя (Passport name): text input
 - Номер паспорта (Passport number): text input
 - Код (Code): text input with value "0"
 - Производитель (Manufacturer): text input
 - Дата выпуска (Issue date): date picker with value "00.00.0000"
 - Срок службы (Service life): text input with value "0d0h0m0s"
 - Максимально допустимая выработка (в физ. ед.) (Maximum permissible production (in phys. units)): text input with value "0"
 - Наработка на отказ (Mean time to failure): text input with value "0d0h0m0s"
 - Фотография (Photo): checkbox
- Приобретение (Acquisition):**
 - Дата (Date): date picker with value "00.00.0000"
 - Цена (Price): text input with value "0"
- Производительность (Productivity):**
 - Объем (физ. ед.) (Volume (phys. units)): text input with value "0"
 - За период (For period): dropdown menu with value "Секунда" (Second)
- Документ о принятии (Document of acceptance):**
 - Дата (Date): date picker with value "00.00.0000"
 - Имя (Name): text input
 - Номер (Number): text input with value "0"
- Документ о выбытии (Document of withdrawal):**
 - Дата (Date): date picker with value "00.00.0000"
 - Имя (Name): text input
 - Номер (Number): text input with value "0"

В этом разделе задаются начальные значения следующих атрибутов:

- **Паспортное имя** – наименование по паспорту, **(099) UName**;
- **Номер паспорта** – номер паспорта, **(102) PasNum**;
- **Код** – код, **(100) FCode**;
- **Производитель** – производитель, **(105) MName**;
- **Дата выпуска** – дата выпуска, **(106) Make_Data**;
- **Срок службы** – срок службы по паспорту, **(012) TimeMaxW**;
- **Максимально допустимая выработка** – максимально допустимая выработка по паспорту, **(013) PHMax**;
- **Наработка на отказ** – наработка на отказ по паспорту, **(107) AvgToErrP**;
- **Фотография** – флаг разрешения использования фотографии – файла с именем, совпадающим с именем канала, **(101) Foto** (см. также ГЭ 'Фотография');
- **Объем** – выработка за интервал, заданный атрибутом **За период** (**(015) NormaPeriod**) по паспорту, **(014) PHNorma**;
- **За период** – временной интервал, **(015) NormaPeriod**. Величина **Объем / Период** – производительность по паспорту;
- **Дата приобретения** – дата покупки, **(116) DataB**;
- **Цена приобретения** – начальная стоимость оборудования, **(117) CostB**.

Параметры документа о вводе в эксплуатацию задаются в одноименном разделе:

- **Дата** – дата ввода в эксплуатацию, **(011) DataIn**;
- **Номер** – номер документа о вводе в эксплуатацию, **(108) DocInNum**;
- **Имя** – наименование документа о вводе в эксплуатацию, **(109) DocInName**.

Параметры документа о списании (выбытии) задаются в одноименном разделе:

- **Дата** – дата списания, **(114) DataOut**;
- **Номер** – номер документа о списании, **(111) DocOutNum**;
- **Имя** – наименование документа о списании, **(112) DocOutName**.

Вкладка 'Параметры объекта'

На этой вкладке задаются начальные значения следующих атрибутов:

- **Тип объекта** – тип оборудования: обслуживаемое (**010, SrvType=0**) или необслуживаемое (**SrvType=1**). Для необслуживаемого оборудования канал D-РЕСУРС автоматически не создается;
- **Ответственный за эксплуатацию** – ответственный за эксплуатацию (ссылка на канал **Персонал**);
- **Приоритет обслуживания** – приоритет техобслуживания, **(071) SRV_PRIOR**;
- **Интервал** – временной интервал, **(035) Period_I**. Эта величина задает период вычисления интервальных атрибутов;
- **Статус по потомку** – если этот флаг установлен для сборки (**084, By_Child=1**), ее статус автоматически устанавливается по статусу с наибольшим приоритетом из тех, которые имеют входя-

щие в сборку единицы. Ниже статусы расположены по убыванию приоритета для данного флага:

- WORK
- RESERVE
- IDLE
- REPAIR
- SERVICE
- OFF
- ERROR

Флаг **Статус по потомку** имеет более низкий приоритет по сравнению с флагом **Статус для родителя**;

- **Статус по родителю** – если этот флаг установлен (**053, By_Parent=1**), оборудование принимает статус сборки;
- **Статус для родителя** – если в состав сборки входит несколько единиц оборудования с установленным флагом **Статус для родителя** (**085, For_Parent=1**), статус сборки устанавливается по той единице, которая имеет статус с наибольшим приоритетом. Ниже статусы расположены по убыванию приоритета для данного флага:

- ERROR
- OFF
- REPAIR
- SERVICE
- IDLE

Статусы WORK и RESERVE не анализируются. Флаг **Статус для родителя** имеет более низкий приоритет, чем флаг **Статус по родителю** и более высокий, чем флаг **Статус по потомку**;

- **Дата капитального ремонта** – дата капитального ремонта, (**115) DataCapRep**;
- **Задание статуса** – задание начального статуса, (**002) In**;
- **Вид оборудования** – коррекция статуса в случае привязки канала к статусу FBD-блока управления устройством, (**070) SRV_MASK (units_subkind.tmc)**:
 - 0 – нет коррекции;
 - 1 – MOTOR;
 - 2 – KLP или ZDV;
- **Алгоритм** – алгоритм, (**034) Algorithm**;
- **При старте** – ссылка на канал CALL, обрабатывается при старте любого из ТО, (**122) Alr1_ID**.
- **По завершении** – ссылка на канал CALL, обрабатывается при окончании любого из ТО, (**054) AlgEnd_ID**;

- **При аварии** – ссылка на канал CALL, обрабатывается при аварии, (121) **Alr0_ID**;
- **Произвольно** – ссылка на канал CALL, обрабатывается при посылке любого значения в атрибут **9, Q, (055) AlgPlan_ID**;
- **Особенности эксплуатации** – особенности эксплуатации оборудования (комментарий), (103) **SpecDoc**.

Вкладки 'Сервис'

На данных вкладках задаются начальные значения некоторых атрибутов ТО (из диапазонов номеров **128-146, 148-166, 168-186** и **188-206**):

- **Тип** – тип планирования ТО, (128) **0_Type**;
- **Дата исполнения последнего** – дата исполнения последнего ТО, (132) **0_LastData**,
- **Статус последнего** – статус последнего ТО, (131) **0_PSts**;
- **Время выдачи задания до старта** – разница между временем выдачи задания на ТО и временем начала ТО, (140) **0_GenConst**;
- **Константа сравнения** – константа сравнения, при задании времени – в секундах, (139) **0_CmpConst**;
- **Время начала** – дата и время начала ТО, (133) **0_Start**;
- **Время окончания** – дата и время окончания ТО, (134) **0_Finish**;
- **Длительность** – плановая длительность ТО, (135) **0_pLen**;
- **Стоимость** – плановая стоимость ТО, (137) **0_pCost**.

В разделе «Ссылки» задаются ссылки на каналы CALL:

- **1** – (142) **0_Link0**;
- **2** – (143) **0_Link1**;
- **3** – (144) **0_Link2**.

Ссылки 1 и 2 обрабатываются при достижении времени выдачи задания на работу, ссылка 3 – по окончании работы (при статусе сервиса 6, FINISH).

В случае обслуживаемого оборудования для каждого сконфигурирован-

ного сервиса монитор автоматически создает соответствующий канал **D-РЕСУРС**. В отличие от ручного создания канала **D-РЕСУРС**, при задании ТО в канале **ЕДИНИЦА ОБОРУДОВАНИЯ** нельзя задать ссылки на необходимые ресурсы. Кроме того, привязка диаграммы Ганта к каналу **ЕДИНИЦА ОБОРУДОВАНИЯ**, для которой сконфигурировано несколько ТО, означает привязку к одному ТО (первому – см. ГЭ 'Диаграмма Ганта').

Атрибуты канала ЕДИНИЦА ОБОРУДОВАНИЯ

При описании атрибутов канала **ЕДИНИЦА ОБОРУДОВАНИЯ** указаны соответствующие поля редактора (см. **Редактор канала ЕДИНИЦА ОБОРУДОВАНИЯ**):

- **(000) R** – текущий статус;
- **(001) A** – не используется;
- **(002) In** – в этот атрибут подается значение статуса:
 - 0 – UNKNOWN (не определен);
 - 1 – WORK (работа);
 - 2 – IDLE (ожидание, простой);
 - 3 – RESERVE (резерв);
 - 4 – ERROR (ошибка);
 - 5 – SERVICE (сервис);
 - 6 – REPAIR (ремонт);
 - 7 – OFF (выключено);
- **(003-006)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(007, P)** – не вычисляется;
- **(008, W)** – этот атрибут имеет стандартное назначение (см. **Общие атрибуты каналов**);
- **(009) Q** – при посылке любого значения в этот атрибут выполняется ссылка 55 (**Параметры объекта / Произвольно**);
- **(010) SrvType** – тип оборудования: обслуживаемое (**010, SrvType=0**) или необслуживаемое (**SrvType=1**) (**Параметры объекта/Тип объекта**);
- **(011) DataIn** – дата ввода в эксплуатацию (**Параметры/Документ о принятии/Дата**);
- **(012) TimeMaxW** – срок службы оборудования по паспорту (**Параметры/Срок службы**);
- **(013) PHMax** – максимально допустимая выработка по паспорту (**Параметры/Максимально допустимая выработка**);
- **(014) PHNorma** – выработка за интервал **(015) NormaPeriod** по

паспорту (**Параметры/Объем**);

- **(015) NormaPeriod** – временной интервал (**Параметры / За период**), автоматически переводится в число секунд. Величина **PHNorma/NormaPeriod** – паспортное значение производительности;
- **(016) PH_in** – значение этого атрибута интерпретируется как выработка за период пересчета;
- **(017) PH_Total** – текущее значение суммарной выработки (в статусе WORK): $PH_Total = PH_Total + PH_in$. Если на вход **PH_in** сигнал не подавался, $PH_Total = PH_Total + T * PHNorma / NormaPeriod$ (**T** – период пересчета канала, автоматически переводится в число секунд);
- **(018) PH_Few** – суммарная выработка за текущий интервал;
- **(019) PH_Previous** – суммарная выработка за предыдущий интервал;
- **(020) Cost_Total** – сумма расходов на ТО;
- **(021) Cost_Few** – расходы на ТО за текущий интервал;
- **(022) Cost_Previous** – расходы на ТО за предыдущий интервал;
- **(023) Time_W** – суммарное время работы (WORK & RESERVE);
- **(024) Time_Few_W** – время работы в текущем интервале;
- **(025) Time_Prev_W** – время работы в предыдущем интервале;
- **(026) Day_Avr_W** – среднесуточное время работы;
- **(027) Use_Total** – коэффициент использования, $Use_Total = Time_W / TraceTime$;
- **(028) PH_iznos** – фактический коэффициент износа, $PH_iznos = PH_Total / PHMax$;
- **(029) Idle_T** – суммарное время простоя (IDLE);
- **(030) Error_T** – суммарное время нахождения в аварийном состоянии (ERROR);
- **(031) Srv_T** – суммарное время техобслуживания и ремонта (SERVICE & REPAIR);
- **(032) OFF_T** – суммарное время нахождения в выключенном состоянии (OFF);
- **(033) UnStop_T** – время безостановочной работы (WORK);
- **(034) Algorithm** – алгоритм (**Параметры объекта/Алгоритм**). Установка битов этого атрибута в 1 соответствует следующим алгоритмам работы канала:
 - бит 0 – в статусе RESERVE считается счетчик **(029) Idle_T**;
 - бит 1 – в статусе IDLE считается счетчик **(063) W_a_Srv**;
 - бит 2 – в статусе IDLE считается счетчик **(067) W_to_err**;

- бит 3 – в статусе IDLE считается счетчик **(023) Time_W**;
- бит 4 – в статусе SERVICE не считается фактическое время ТО;
- бит 5 – статус автоматически переведется в SERVICE при переводе статуса ТО в START;
- бит 6 – снять необходимость подтверждения получения задания на ТО;
- бит 7 – изменение алгоритма вычисления общего времени наблюдения за оборудованием (атрибута **250, TraceTime**);
- **(035) Period_I** – интервал (**Параметры объекта/Интервал (sts_PERIOD_FP.tmc)**):
 - 2 – минута;
 - 4 – час;
 - 8 – день;
 - 16 – неделя;
 - 32 – месяц;
 - 64 – год.

Значение, большее 32, задает интервал в секундах;

- **(036) Srv_Avr_T** – среднее время ТО/ремонта, **Srv_Avr_T = Srv_T / Srv_Count**;
- **(037) Day_Few_W** – время работы за текущий день;
- **(038, 039)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(040) Mode** – зарезервировано;
- **(041-049, 051)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(052) FS** – не вычисляется;
- **(053) By_Parent** – если **By_Parent=1**, оборудование принимает статус сборки (**Параметры объекта/Статус по родителю**);
- **(054) AlgEnd_ID** – ссылка по завершении ТО (**Параметры объекта/По завершении**);
- **(055) AlgPlan_ID** – ссылка по посылке значения в атрибут 9 (**Параметры объекта/Произвольно**);
- **(056–061)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(062) PH_a_Srv** – выработка после последнего ТО;
- **(063) W_a_Srv** – время работы после последнего ТО;
- **(064) Srv_Count** – счетчик ТО и ремонтов;
- **(065) AvrToErr_T** – фактическая наработка на отказ, **W_to_err /**

Err_Count;

- **(066) Err_Count** – число переходов в статус ERROR. Переход из ERROR в WORK (RESERVE) интерпретируется как ошибка датчика аварии, поэтому в случае такого перехода счетчик ошибок уменьшается на 1;
- **(067) W_to_err** – суммарное время работы до ошибки (WORK & RESERVE);
- **(070) SRV_MASK** – вид коррекции статуса оборудования (**Параметры объекта/Вид оборудования**);
- **(071) SRV_PRIOR** – приоритет ТО (**Параметры/Приоритет обслуживания**):
 - 0 – пониженный;
 - 1 – обычный;
 - 2 – повышенный;
 - 3 – высокий;
- **(072) ID_HUMO_SRV** – ответственный за эксплуатацию (ссылка на канал Персонал) (**Параметры объекта/Ответственный за эксплуатацию**);
- **(076) Last_CLC** – время последнего пересчета канала;
- **(077) FewSTS_T** – время нахождения в текущем статусе;
- **(078–083)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(084) By_Child** – флаг задания статуса сборки по потомку (**Параметры объекта/Статус по потомку**);
- **(085) For_Parent** – флаг установки статуса родителя (**Параметры объекта/Статус для родителя**);
- **(086, 087)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(089–098)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(099) UName** – паспортное имя (**Параметры/Паспортное имя**);
- **(100) FCode** – код оборудования (**Параметры/Код**);
- **(101) Foto** – флаг разрешения использования фотографии (**Параметры/Фотография**);
- **(102) PasNum** – номер паспорта оборудования (**Параметры/Номер паспорта**);
- **(103) SpecDoc** – особенности эксплуатации оборудования (**Параметры объекта/Особенности эксплуатации**);
- **(105) MName** – производитель оборудования (**Параметры/Производитель**);

- **(106) Make_Data** – дата изготовления оборудования (**Параметры/Дата выпуска**);
- **(108) DocInNum** – номер документа о вводе в эксплуатацию (**Параметры/Документ о принятии/Номер**);
- **(109) DocInName** – наименование документа о вводе в эксплуатацию (**Параметры/Документ о принятии/Имя**);
- **(111) DocOutNum** – номер документа о списании/выбытии (**Параметры/Документ о списании/Номер**);
- **(112) DocOutName** – наименование документа о списании/выбытии (**Параметры/Документ о списании/Имя**);
- **(114) DataOut** – дата списания (**Параметры/Документ о списании/Дата**);
- **(115) DataCapRep** – дата капремонта (**Параметры объекта/Дата капитального ремонта**);
- **(116) DataB** – дата покупки оборудования (**Параметры/Дата приобретения**);
- **(117) CostB** – стоимость покупки оборудования (**Параметры/Цена приобретения**);
- **(118–120)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(121) Alr0_ID** – ссылка при аварии (**Параметры объекта/При аварии**);
- **(122) Alr1_ID** – ссылка при старте ТО (**Параметры объекта/При старте**);
- **(123–127)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(128) 0_Type** – тип планирования обслуживания (**Сервис/Тип**);
- **(129) 0_Sts0** – статус ТО:
 - 0 – **Норма**;
 - 1 – **Не установлен**;
 - 2 – **Не спланирован**;
 - 3 – **Не утвержден**;
 - 4 – **Не выдан**;
 - 5 – **Не подтвержден**;
 - 6 – **Не начат**;
 - 7 – **Не закончен**;

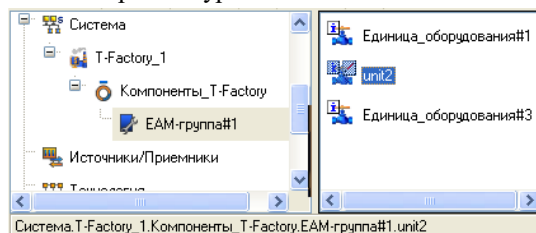
Статус может быть задан непосредственной установкой этого атрибута, однако в этом случае корректность переходов между статусами ТО не проверяется;
- **(130) 0_Sts1** – мониторинг статуса ТО;

- (131) **0_PSts** – статус предыдущего обслуживания (**Сервис/Статус последнего**);
- (132) **0_LastData** – дата и время последнего ТО (**Сервис/Дата исполнения последнего**);
- (133) **0_Start** – дата и время начала ТО (**Сервис/Время начала**);
- (134) **0_Finish** – дата и время окончания ТО (**Сервис/Время окончания**);
- (135) **0_pLen** – плановая длительность ТО (**Сервис / Длительность**);
- (136) **0_fLen** – фактическая длительность ТО;
- (137) **0_pCost** – плановая стоимость ТО (**Сервис/Стоимость**);
- (138) **0_fCost** – фактическая стоимость ТО;
- (139) **0_CmpConst** – константа сравнения (**Сервис/Константа сравнения**);
- (140) **0_GenConst** – разница между временем выдачи задания на ТО и временем начала ТО (**Сервис/Время выдачи задания до старта**), переведенное в секунды;
- (141) **0_TimeBefore** – текущее время до начала ТО;
- (142) **0_Link0** – ссылка 1 (**Сервис**);
- (143) **0_Link1** – ссылка 2 (**Сервис**);
- (144) **0_Link2** – ссылка 3 (**Сервис**);
- (146) **0_SetSts** – задание статуса ТО с проверкой корректности переходов. Кроме 0-7, допускаются следующие значения:
 - 8 – выдача немедленно при переходе в статус **Выдано**;
 - 9 – через час;
 - 10 – через день;
 - 11 – за день до;
 - 12 – за неделю до;
- (148-166) – аналог (128-146) для Сервиса2;
- (168-186) – аналог (128-146) для Сервиса3;
- (188-206) – аналог (128-146) для Сервиса4;
- (225) **kU0** – интервальный коэффициент. использования 1 (верхний в стеке);
- (226) **kU1** – интервальный коэффициент. использования 2;
- (227) **kU2** – интервальный коэффициент. использования 3;
- (228) **kU3** – интервальный коэффициент. использования 4;
- (229) **kU4** – интервальный коэффициент. использования 5;

- (230) **kU5** – интервальный коэффициент. использования 6;
- (231) **kU6** – интервальный коэффициент. использования 7;
- (232) **kU7** – интервальный коэффициент. использования 8;
- (236) **pVer0** – прогноз работоспособности;
- (237) **pVer1** – прогноз аварийного состояния;
- (238) **verTime0** – начало интервала;
- (239) **verTime1** – конец интервала;
- (240) **ReqQ** – запрос прогноза, посылается 0, возвращается:
 - 0 – идет вычисление;
 - 1 – норма;
 - 2 – периодическое ТО;
 - 3 – ТО по времени работы;
 - 4 – ТО по выработке;
 - 5 – ТО;
 - 6 – предел срока эксплуатации;
 - 7 – ошибка.

Одновременно устанавливаются атрибуты 236, 237 (вероятности);

- (243) **ID_SRV0** – ID ТО 1;
- (244) **ID_SRV1** – ID ТО 2;
- (245) **ID_SRV2** – ID ТО 3;
- (246) **ID_SRV3** – ID ТО 4;
- (250) **TraceTime** – общее время наблюдения за оборудованием:
 - **TraceTime = T_current – DataIn** (11), если не установлен бит 7 атрибута **Algorithm** (34);
 - **TraceTime = T_current – DataB** (116), если бит 7 атрибута **Algorithm** (34) установлен;
- (251) **InObj** – ID группы, которой принадлежит данный канал;
- (252) **Level** – иерархический уровень, на котором находится данный канал (узел находится на уровне 0). Например, канал **unit2** расположен на третьем уровне:



- (253) **Root** – служебная переменная.

В профайлере канал **ЕДИНИЦА ОБОРУДОВАНИЯ** индицируется как **C14_Passport** (атрибут 126, TsT).

Канал класса ПЕРСОНАЛ

Канал класса **ПЕРСОНАЛ** предназначен для учета работника и вычисления ряда его характеристик.

Канал **ПЕРСОНАЛ**, имя которого совпадает с именем группы, в которую он входит, выполняет функции группы пользователей (членами являются другие каналы **ПЕРСОНАЛ** группы).

Редактор канала ПЕРСОНАЛ

В редакторе канала **ПЕРСОНАЛ** задаются начальные значения некоторых его атрибутов (см. **Атрибуты канала ПЕРСОНАЛ**).

Общий раздел редактора

| | | | |
|----------------------------|------------|---------------------|------------|
| Фамилия | Ivanov | Телефон | 111-11-11 |
| Имя | Ivan | Факс | 222-22-22 |
| Отчество | Ivanovich | Е-mail | x@x.ru |
| Дата рождения | 01.01.1980 | Возраст | 24 |
| Должность | manager | Допуск | 3 |
| Квалификация | | | |
| Принят на работу | | Категория персонала | 7 |
| Дата | 01.01.2000 | Др. средства связи | ICQ6666666 |
| Тип документа | 10 | Контракт до | 01.01.2010 |
| Номер документа | 1234 | Место работы | Dep.3 |
| Имя документа | doc1 | Приоритет вызова | 2 |
| Местонахождение | | Стаж работы | time#0s |
| Непосредственный начальник | | | |

В этом разделе задаются начальные значения следующих атрибутов:

- **Фамилия** – фамилия, **(102) LastName**;
- **Имя** – имя, **(103) Name**;
- **Отчество** – отчество, **(104) MidName**;
- **Дата рождения** – дата рождения, **(106) BIRTHDAY**;
- **Должность** – должность;
- **Квалификация** – квалификация, **(010) QUALIF**;
- **Дата** – дата принятия на работу, **(011) DATA_in**;
- **Тип документа** – тип документа о принятии на работу, **(107)**

DocInType;

- **Номер документа** – номер документа о принятии на работу, (108) **DocInNum**;
- **Имя документа** – наименование документа о принятии на работу, (109) **DocInName**;
- **Местонахождение** – местонахождение, (114) **Location**;
- **Непосредственный начальник** – ссылка на канал ПЕРСОНАЛ, (072) **CHIFF**;
- **Телефон** – номер служебного телефона, (111) **Telefon**;
- **Факс** – номер факса, (112) **Fax**;
- **Email** – адрес электронной почты, (241) **EMail**;
- **Возраст** – возраст, (016) **Age**;
- **Допуск** – номер допуска, (235) **Dopusk**;
- **Категория персонала** – категория персонала, (068) **categ**;
- **Др. средства связи** – дополнительные средства связи, (242) **ICQ**;
- **Контракт до** – дата окончания контракта, (115) **Contract**;
- **Место работы** – подразделение;
- **Приоритет вызова** – приоритет для выдачи работы, (071) **PRI-OR_CALL**;
- **Стаж работы** – стаж работы, (012) **STAGE**.

Вкладки ‘Обслуживание’

На этих вкладках задаются начальные значения некоторых атрибутов работ (из диапазонов номеров **128-146**, **148-166**, **168-186** и **188-206**):

- **Тип** – тип планирования работы, (128) **0_Type**;
- **Дата исполнения последнего** – дата выполнения последней работы, (132) **0_LastData**,

- **Статус последнего** – статус последней работы, (131) **0_PSts**;
- **Время выдачи задания до старта** – разница между временем выдачи задания на работу и временем начала работы, (140) **0_GenConst**;
- **Константа сравнения** – константа сравнения, (139) **0_CmpConst**;
- **Время начала** – дата и время начала работы, (133) **0_Start**;
- **Время окончания** – дата и время окончания работы, (134) **0_Finish**;
- **Длительность** – плановая длительность работы, (135) **0_pLen**;
- **Стоимость** – плановая стоимость работы, (137) **0_pCost**.

В разделе «Ссылки» задаются ссылки ГО:

- **1** – (142) **0_Link0**;
- **2** – (143) **0_Link1**;
- **3** – (144) **0_Link2**;
- **Ресурс** – (145) **0_Humo_ID**.

Вкладка 'Дополнительно'

| Обслуживание №1 | Обслуживание №2 | Обслуживание №3 | Обслуживание №4 | Дополнительно |
|--------------------------------|-----------------|------------------------------|-----------------|---------------|
| Работы | | | | |
| Плановая общая стоимость | | 44 | | |
| Фактическая общая стоимость | | 55 | | |
| Плановая общая длительность | | 33 | | |
| Фактическая общая длительность | | 22 | | |
| Число выполненных | | 9 | | |
| Идентификатор | | 333 | | |
| Тип | | 8 | | |
| Рабочее время | | | | |
| Тип графика | | 11 | | |
| Начало | | 10:00:00 | | |
| Конец | | 14:00:00 | | |
| По обслуживанию | | | | |
| По завершению | | <input type="checkbox"/> ... | | |
| По планированию | | <input type="checkbox"/> ... | | |
| График работы | | 111 | | |
| Дополнительные данные | | extra | | |
| Маска обслуживания | | 6 | | |
| Единицы границ интервала | | Месяц | | |

На этой вкладке задаются начальные значения следующих атрибутов:

- **Плановая стоимость** – суммарная стоимость выполненных работ (план), (212) **p_Cost**;
- **Фактическая общая стоимость** – суммарная стоимость выполненных работ (факт), (213) **f_Cost**;
- **Плановая общая длительность** – суммарная длительность выполненных работ (план), (214) **pLen**;
- **Фактическая общая длительность** – суммарная длительность выполненных работ (факт), (215) **fLen**;
- **Число выполненных** – число выполненных работ/заданий,

(216) WorkCount;

- **Идентификатор** – ID, **(224) WorkID**;
- **Тип** – номер типа работы;
- **График работы** – номер графика работы;
- **Дополнительные данные** – комментарий;
- **Тип графика** – тип рабочего графика;
- **Начало** – начало рабочего дня;
- **Конец** – конец рабочего дня;
- **По завершению** – ссылка по завершению работы, **(054) AlgEnd_ID**;
- **По планированию** – ссылка по планированию работы, **(055) AlgPlan_ID**;
- **Маска обслуживания** – права на обслуживание, **(070) SRV_MASK**;
- **Единицы границ интервала** – единицы границ интервала, **(035) Period_I**.

Атрибуты канала ПЕРСОНАЛ

При описании атрибутов канала **ПЕРСОНАЛ** указаны соответствующие поля редактора (см. **Редактор канала ПЕРСОНАЛ**):

- **(000) R** – текущий статус;
- **(001) A** – не используется;
- **(002) In** – задание статуса;
- **(003-008)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(009) Q** –
- **(010) QUALIF** – квалификация (общий раздел/**Квалификация**);
- **(011) DATA_in** – дата принятия на работу (общий раздел/**Дата**);
- **(012) STAGE** – стаж работы (общий раздел/**Стаж работы**);
- **(016) Age** – возраст (общий раздел/**Возраст**),
- **(023) FactWork** – фактически отработанное время;
- **(024) FewInterW** – фактически отработанное время в текущем интервале;
- **(027) Total_KU** – коэффициент использования;
- **(028) Message** – при каждом вводе в этот атрибут текстовой строки она отправляется в виде SMS-сообщения на мобильный телефон работника. При вводе текстовой строки в данный атрибут канала-группы она отправляется в виде SMS-сообщения на мобильные телефоны членов группы. Чтение этого атрибута невоз-

можно;

- **(029) Idle** – суммарное время простоев;
- **(031) Bad** – суммарное время болезни;
- **(032) III** –
- **(035) Period_I** – единицы границ интервала (**Дополнительно/Единицы границ интервала**),
- **(038-039)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(040) Mode** –
- **(041-052)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(053) LoadAll** – флаг загрузки всех атрибутов;
- **(054) AlgEnd_ID** – ссылка по завершению работы (**Дополнительно/По завершению**),
- **(055) AlgPlan_ID** – ссылка по планированию работы (**Дополнительно/По завершению**),
- **(056-061)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(068) categ** – категория персонала (общий раздел/**Категория персонала**);
- **(070) SRV_MASK** – на обслуживание (**Дополнительно/Маска обслуживания**);
- **(071) PRIOR_CALL** – приоритет для выдачи работы (общий раздел/**Приоритет вызова**);
- **(072) CHIFF** – непосредственный начальник (общий раздел/**Непосредственный начальник**);
- **(076) Last_CLC** – время последнего пересчета канала;
- **(077) FewSTS_T** – время нахождения в текущем статусе;
- **(078-083)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(084) b11** –
- **(085) FuP** –
- **(086-098)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(101) Foto_ID** – ID фотографии;
- **(102) LastName** – фамилия (общий раздел/**Фамилия**);
- **(103) Name** – имя (общий раздел/**Имя**);
- **(104) MidName** – отчество (общий раздел/**Отчество**);
- **(105) PLC_OF** –

- **(106) BIRTHDAY** – дата рождения (общий раздел/**Дата рождения**);
- **(107) DocInType** – тип документа о принятии на работу (общий раздел/**Тип документа**);
- **(108) DocInNum** – номер документа о принятии на работу (общий раздел/**Номер документа**);
- **(109) DocInName** – наименование документа о принятии на работу (общий раздел/**Имя документа**);
- **(111) Telefon** – номер служебного телефона (общий раздел/**Телефон**);
- **(112) Fax** – номер факса (общий раздел/**Факс**);
- **(113) Data_out** – дата увольнения;
- **(114) Location** – местонахождение (общий раздел/**Местонахождение**);
- **(115) Contract** – дата окончания контракта (общий раздел/**Контракт до**);
- **(118-127)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(128) 0_Type** – тип планирования работы 1 (**Обслуживание 1/Тип**);
- **(129) 0_Sts0** – статус работы;
- **(130) 0_Sts1** – мониторинг статуса работы;
- **(131) 0_PSts** – статус последней работы (**Обслуживание 1/Статус последнего**);
- **(132) 0_LastData** – дата выполнения последней работы (**Обслуживание 1/Дата исполнения последнего**);
- **(133) 0_Start** – дата и время начала работы (**Обслуживание 1/Время начала**);
- **(134) 0_Finish** – дата и время окончания работы (**Обслуживание 1/Время окончания**);
- **(135) 0_pLen** – длительность работы (план) (**Обслуживание 1/Длительность**);
- **(136) 0_fLen** – длительность работы (факт);
- **(137) 0_pCost** – стоимость работы (план) (**Обслуживание 1/Стоимость**);
- **(138) 0_fCost** – стоимость работы (факт);
- **(139) 0_CmpConst** – константа сравнения (**Обслуживание 1/Константа сравнения**);
- **(140) 0_GenConst** – разница между временем выдачи задания на работу и временем начала работы (**Обслуживание 1/Время вы-**

- дачи задания до старта), переведенная в секунды;
- (141) **0_TimeBefore** – текущее время до начала работы;
 - (142) **0_Link0** – ссылка 1 (**Обслуживание**);
 - (143) **0_Link1** – ссылка 2 (**Обслуживание**);
 - (144) **0_Link2** – ссылка 3 (**Обслуживание**);
 - (145) **0_Humo_ID** – ссылка 4 (**Обслуживание**);
 - (146) **0_SetSts** – задание статуса работы с проверкой корректности переходов;
 - (148-166) – аналог (128-146) для работы 2;
 - (168-186) – аналог (128-146) для работы 3;
 - (188-206) – аналог (128-146) для работы 4;
 - (212) **p_Cost** – суммарная стоимость выполненных работ (план) (**Дополнительно/Плановая стоимость**);
 - (213) **f_Cost** – суммарная стоимость выполненных работ (факт) (**Дополнительно/Фактическая общая стоимость**);
 - (214) **Plen** – суммарная длительность выполненных работ (план) (**Дополнительно/Плановая общая длительность**);
 - (215) **FLen** – суммарная длительность выполненных работ (факт) (**Дополнительно/Фактическая общая длительность**);
 - (216) **WorkCount** – число выполненных работ/заданий (**Дополнительно/Число выполненных**);
 - (222) **Total_Cost** – суммарная стоимость выполненных работ (факт);
 - (223) **DayCost** – стоимость выполненных работ за текущий день;
 - (224) **?WorkID** – ID (**Дополнительно/Идентификатор**);
 - (225) **k0** – интервальный коэффициент использования 1 (верхний в стеке);
 - (226) **k1** – интервальный коэффициент использования 2;
 - (227) **k2** – интервальный коэффициент использования 3;
 - (228) **k3** – интервальный коэффициент использования 4;
 - (229) **k4** – интервальный коэффициент использования 5;
 - (230) **k5** – интервальный коэффициент использования 6;
 - (231) **k6** – интервальный коэффициент использования 7;
 - (232) **k7** – интервальный коэффициент использования 8;
 - (233) **PSTN** –
 - (235) **Dopusk** – номер допуска (общий раздел/**Допуск**);
 - (241) **EMail** – адрес электронной почты (общий раздел/**Email**);
 - (242) **ICQ** – дополнительные средства связи (общий раздел/**Др.**

средства связи);

- (243) ID_SRV0 – ID работы 1;
- (244) ID_SRV1 – ID работы 2;
- (245) ID_SRV2 – ID работы 3;
- (246) ID_SRV3 – ID работы 4;
- (250) TraceTime – общее время работы на предприятии;
- (251) InObj – ID группы, которой принадлежит данный канал;
- (252) Level – иерархический уровень, на котором находится данный канал (узел находится на уровне 0);
- (253) Root – служебная переменная.

В профайлере канал ПЕРСОНАЛ индицируется как C15_Homo (атрибут 126, TsT).

Канал класса D-РЕСУРС

Канал класса **D-РЕСУРС** предназначен для планирования работы (например, техобслуживания единицы оборудования) и мониторинга ее выполнения.

При задании параметров ТО для канала **ЕДИНИЦА ОБОРУДОВАНИЯ** или **ПЕРСОНАЛ** монитор автоматически создает для каждого ТО соответствующий канал **D-РЕСУРС**.

Редактор канала D-РЕСУРС

На вкладках редактора канала **D-РЕСУРС** задаются начальные значения некоторых его атрибутов (см. **Атрибуты канала D-РЕСУРС**).

Вкладка 'Параметры'

Параметры Сервис Ссылки

Идентификатор

Владелец X ...

Приоритет обслуживания ▲ ▼

Автоматическая выдача ресурсов

Разрешение установки статуса владельца по началу сервиса

Отмена автоматического перехода в WAIT_START

На этой вкладке задаются начальные значения следующих атрибутов:

- **Идентификатор** – ID работы, **(033) WorkID**;
- **Владелец** – привязка к каналу, **(145) 0_Link_ID0**. Если выбирается канал **ЕДИНИЦА ОБОРУДОВАНИЯ** или **ПЕРСОНАЛ**, атрибут для привязки выбирается автоматически в зависимости от типа планирования работы; для каналов других классов выбирается атрибут **0, R**. Для привязки к произвольному атрибуту следует сконфигурировать свойство **связь** канала D-РЕСУРС;
- **Приоритет обслуживания** – приоритет **TO (071)**;
- **Автоматическая выдача ресурсов** – если этот флаг установлен, то **(053) Set = 1**, и ссылки на каналы М-РЕСУРС обрабатываются автоматически при переходе **TO** в статус **WAIT_START**;
- **Разрешение установки статуса владельца по началу сервиса** – если этот флаг установлен, то **(084) Set1 = 1**, и при статусе **TO START** статус владельца автоматически переводится в **SERVICE**;
- **Отмена автоматического перехода в WAIT_START** – если этот флаг установлен, то **(085) Set2 = 1**, и переход из статуса **SUBMITTED** в **WAIT_START** должен быть выполнен вручную. Если флаг не установлен, переход выполняется автоматически.

Вкладка 'Сервис'

Параметры Сервис Ссылки

Обслуживание

Тип Ручное

Дата исполнения последнего 00.00.0000 00:00:00

Статус последнего Неизвестен

Время выдачи задания до старта 0d0h0m0s

Константа сравнения 0

Время

Начала 00.00.0000 00:00:00

Окончания 00.00.0000 00:00:00

Плановые

Длительность 0d0h0m0s

Стоимость 0

На этой вкладке задаются начальные значения следующих атрибутов:

- **Тип** – тип планирования, **(128) 0_Type**;
- **Дата исполнения последнего** – дата и время выполнения последнего ТО, **(132) 0_LastData**;
- **Статус последнего** – статус последнего ТО, **(131) 0_PSts**;
- **Время выдачи задания до старта** – разница между временем выдачи задания на работу и временем начала работы, **(140) 0_GenConst**;
- **Константа сравнения** – константа сравнения, при задании времени – в секундах, **(139) 0_CmpConst**;
- **Время начала** – время начала работы, **(133) 0_Start**;
- **Время окончания** – время окончания работы, **(134) 0_Finish**;
- **Плановая длительность** – плановая длительность работы, **(135) 0_pLen**;
- **Плановая стоимость** – плановая стоимость работы, **(137) 0_pCost**.

Вкладка 'Ссылки'

Параметры Сервис Ссылки

Обслуживание - ссылки

№1 X ...

№2 X ...

№3 X ...

Ссылки на ресурсы

№1 X ... 0

№2 X ... 0

№3 X ... 0

№4 X ... 0

В разделе «Обслуживание – ссылки» этой вкладки задаются ссылки на каналы вызова шаблонов документов или шаблонов связей с БД:

- 1 – **(142) 0_Link0**;
- 2 – **(143) 0_Link1**;
- 3 – **(144) 0_Link2**;

Ссылки 1 и 2 отработываются при достижении времени выдачи задания на

работу, ссылка 3 – по окончании работы (при статусе сервиса 6, FINISH). Если ссылка задана на канал вызова связи с БД, то при обработке ссылки в канал посылается значение 1 (обрабатывается SQL-запрос с номером 1). Если ссылка задана на канал вызова шаблона документа, то при обработке ссылки в канал посылается значение 2 (без вывода на печать).

В разделе «Ссылки на ресурсы» задаются 4 ссылки на каналы **M-РЕСУРС (243–246, ID_RES0–ID_RES4)** и количество ресурсов (**247–250, Q_R0–Q_R4**), необходимое для проведения ТО. Если запрошенное количество хотя бы одного ресурса превышает остаток на складе, сервис не переходит в состояние START. Сравнение с реальным значением производится также в случае ссылок на любые каналы, не относящиеся к T-FACTORY. В случае ссылок на каналы T_FACTORY, отличные от каналов **M-РЕСУРС**, проверка производится:

- для каналов **D-РЕСУРС** – на статус 5 (START);
- для каналов **ЕДИНИЦА ОБОРУДОВАНИЯ** – на статус WORK;
- для каналов **ПЕРСОНАЛ** – на статус WORK/IDLE.

Атрибуты канала D-РЕСУРС

При описании атрибутов канала **D-РЕСУРС** указаны соответствующие поля редактора (см. **Редактор канала D-РЕСУРС**):

- **(000) R** – текущий статус работы;
- **(001) A** – не используется;
- **(002) In** – задание статуса работы с проверкой корректности перехода между статусами:
 - 0 – CALC (планируется/вычисляется);
 - 1 – PLANNED (спланировано);
 - 2 – ACCEPTED (утверждено);
 - 3 – SUBMITTED (выдано/направлено). Переход из статуса 2, ACCEPTED в статус 3, SUBMITTED (и обработка ссылок **0_Link0** и **0_Link1**) производится автоматически в момент времени $t = 0_Start - 0_GenConst$. При попытке ручного перехода из статуса 2 в статус 3 **0_GenConst < 0**, и значение этого атрибута необходимо исправить;
 - 4 – WAIT_START (готово к выполнению). Если **(053) Set = 1**, при переходе в статус WAIT_START обрабатываются ссылки на ресурсы **ID_RES0 – ID_RES4**;
 - 5 – START (начато);
 - 6 – FINISH (закончено). При переходе в этот статус обрабатывается ссылка 3 (**0_Link2**). Из статуса FINISH сервис автоматически переходит в статус CALC;
 - 7 – WAIT (отложено), вспомогательный статус, из него воз-

возможен переход в любой другой статус;

- **(003-006)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(007) P** – этот атрибут принимает значение 1, когда до старта ТО остается менее двух дней, и 2, если ТО не переходит в статус START в заданное время;
- **(008) W** – этот атрибут имеет стандартное назначение (см. **Общие атрибуты каналов**);
- **(009) Q** – мониторинг/задание значения привязки;
- **(017) PH** – засечка значения привязки;
- **(025) All_Plen** – суммарная длительность работ (план);
- **(026) All_FLen** – суммарная длительность работ (фактически);
- **(027) All_PCost** – суммарная стоимость работ (план);
- **(028) All_FCost** – суммарная длительность работ (фактически);
- **(032) Count** – счетчик числа работ;
- **(033) WorkID** – ID работы (**Параметры/Идентификатор**);
- **(038-052)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(053) Set** – если **Set = 1**, ссылки на каналы M-РЕСУРС обрабатываются автоматически при переходе ТО в статус WAIT_START (**Параметры/Автоматическая выдача ресурсов**);
- **(056-061, 078-063)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(084) Set1** – если **Set1 = 1**, при статусе ТО START статус владельца автоматически переводится в SERVICE (**Параметры/Разрешение установки статуса владельца по началу сервиса**);
- **(085) Set2** – если **Set2 = 1**, автоматический переход из статуса SUBMITTED в WAIT_START запрещен (**Параметры/Отмена автоматического перехода в WAIT_START**);
- **(086-098, 118-120, 123-127)** – эти атрибуты имеют стандартное назначение (см. **Общие атрибуты каналов**);
- **(128) 0_Type** – тип планирования (**Сервис/Тип**) (**sts_WORK_TYPE.tmc**) (см. также **Временные диаграммы канала D-РЕСУРС** ниже):
 - 0, **Ручное** – при этом типе плановая длительность или время окончания работы вычисляются автоматически:

$$(135) 0_pLen = (134) 0_Finish - (133) 0_Start$$

$$(134) 0_Finish = (133) 0_Start + (135) 0_pLen$$
 - 1, **Периодическое** – в статусе CALC время проведения

работы вычисляется автоматически:

$$0_Start = 0_LastData + 0_CmpConst$$

$$0_Finish = 0_Start + 0_pLen$$

В других статусах время проведения работы может быть скорректировано вручную.

- 2, **По времени** – в статусе CALC время проведения ТО оборудования вычисляется автоматически:

$$0_Start = 0_LastData + 0_CmpConst * 86400 / Day_Avr_W$$

$$0_Finish = 0_Start + 0_pLen$$

В приведенной формуле **Day_Avr_W** – атрибут 26 (средне-суточное время работы в секундах) канала ЕДИНИЦА ОБОРУДОВАНИЯ, привязанного к каналу D-РЕСУРС, **86400** – число секунд в сутках. Время начала ТО не превышает времени окончания срока службы оборудования (суммы значений атрибутов **(011) DataIn** и **(012) TimeMaxW** канала ЕДИНИЦА ОБОРУДОВАНИЯ).

В других статусах время проведения работы может быть скорректировано вручную.

- 3, **По выработке** – в статусе CALC время проведения ТО вычисляется автоматически по параметрам **(014) PHNorma**, **(015) NormaPeriod** и **(027) Use_Total** привязанного канала ЕДИНИЦА ОБОРУДОВАНИЯ (см. **Атрибуты канала ЕДИНИЦА ОБОРУДОВАНИЯ**):

$$0_Start = 0_LastData + 0_CmpConst * NormaPeriod / (PHNorma * Use_Total)$$

$$0_Finish = 0_Start + 0_pLen$$

Время начала ТО не превышает времени окончания срока службы оборудования (суммы значений атрибутов **(011) DataIn** и **(012) TimeMaxW** канала ЕДИНИЦА ОБОРУДОВАНИЯ).

В других статусах время проведения работы может быть скорректировано вручную.

- 4, **TraceT** – отслеживание времени;
- 5, **ManTrace** – вручную, отслеживание константы сравнения;
- 6, **TraceT&E** – отслеживание времени;
- 7, **TraceW** – отслеживание выработки;
- 8, **TraceW&E** – отслеживание выработки.

Типы 4-8 предназначены для отслеживания значения привязки и сравнения его изменения с момента перехода в статус START с константой сравнения. Типы с буквой «E» переходят в статус

START автоматически (по времени старта), а в статус FINISH – по времени окончания или в случае, когда сравнение с константой истинно (в зависимости от того, какое событие наступило раньше). Типы без буквы «E» переходят в статус START вручную, а в статус FINISH – когда сравнение с константой истинно (время окончания работы в этом случае не анализируется). Тип **ManTrace** переходит в статус START вручную, а в статус FINISH – по времени окончания или в случае, когда сравнение с константой истинно (в зависимости от того, какое событие наступило раньше);

- **(129) 0_Sts0** – задание статуса работы без проверки корректности перехода между статусами;
- **(130) 0_Sts1** – мониторинг состояния работы (**sts_WORK_STS1.tmc**):
 - 0 – Норма
 - 1 – Норма, не спланирован
 - 2 – Норма, не утвержден
 - 3 – Норма, не выдан
 - 4 – Норма, не подтвержден
 - 5 – Норма, не начат
 - 6 – Норма, не закончен
 - 7 – Норма, закончен
 - 8 – Превышено время, не спланирован
 - 9 – Превышено время, не утвержден
 - 10 – Превышено время, не выдан
 - 11 – Превышено время, не подтвержден
 - 12 – Превышено время, не начат
 - 13 – Превышено время, не закончен
 - 14 – Превышено время, закончен
- **(131) 0_PSts** – статус последнего ТО (**Сервис/Статус последнего**) (**sts_WORK_PSTS.tmc**):
 - 0 – Неизвестен;
 - 1 – Выполнен;
 - 2 – С превышением по времени;
 - 3 – С превышением по стоимости;
 - 4 – С превышением по времени и стоимости;
 - 5 – Не начат;
 - 6 – Не закончен;
- **(132) 0_LastData** – дата и время выполнения последнего ТО (**Сервис/Дата исполнения последнего**);
- **(133) 0_Start** – время начала работы (**Сервис/Время начала**);

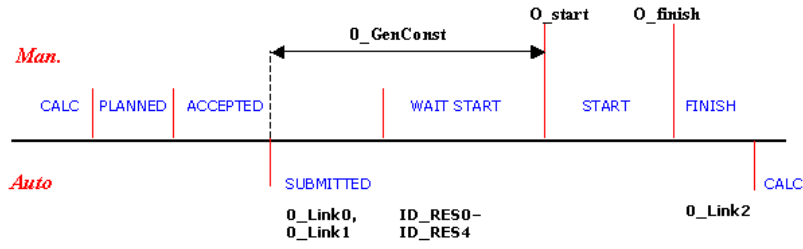
- (134) **0_Finish** – время окончания работы (**Сервис/Время окончания**);
- (135) **0_pLen** – плановая длительность работы (**Сервис/Плановая длительность**);
- (136) **0_fLen** – фактическая длительность работы;
- (137) **0_pCost** – плановая стоимость работы (**Сервис/Плановая стоимость**);
- (138) **0_fCost** – фактическая стоимость работы. При переходе ТО в статус WAIT_START в этот атрибут записывается:
 - если (053) **Set = 1** – стоимость выданных ресурсов;
 - если (053) **Set = 0** – значение (137) **0_pCost**;
- (139) **0_CmpConst** – константа сравнения (**Сервис/Константа сравнения**);
- (140) **0_GenConst** – разница между временем выдачи задания на работу и временем начала работы (**Сервис/Время выдачи задания до старта**), переведенное в секунды;
- (141) **0_TimeBef** – время до начала работы, $T_{current} - 0_Start$;
- (142) **0_Link0** – ссылка 1 (**Ссылки/Обслуживание – ссылки/1**);
- (143) **0_Link1** – ссылка 2 (**Ссылки/Обслуживание – ссылки/2**);
- (144) **0_Link2** – ссылка 3 (**Ссылки/Обслуживание – ссылки/3**);
- (145) **0_Link_ID0** – ссылка на канал (**Параметры/Владелец**);
- (146) **0_SetSts** – следующий статус работы;
- (243 – 246) **ID_RES0 – ID_RES4** – ссылки на ресурсы (**Ссылки/Ссылки на ресурсы**);
- (247 – 250) **Q_R0 – Q_R4** – количество ресурсов, необходимое для ТО (**Ссылки/Ссылки на ресурсы**).

В профайлере канал **D-Ресурс** индексируется как **C10_D-Resource** (атрибут 126, **TsT**).

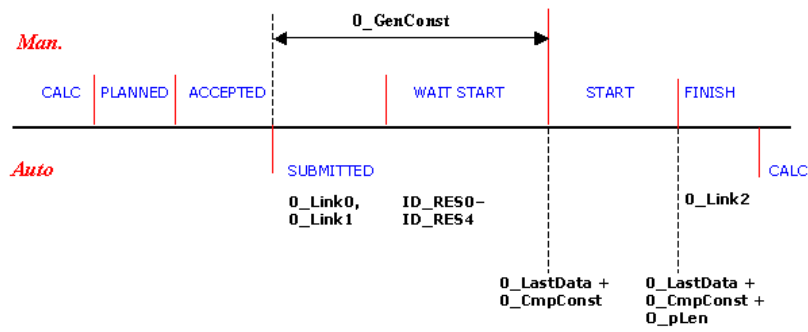
Временные диаграммы канала D-РЕСУРС

Ниже представлены временные диаграммы канала D-РЕСУРС при различных типах планирования.

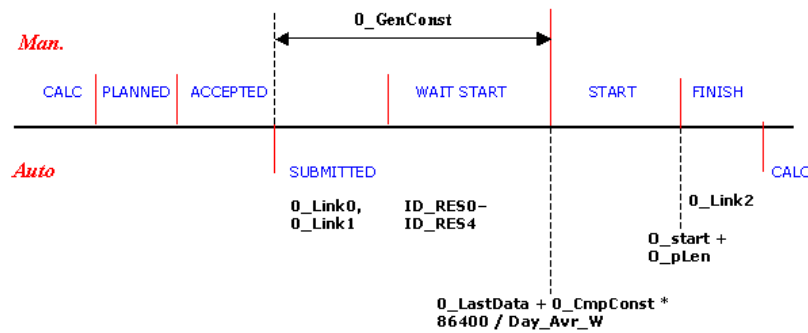
0, MANUAL



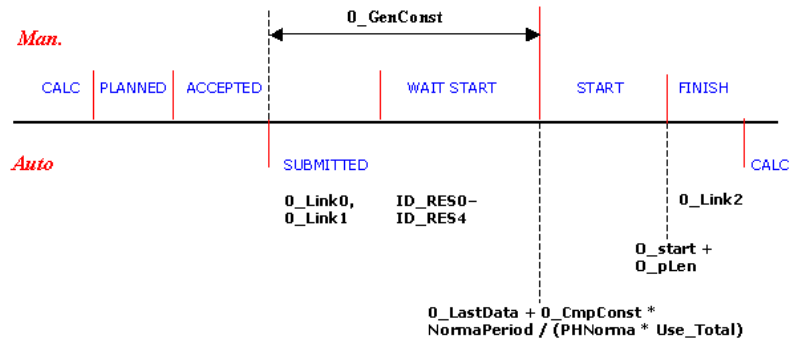
1, PERIODICAL



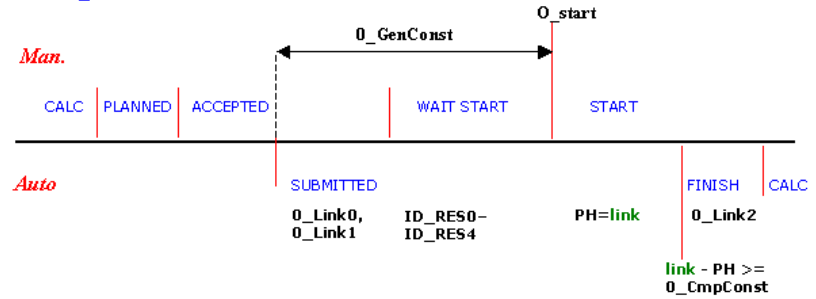
2, BY_TIME



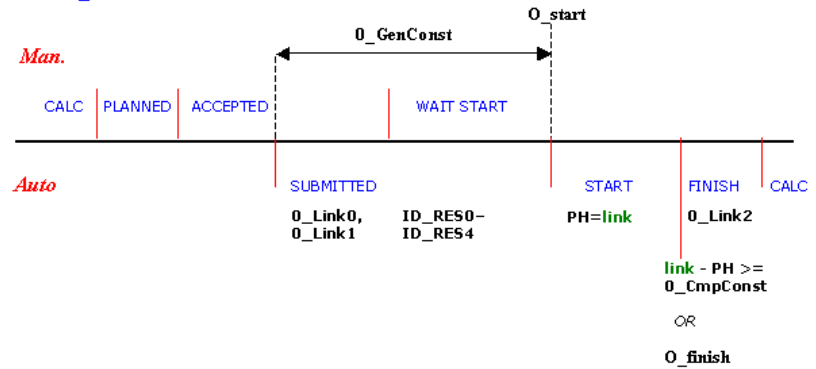
3, BY_VOLUME



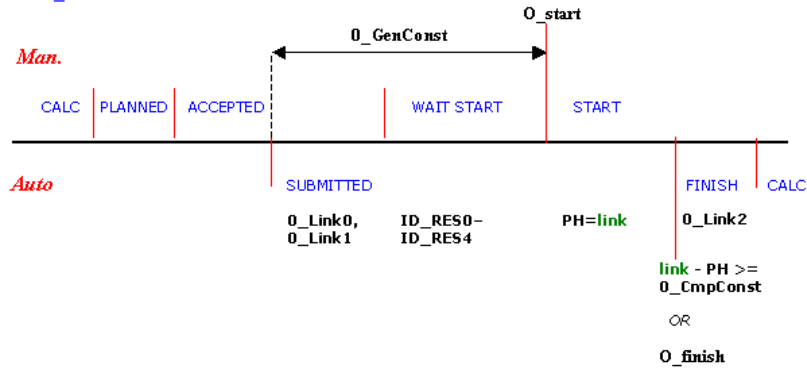
4, TRACE_T
7, TRACE_W



6, TRACE_T&E
8, TRACE_W&E



5, MAN_TRACE



Глава 13

**TRACE MODE 6 в
решении
отраслевых задач**

Электроэнергетика

АСКУЭ

Для выполнения требований, предъявляемых к автоматизированным системам коммерческого учета электроэнергии и мощности (АСКУЭ), TRACE MODE 6 располагает следующими средствами:

- В TRACE MODE встроен универсальный механизм обмена с электросчетчиками – см. **Универсальный механизм обмена с электросчетчиками**.
- Обеспечение работы распределенной системы, в том числе передача на верхний уровень необходимой коммерческой информации по используемым линиям связи – TRACE MODE не накладывает практически никаких ограничений на топологию АСУ, используемые в них аппаратные средства и линии связи (см. **Обеспечение работы распределенных АСУ**).
- Привязка технических средств к системе единого астрономического времени – для решения данной задачи предусмотрены специальные переменные TRACE MODE 6 (**@t_Set_Time** и **@t_Set_Date** – см. **Группа СИСТЕМНЫЕ**).
- Автоматическая диагностика работоспособности системы – для диагностики работоспособности системы могут быть использованы различные функции TRACE MODE 6 – например, установка каналу аппаратной/программной недостоверности (см. **Атрибуты каналов, отображаемые профайлером**) в различных ситуациях, – а также системные переменные TRACE MODE 6 (см. **Группа СИСТЕМНЫЕ** и **Группа ДИАГНОСТИКА**).
- Защита от несанкционированного доступа – для защиты от несанкционированного доступа к редактированию проекта и/или управлению АСУ для каждого узла могут быть определены пользователи и заданы их права (см. **Защита проекта** и **Канал класса ПОЛЬЗОВАТЕЛЬ**).
- Расчет балансовых показателей по поставленной/полученной электроэнергии – для учета любого вида материального ресурса (в том числе электроэнергии) в физическом и стоимостном выражении в TRACE MODE 6 предусмотрен канал класса **М-Ресурс** (см. **Канал класса М-РЕСУРС**).
- Хранение результатов баланса – для архивирования информации могут быть использованы архивы SIAD (см. **Архивирование каналов узла, Архивирование каналов проекта** и **Архивы SIAD**) или базы данных (см. **Обмен с базами данных**).
- Генерация графиков текущих/усредненных значений электроэнер-

гии/мощности – для хранения текущих/усредненных значений электроэнергии/мощности в контроллерах могут быть использованы индивидуальные архивы (см. **Индивидуальный архив**). Данные этих архивов могут быть переданы на операторскую станцию и записаны в архив SIAD (см. **Запрос удаленного индивидуального архива**). Индивидуальный архив может быть выведен в генерируемый документ (на тренд – см. **Вставка тренда**).

- Архивирование коммерческой информации в случае отказа штатных линий связи – при отказе штатных линий связи данные коммерческого учета могут быть переданы, например, по телефону и внесены в архив SIAD (см. **Канал CALL.Writer**).
- Генерация форм отчетности – TRACE MODE 6 обеспечивает генерацию отчетов произвольной формы по заданным шаблонам (см. **Использование разработанных шаблонов и Редактор шаблонов документов (отчетов)**). Кроме того, предусмотрены средства генерации специальных отчетов (см. **Отчеты АСКУЭ**);
- Защита данных при сбоях аппаратных средств – для восстановления параметров АСУ (в том числе значений атрибутов каналов T-FACTORY) в TRACE MODE 6 предусмотрено создание дампа узла (см. **Файл восстановления**).

Отчеты АСКУЭ

В TRACE MODE предусмотрены средства генерации отчетов в соответствии с Приложением 11.1.1 к Положению о порядке получения статуса субъекта оптового рынка и ведения реестров субъектов оптового рынка электрической энергии и мощности.

Перед генерацией отчетов необходимые архивные данные должны быть извлечены, обработаны и записаны в аргументы каналов CALL.ChGroupReq или CALL.AS_DATA.

Для генерации документов используются каналы CALL с типом вызова 124, **AS_DOCUMENT** (см. ниже).

Для корректировки и анализа календаря рабочих/выходных дней используются соответственно каналы CALL с типом вызова 126, **CALL.EXCEPT_DAY** (см. **Канал CALL.EXCEPT_DAY**) и каналы TIME (см. **Канал класса TIME**).

Для отправки сгенерированных документов по электронной почте используются каналы CALL с типом вызова 127, **EMAIL** (см. **Канал CALL.EMAIL**).

Канал CALL.AS_DOCUMENT

При 2, **In** > 0 канал CALL с типом вызова 124, **AS_DOCUMENT** генерирует и сохраняет в папке узла XML-документ класса 800x0 (см. Приложение-

ние 11.1.1). После генерации **In** = 0 автоматически.

Имя канала CALL.AS_DOCUMENT используется в качестве параметра **Номер АИИС** в имени файла документа и в тексте.

Аргументы канала:

- **arg0** и **arg1** – служебные;
- последующие аргументы – данные. К этим аргументам привязываются атрибуты 0, **R** или атрибуты 140-186 каналов CALL.AS_DATA (см. ниже) или CALL.ChGroupReq.

Дата и время генерации документа определяются атрибутом 59, **DR**.

Значение атрибута 1, **A** автоматически увеличивается на 1 при старте генерации документа.

Значение младшего полубайта атрибута **Параметр** (34, **FPnt**) задает тип генерируемого документа:

- 2 – тип 80020;
- 3 – тип 80030;
- 4 – тип 80040;
- 5 – тип 80050.

Значение атрибута **Комментарий** (80, **CMNT**) используется как значение элемента **<inn>** документа.

Для документа типа 80050: если **Параметр** & 0x40 = 0, значение элемента **<daytype>** документа равно 1, в противном случае – 2 (в бит 6 атрибута **Параметр** может быть передано значение 0 или 1 атрибута **Тенденция** (6, **D**) канала TIME).

Для всех типов документов: если **Параметр** & 0x10 = 1, **timezone** = 3 (**timezone** – атрибут элемента **<area>** документа).

Если в канале CALL.AS_DOCUMENT установлен флаг **Запрос времени значения**, в привязанных к его аргументам каналах CALL проверяется атрибут 120, **АСК**.

Для отправки документа по электронной почте нужно к каналу CALL.AS_DOCUMENT привязать соответствующий канал CALL.EMAIL.

Канал CALL.AS_DATA

Канал CALL с типом вызова 125, **AS_DATA** обеспечивает данными документ, генерируемый с помощью канала CALL.AS_DOCUMENT.

Если атрибут 0, **R** канала CALL.AS_DATA привязан к аргументу канала

CALL.AS_DOCUMENT, конфигурация зависит от значения **К** (1...7) младшего полубайта (0xF) атрибута **Параметр** канала CALL.AS_DATA:

- если **К** = 1 или 2, к аргументам CALL.AS_DATA должны быть привязаны каналы CALL, несущие данные для генерации документа;
- если **К** = 3...7, аргументы CALL.AS_DATA должны содержать числовые данные для генерации документа:
 - **К** = 1 – точка измерения;
 - **К** = 2 – точка поставки;
 - **К** = 3 – группа поставки;
 - **К** = 4 – переток;
 - **К** = 5 – объект измерения;
 - **К** = 6 – средство измерения;
 - **К** = 7 – схема измерения.

Если атрибут 140...186 канала CALL.AS_DATA привязан к аргументу канала CALL.AS_DOCUMENT, то **К** определяется как **<номер атрибута> – 140**.

Бит **Параметр** & 0x20 формирует статус переменной в документе.

По количеству аргументов в каналах CALL.ChGroupReq (при **К** = 1 или 2) или каналах CALL.AS_DATA (при других значения **К**) формируются временные метки для этих аргументов:

- 48-50 точек – получасовые интервалы за сутки;
- 24-25 точек – часовые интервалы;
- 288/230 точек – 5-минутные;
- 1440/1500 – 1-минутные;
- 96/100 – 15-минутные;
- 144/150 – 10-минутные.

Временные метки значений аргументов канала CALL.AS_DATA (CALL.ChGroupReq) вычисляются по атрибутам 59 и 252, если эти атрибуты отличны от 0.

С указанными временными метками данные канала CALL.AS_DATA могут быть отображены на тренде и записаны в архив. Для указания канала, который должен архивироваться, используется атрибут 41, **AB** (допускается ввод ID канала (число) или имени канала (строка)).

Приложения

Используемые сокращения

АСУ – автоматизированная система управления.

АСУ ТП – автоматизированная система управления технологическим процессом.

АСУП – автоматизированная система управления предприятием.

БД – база данных.

ГЭ – графический элемент графического экрана.

ГО – графический объект.

еГЭ – графический элемент графической панели.

ИС – интегрированная среда разработки проекта TRACE MODE 6.

ЛК – левая кнопка мыши.

МРВ – монитор реального времени, исполнительный модуль TRACE MODE.

ОТ – отчет тревог.

ПК – правая кнопка мыши.

РПД – редактор для разработки шаблонов графических экранов.

еРПД – редактор для разработки шаблонов графических панелей (модификация РПД).

СПАД, SIAD – архив TRACE MODE.

ТО – техобслуживание.

Задание параметров работы мониторов

Некоторые параметры работы мониторов могут быть заданы с помощью ключей команды запуска или с помощью файла **TMcom_<ordinal>.cnf** (создается вручную в папке узла).

Ключи команды запуска

Для запуска монитора может быть использована следующая команда:

```
<MPB> <узел> [ключ] [ключ] . . .
```

где

- **MPB** – полный путь к файлу монитора;
- **узел** – полный путь к файлу узла.

Ключи описаны в разделах **Профайлер с поддержкой графических экранов** и **Профайлер без поддержки графических экранов**.

Файл CNF

Файл **TMcom_<ordinal>.cnf** имеет текстовый формат, каждый ключ записывается в отдельной строке прописными буквами. Если в нулевой позиции строки находится точка с запятой, строка интерпретируется как комментарий. Предпоследняя строка должна содержать ключ завершения **END_OF_CNF**, последняя строка должна быть пустой:

```
<ключ1>  
;<комментарий>  
. . .  
END_OF_CNF  
<пустая строка>
```

В нижеследующем описании ключи сгруппированы в соответствии с конфигурируемыми функциями.

Ключи могут быть также заданы на вкладке **Компоненты** (см. **Панель MPB**)

Системные

- **RUN** – автоматический запуск узла. Для **rtc.exe** ключ работает при выполнении следующих условий:

- профайлер запускается из ИС;
- узел не содержит пользователей;
- узел содержит хотя бы один канал вызова экрана;
- **ANSI** – вывод сообщений в кодировке ANSI;
- **E15=<ndde>** – задание значения системной переменной **@Net_DDE** типа OUTPUT (см. **Группа СИСТЕМНЫЕ**, а также **Задание параметров узла**);
- **E18=<nlog>** – задание значения системной переменной **@Logging** типа OUTPUT (см. **Группа СИСТЕМНЫЕ**, а также **Задание параметров узла**);
- **E20=<ninout>** – задание значения системной переменной **@Input_Output** типа OUTPUT (см. **Группа СИСТЕМНЫЕ**, а также **Задание параметров узла**);
- **NIX=<nn>** – индивидуальный номер узла. В частности, этот ключ позволяет запустить один и тот же узел Console на нескольких ПК под управлением консоли с групповой лицензией (для каждого ПК нужно задавать свое значение NIX);
- **CLCLOOP=<cl>** – время цикла CALC в мс;
- **GTTNRM=<ResetWaitTime>** – время нормальной работы (в секундах), по истечении которого биты переменной **err_resource** сбрасываются; значение по умолчанию – 600с;
- **AVB_PATH=<avbPATH>** – директория сохранения файлов, генерируемых MPB, может указываться как в кавычках, так и без них;
- **IDLLOOP=<idlLoop>** – время цикла IDLE как число основных циклов (ср. **@Calc_Loop** OUTPUT с **Параметр=18**). Значение по умолчанию:
 - 1, если время основного цикла больше 1 с;
 - 1000мс/<время основного цикла, мс>, если время основного цикла меньше 1 с;
- **FSTLOOP=<fstLoop>** – время цикла FAST, мс (**@Calc_Loop** OUTPUT с **Параметр=9**);
- **RS_FAST_LOOP=<номер COM-порта>** – если этот ключ задан, каналы с типом пересчета **цикл FAST** и **FAST EXE**, использующие для обмена указанный порт, работают синхронно (таких каналов может быть не более 4);
- **TFCLOOP=<tfcLoop>** – время цикла T-Factory, мс (**@Calc_Loop** OUTPUT с **Параметр=5**);
- **ONCERUN** – запрет запуска другого MPB;
- **NOOXSTART** – запрет работы с Data Center;
- **MAXNODE=<mNode>** – максимальное число узлов в проекте (если этот ключ не задан, максимальное число узлов в проекте равно **<число узлов в addr.ind >+1**). В частности, все используе-

мые значения **NIX** должны быть меньше **MAXNODE**;

- **IPGETPRT**=<ip_get> – приоритет потока IP_RECEIVE, **ip_get** может принимать следующие значения: NORMAL, ABOVE, HIGHER, BELOW, IDLE, REALTIME;
- **IPSNDRPT**=<ip_send> – приоритет потока IP_SEND, **ip_send** может принимать следующие значения: NORMAL, ABOVE, HIGHER, BELOW, IDLE, REALTIME.
- **FRQSHF0**=<число секунд, DEC> – сдвиг для типа пересчета (25) *мин (см. **Период пересчета канала**);
- **FRQSHF1**=<число минут, DEC> – сдвиг для типа пересчета (26) *час (см. **Период пересчета канала**);
- **FRQSHF2**=<число минут, DEC> – сдвиг для типа пересчета (27) *день (см. **Период пересчета канала**). Если заданное число меньше 60, оно интерпретируется с точностью до минуты, а если больше 60 (но меньше 24*60), то с точностью до часа;
- **FRQSHF3**=<число минут, DEC> – сдвиг для типа пересчета (28) *месяц (см. **Период пересчета канала**). Если заданное число меньше 60, оно интерпретируется с точностью до минуты, если больше 60 и меньше 24*60 – с точностью до часа, если больше 24*60 – с точностью до дня;
- **FRQSHF4**=<число минут, DEC> – сдвиг для типов пересчета (23) день недели и (24) *день недели (см. **Период пересчета канала**).
- **KBD_VIRTUAL**<0..9> = <ASCII-код клавиши> – см. описание **@Key_Code** в разделе **Группа СИСТЕМНЫЕ**;
- ключи задания параметров модемов (см. **Задание параметров узла**):
 - TM_N_PHONE_NODE_<xxx>=<телефон 1>
 - TM_S_PHONE_NODE_<xxx>=<телефон 2>
 - TM_3_PHONE_NODE_<xxx>=<телефон 3>
 - TM_4_PHONE_NODE_<xxx>=<телефон 4>
 - TM_PHONE_INITH=<строка инициализации модема 1>
 - TM_PHONE_INITS=<строка инициализации модема 2>
 - TM_PHONE_INIT3=<строка инициализации модема 3>
 - TM_PHONE_INIT4=<строка инициализации модема 4>
- ключи задания параметров дополнительных модемов:

Дополнительный модем можно задать только в том случае, если задан соответствующий основной. Дополнительные модемы могут только принимать SMS.

TM_PHREC_INIT0=<строка инициализации модема 1>
 TM_PHREC_INIT1=<строка инициализации модема 2>

TM_PHREC_INIT2=<строка инициализации модема 3>
 TM_PHREC_INIT3=<строка инициализации модема 4>
 TM_PHREC_PNUM0=<телефон 1>
 TM_PHREC_PNUM1=<телефон 2>
 TM_PHREC_PNUM2=<телефон 3>
 TM_PHREC_PNUM3=<телефон 4>

- **SOUND_MAX_PLAY**=<число секунд> – максимальная длительность звучания;
- **TM6_MSTRSRV_SUPPORT**= – при ненулевом значении ключа – поддержка терминального сервера Windows;
- **TM_BYFRQ**<m> – для типа пересчета (13) **по времени** (см. **Период пересчета канала**);
- **CH_IN_FAST_RS**=<номер канала> – перевод канала в FAST RS (см. **Период пересчета канала**);
- **CALL_MAX_RESIZE**=<число> – ограничение числа аргументов каналов CALL при их создании в реальном времени;
- **EEMLULO**= и **PEMLULO**= – для отладки проекта Windows CE на обычных средствах. Второй ключ задает число значений, которые нужно считать (в качестве значений используются текущие секунды);
- **STRDD**= – только для разработчиков TRACE MODE;
- **NXRDD**= – только для разработчиков TRACE MODE;
- **EXP_NAME**=<имя файла> – см. **Канал класса СОБЫТИЕ**;
- **WDOG_ENABLE**=<период в секундах> – задание работы сторожевого таймера;
- **WDOG_ENABLE**=OFF – не использовать сторожевой таймер;
- **WDOG_WAIT**=ON – задержка старта сторожевого таймера (таймер ожидает запуска узла);
- **TOFFT_NODE**<номер узла>=<число минут> – компенсация разницы во времени для узлов, находящихся в других часовых поясах.

Отладка

- **DEBUG**=, **DEBUGON**=, **DEBUGOFF**= – эти ключи задают значение переменной 14.14 **@Debug** с **Параметр**=0 (14.14.0, см. **Группа СИСТЕМНЫЕ**). Ключ **DEBUG**= обеспечивает установку и сброс битов, **DEBUGON**= – только установку, **DEBUGOFF**= – только сброс. Значение ключа задается в формате HEX;
- **ERROR_SHOWON**=<K> – в зависимости от значения **K**, данный ключ задает следующие режимы:
 - 10 – вывод в протокол MPB информации о перепривязке аргументов CALL;

- 20 – вывод в протокол MPB информации об изменении числа аргументов CALL;
- 40 – вывод в файл реальных значений;
- 100 – вывод в протокол MPB дополнительной информации о некоторых E_METER;
- 200 – вывод в протокол MPB значений, передаваемых в экран (однократно);
- 400 – разрешение отображения дополнительных атрибутов;
- 1000 – вывод в протокол MPB счетчиков потоков (однократно);
- 4000 – вывод в протокол MPB дополнительной информации о некоторых потоках, не показываемых по команде **@Debug=0x4000**;
- 8000 – вывод в протокол MPB дополнительной информации;
- 10000 – запрет отсылки сообщений по сети;
- 20000 – запрет ведения буфера сообщений для NetLink Light и документа;
- 40000 – запрет ведения буфера сообщений для графики.

Данные режимы могут быть заданы также с помощью команд **errlock=K** или **errlockon=K** на вкладке **Компоненты** панели MPB (см. **Панель MPB**);

- **ERRSHOW=**, **ERROR_SHOW=**, **ERRSHOWON=**, **ERRSHOWOFF=** – эти ключи задают значение переменной 14.14 **@Debug** с **Параметр=1** (14.14.1, см. **Группа СИСТЕМНЫЕ**). Ключ **ERRSHOW=** или **ERROR_SHOW=** обеспечивает установку и сброс битов, **ERRSHOWON=** – только установку, **ERRSHOWOFF=** – только сброс;
- **DBG_INFO_NET=ON** – то же, что установка бита 2 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_INFO_NET=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_NET=OFF**;
- **DBG_INFO_NODE=ON** – то же, что установка бита 9 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_INFO_NODE=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_NODE=OFF**;
- **DBG_INFO_SNMP=ON/OFF** – см. **@Debug** с **Параметр=1**;
- **DBG_INFO_TCP=ON/OFF** – то же, что установка бита 16 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_INFO_TCP=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_TCP = OFF**.
- **DBG_INFO_RS=ON/OFF** – см. **@Debug** с **Параметр=0**;
- **DBG_INFO_MARKER=ON** – то же, что установка бита 7 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также

Сообщения при **DBG_INFO_MARKER=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_MARKER = OFF**.

- **DBG_INFO_SLAVE=ON/OFF** – то же, что установка бита 27 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_INFO_SLAVE=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_SLAVE = OFF**.
- **DBG_PEVE_TCP=ON/OFF** – то же, что установка бита 28 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_PEVE_TCP=ON**). Для отмены генерации сообщений используется ключ **DBG_PEVE_TCP = OFF**.
- **DBG_INFO_IEC104=ON/OFF** – то же, что установка бита 25 переменной **ERRSHOW** (14.14 **@Debug** с **Параметр=1**; см. также **Сообщения при DBG_INFO_IEC104=ON**). Для отмены генерации сообщений используется ключ **DBG_INFO_IEC104 = OFF**.
- **SYNC_MARKER=ON/OFF** – см. **Сообщения при SYNC_MARKER=ON**.
- **DBG_INFO_LOGGER=ON/OFF** – см. **Сообщения при DBG_INFO_LOGGER=ON**.

Параметры переменной **stress**:

- **STRESS_VAL=**
- **STRESS_CLC=**
- **STRESS_NET=**
- **STRESS_THC=**

Данные ключи описаны в разделе **Группа СИСТЕМНЫЕ** (см. **@Debug** с **Параметр=3**).

Отчет тревог

- **CEVENTSTS<n>=<строка без кавычек и пробелов>** – задание сообщения OT с номером **n** по каналу СОБЫТИЕ.

Следующие ключи описаны в разделе **Формат строки OT**:

- **ALR_FRMT_MSM=1 .. 5**
- **ALR_FRMT_MSM=ON/OFF**
- **ALR_FRMT_MSM=STD**
- **ALR_FRMT_MSM=BOTH**
- **ALR_HEX_ADDVALUE=ON**
- **ALR_FLOAT_ADDVALUE=ON**

- **ALR_DEC_ADDVALUE=ON**
- **ALR_ADDERROR=ON/OFF:**
- **ALR_DIMENSION=ON**
- **ALR_CEV_TWOTIME=ON**
- **ALR_ORDER_BY=DEF**
- **ALR_ORDER_BY=TIME**

- **ALANET_REQ_START=ON/OFF** – в консолях: запрос ОТ у удаленных МРВ разрешен/запрещен;
- **ALANET_RESP_START=ON/OFF** – разрешение отправки ОТ на запрос консоли на старте;
 - **ALANET_SEND_WORK=ON/OFF** – разрешение/запрет отсылки в режиме WORK;
 - **ALANET_SEND_TRACE=ON/OFF** – разрешение/запрет отсылки в режиме TRACE;
- **ALANET_REC_WORK=ON/OFF** – в консолях: прием ОТ от удаленных МРВ разрешен/запрещен;
- **ALANET_SEND_TONODE=<порядковый номер узла>** – таких строк в файле *.cnf может быть не более 4-х. Консоль запрашивает только у таких узлов, и МРВ рассылает только заданным;
- **ALA_DISABLE_CASH=ON** – отключение списка для CALL.DOCUMENT, и такой узел не сможет отсылать на запрос на старте;
- **ALA_DISABLE_GRAPH=ON** – отключение списка для CALL.SCREEN и CALL.PANEL;
- **ALA_DISABLE_SEND=ON** – отключение списка для отсылки по направлениям **GSM, PRN** и всем направлениям воспроизведения файла (**Play, @Sound_File** и т.п.);
- **ALA_DISABLE_NET=ON** – запрет отправки сообщений ОТ в сеть;
- **ALANET_SEND_ALL=ON** – разрешение отправки всех сообщений ОТ в сеть;
- **ALA_DISABLE_FILE=ON** – отключение списка для записи в файл;
- **ALR_FRMT_MSM=ON/OFF** – первый ключ включает, а второй выключает режим, в котором в поле **Coding** ОТ вместо кодировки записывается время изменения (значение атрибута 45, **T**);
- **ALR_CEV_TWOTIME=ON** – при записи сообщений по каналам **Событие** в ОТ записывается время возникновения и исчезновения;
- **ALR_HEX_ADDVALUE=ON** – к стандартному сообщению ОТ добавляется значение канала в формате HEX;
- **ALR_DEC_ADDVALUE=ON** – к стандартному сообщению ОТ

добавляется значение канала в формате DEC;

- **ALR_FLOAT_ADDVALUE=ON** – к стандартному сообщению ОТ добавляется значение канала в формате FLOAT;
- **ALR_DIMENSION=ON** – к стандартному сообщению ОТ добавляется размерность канала;
- **ALR_FLOAT_ADDLIMMES=ON** – этот ключ нужно задать для генерации сообщений ОТ по каналам FLOAT без границ, но со словарем;
- **ALR_ADDERROR=ON(по умолчанию)/OFF** – если канал недостоверен, то в поле категории сообщения выводится/не выводится знак вопроса;

Сеть

- **IPP_CHTOFA=<число циклов>** – для запросов по сети. Если в течение заданного числа циклов ответ не получен, в канале устанавливается признак аппаратной недостоверности;
 - **TCP_SNDTMO=<число секунд>** – таймаут посылки через сокет подключения. Сокет возвращает ошибку, если в течение заданного времени не смог ничего отправить;
 - **TCP_SYNC_R=<число>** – ограничение числа блоков через сокет подключения (у сокета есть буфер, длина которого задается, но некоторые ОС (Windows CE, например) не умеют работать с длиной буфера);
 - **TCP_SYNC_W=<число мс>** – если сокет отправки занят, попытка отправки будет произведена спустя заданное время. Число таких попыток задает ключ **TCP_WHEN_B**;
 - **TCP_WHEN_B=<число попыток>** – см. выше **TCP_SYNC_W**;
 - **USEBRIDGE=<номер узла>** – указанный узел будет использоваться в качестве моста;
 - **TM_IPS_FIND_AD0=<IP адрес>** – обмен между узлами будет производиться через адаптер с IP-адресом, заданным этим ключом;
 - **TM_IPS_FIND_AD1=** – зарезервировано;
 - **TM_IPS_FIND_AD2=** – зарезервировано;
 - **TM_IPS_FIND_AD3=** – зарезервировано;
 - **TM_TCP_BIND_CLI=<IP-адрес>** – IP-адрес сокета клиента;
 - **TM_TCP_BIND_SRV=<IP-адрес>** – IP-адрес сокета прослушивания, к которому будут подключаться другие узлы;
 - **UNITV_REGIM=<◇>** – см. Канал **CALL.Vector**;
 - **NET_SENDC=<число>** – маска адаптеров посылки (ИС);
 - **NET_RECVC=<число>** – маска адаптеров приема (ИС).
- **NET_TCPCARDS=<число HEX>**:

- биты 0-3 – адаптер прослушивания сети по TCP;
- биты 4-7 – адаптер для подключения по TCP.

Расшифровка значений:

- 0 – адаптер выбирается в зависимости от маски UDP. Если в UDP используется по приему один адрес, то берется он, иначе – **INADDR_ANY** (т.е. автоматически выбирается ОС);
- 1, 2, 3 – используется соответственно 1-й, 2-й или 3-й найденный адрес на компьютере;
- 0xe – если на старте задан **tcp_addr** (см. ниже), то используется он, иначе – **INADDR_ANY** (т.е. автоматически выбирается ОС);
- 0xf – **INADDR_ANY** (автоматически выбирается ОС).

Внутри конструкции **TM_NODE_SET .. TM_NODE_END** могут располагаться ключи, задающие параметры указанного узла:

- **TM_NODE_SET:<ordinal number>** – ключ-начало конструкции; задает номер узла, для которого действуют внутренние ключи конструкции:
 - **ADD_NODE_INDEX** – добавить узел;
 - **LOGGER** – узел является регистратором;
 - **ENB_NO_RESP_REQ_CON** –
 - **ALLNETTCP** – не отправлять узлу маркеры по UDP;
 - **TNOND=<число секунд, DEC>** – если в течение заданного времени с узлом не было обмена, узел считается отсутствующим;
 - **LINK_BY=NET, 0 .. 7** – интерфейс обмена с узлом (**NET** – сеть);
 - **UDP_ADDR=<IP-адрес>** – IP-адрес узла для обмена по UDP;
 - **TCP_ADDR=<IP-адрес> [STATIC]** – IP-адрес узла для обмена по TCP; необязательный параметр **STATIC** запрещает автоматическое изменение адреса;
 - **ONLY_TCP=ON** – для узла разрешен обмен только по TCP;
 - **NOBROADCAST=REC** – узел не принимает широковещательные сообщения;
 - **NOBROADCAST** – узел не посылает широковещательные сообщения;
 - **ALTERNATE_TCP_PORT** – если этот ключ задан, для обмена по TCP узел использует альтернативный порт с номером (1027+<номер узла>);
 - **TIME_OFFSET=**

- **RS1TO=COM**<номер порта> – для обмена использовать этот COM-порт;
- **RS2TO=COM**<номер порта> – для обмена использовать этот COM-порт;
- **TM_NODE_END** – ключ-конец конструкции.

TCP/IP

- **TCP_RECONT**=<tcp_t1> – задержка следующей попытки соединения по TCP (в секундах); по умолчанию – 20с;
- **TCP_DICONT**=<tcp_t2> – задержка разрыва соединения по TCP в случае отсутствия данных для обмена (в секундах); по умолчанию – 60с;
- **IPLSLP**=<ip_sleep> – принудительная задержка между отправкой IP-пакетов (в миллисекундах, по умолчанию – 0);
- **TCP_MAX_LBUF**=<число IP-пакетов> – размер буфера сокета прослушивания сети;
- **TCP_MAX_CBUF**=<число IP-пакетов> – размер буфера сокета передачи;
- **IP_MAX_INBUF**=<число IP-пакетов> – максимальное число пакетов в буфере приема;
- **IP_MAX_OUTBUF**=<число IP-пакетов> – максимальное число пакетов в буфере передачи;
- **QUEUE**=<IP_send_q> – максимальный размер очереди на отправку по IP (число пакетов, по умолчанию – 1024) (@q_IP_Send_Q с Параметр=1);
- **TCPCARDS**=<число HEX> – задание переменной **tcpcards** (задается также в редакторе узла). Байт 0 задает адаптер для прослушивания сети, байт 1 – адаптер для соединения. Установленные в 1 биты байтов соответствуют следующим адаптерам:
 - бит 0 – системный;
 - бит 1 – адаптер 1;
 - бит 2 – адаптер 2;
 - бит 3 – адаптер 3;
- **SENDTIME**=<st> – период посылки сообщения о своем присутствии в сети (@RTM_Parameter с Параметр=1);
- **IPADDR_BLOCK**, **IPMASK_BLOCK**, **IPADDR_ENBL** – для обмена по сети, GPRS и SLAVE-протоколов.

Если к данному узлу пытается подключиться удаленный узел с таким IP-адресом, что <IP адрес удаленного узла> * **IPMASK_BLOCK** = **IPADDR_BLOCK**, соединение разрывается (блокировка обмена).

Перед проверкой на блокировку обмена проверяется <IP адрес уда-

ленного узла> * IPMASK_BLOCK = IPADDR_ENBL;

- IPTCP_DIR=, IPTCP_DIR_ON= и IPTCP_DIR_OFF= – конфигурирование IpTcpRedirect (см. описание переменной 14.21 @IP_parameter с Параметр=8). Ключ IPTCP_DIR= обеспечивает установку и сброс битов, IPTCP_DIR_ON= – только установку, IPTCP_DIR_OFF= – только сброс.

Механизм предотвращения потерь при обмене по сети

- LOG_SND_QSIZE
- LOG_STORY_TIME

Эти ключи описаны в разделе **Механизм предотвращения потерь при обмене по сети**.

Обмен по встроенным протоколам

- DISABLEUNITTCP;
- IEC_KFROM=<число>;
- IEC_WFROM=<число>;
- OEMTUSE= <>;
- SLAVE_UDPPORT=;
- SLAVE_UDPADDR=;
- SLVPCOM=
- HOSTTCP=
- SETCARDHOST=1 и SETCARDHOST=2
- TCP_DIFCONN<nn>=<число секунд>
- TCP_DISCONN<nn>=<число секунд>

Эти ключи описаны в разделе **Управление и диагностика обмена**.

Следующие ключи описаны в разделе **Обмен по встроенным протоколам по сети**:

- UNIT
- PORT
- TIMEOUT
- CSC_ON, CSC_OFF
- IDNT_OFF, IDNT_ON
- MDBRTU_ON, MDBRTU_OFF
- AUTO_ON
- PING_ON
- FAST_ON
- TH<IND>
- THONE

- TCP=<IP-адрес>
- TCP2=<IP-адрес>
- IP=<IP-адрес>
- IPADDR= <IP-адрес>
- IP2=<IP-адрес>
- IPADDR2=<IP-адрес>
- CLOSE_BY_SEND_ERR_ON –
- FRMT=REGSIZE=4
- FRMT=TIME_1, FRMT=TIME_2
- FRMT=DEFAULT
- FRMT=INTEL
- FRMT=SWAP0
- FRMT=SWAP3
 - FRMT=SWAP_FLOAT
 - FRMT=SWAP_LONG
 - FRMT=SWAP_TIME
 - FRMT=SWAP_DOUBLE
 - FRMT=SWAP_FLOAT0, FRMT=SWAP_FLOAT1,
FRMT=SWAP_FLOAT2 и FRMT=SWAP_FLOAT3
 - FRMT=SWAP_LONG0, FRMT=SWAP_LONG1,
FRMT=SWAP_LONG2 и FRMT=SWAP_LONG3
 - FRMT=SWAP_TIME0, FRMT=SWAP_TIME1,
FRMT=SWAP_TIME2 и FRMT=SWAP_TIME3
 - FRMT=SWAP_DOUBLE0, FRMT=SWAP_DOUBLE1,
FRMT=SWAP_DOUBLE2 и FRMT=SWAP_DOUBLE3

MODBUS TCP/IP

- **NOMDBIDNT** – отключение проверки идентификатора транзакции при обмене по MODBUS TCP/IP.
- **MDB_SLAVE_SWAP_FLOAT**

MODBUS RTU

Следующие ключи описаны в разделе **Обмен по MODBUS**:

- **MDB_FLOATCNV**
- **MDB_FRMT_ALL**
- **MDB_FRMT_FLOAT**
- **MDB_FRMT_REGSIZE**
- **MDB_FRMT_WORD**
- **MDB_FRMT_TIME**

- **MDB_FRMT_LEN_TIME**
- **MDB_FRMT_SWAP**

SIAD

- **POOL1=<p1>**, **POOL2=<p2>**, **POOL3=<p3>** – размеры пулов соответственно SIAD1-3, в блоках по 8 MB;
- **POOL5=<p5>**, **POOL6=<p6>**, **POOL7=<p7>** – то же для первых копий SIAD1-3;
- **POOL9=<p9>**, **POOL10=<p10>**, **POOL11=<p11>** – то же для вторых копий SIAD1-3;
- **CSIAD=<cs>** – период проверки архива в секундах (≥ 10);
- **TSIAD=<ts>** – расстояние между очередями архива в секундах (≥ 60);
- **QSIAD=<qs>** – число очередей архива (≤ 1024);
- **ANS_ZERO=<blockToSiad>** – **@RTM_Parameter** с Параметр=191;
- **ANSZERO** – то же, что **ANS_ZERO**;
- **SIADWRMT<номер SIAD> = <число секунд, DEC>** – для принудительной записи в архив с помощью канала 14.7 **@Data_from_SIAD**:
 - вычисляется T_0 как текущее время, которое выровнено влево с точностью, зависящей от заданного числа секунд;
 - вычисляется $T_1 = T_0 - \text{<заданное число секунд>}$;
 - если время изменения канала меньше T_1 , его значение пишется в архив с меткой времени T_0 ;
- **TVC_RSZ=<n>** – см. **Выборка и обработка данных SIAD**;
- **SIAD_PACK<n>** – очистка SIAD с номером $n=1..3$ при старте.
- **SIAD_CRSDFOR=<число>** – если архив пуст (например, создается), с помощью данного ключа он заполняется данными. Число определяет способ заполнения.
- **HNRDSIAD=OFF** – отмена поиска первого архивного значения левее запрошенного T_FROM для передачи в документ (см. **Вставка тренда**);
- **HNRSSIAD=OFF** (по умолчанию)/**ON** – **OFF**, если графический экран сам присылает дополнительный запрос об архивном значении левее запрошенного T_FROM . Если **ON**, такое значение ищет МРВ;
- **ARCH_WRITE_DEBUG=ON** – если этот ключ задан, все сообщения, направляемые в архив, записываются также и в протокол;
- **ARCH_DEBUG=OFF/ON** – вывод дополнительной информации об архивах в протокол запрещен/разрешен;

Перенаправление архива в базу данных

Перенаправление архива в БД конфигурируется с помощью следующих ключей в (см. **Перенаправление архива в базу данных**):

- **SQL_CHANNEL_NUMBER;**
- **SQL_NSIAD_NUMBER;**
- **SQL_QUEUE_SIZE;**
- **SQL_AGE_PERIOD;**
- **SQL_COMMENT;**
- **SQL_TIME_DISCONNECT;**
- **SQL_TRACE_WORK;**
- **SQL_DELETE_BEFORE;**
- **SQL_WORK_FLAG;**
- **SQL_FLUSH_PERIOD;**
- **SQL_FLUSH_COUNT.**

GSM, SMS, GPRS

- **GSM_LOG**=<число HEX> – **@RTM_Parameter** с **Параметр**=183;
- **GPRS_USE_PORT** – возможны следующие форматы данного ключа:
 - **GPRS_USE_PORT=GPRS** – GPRS-обмен через предопределенный порт;
 - **GPRS_USE_PORT=STD** – GPRS-обмен через стандартный порт;
 - **GPRS_USE_PORT**=<номер порта> – GPRS-обмен через указанный порт;
- **GPRS_BIND_ADDR**=<IP-адрес> – если этот ключ не задан, сервер анализирует данные, приходящие со всех IP-адресов через заданный GPRS-порт, в противном случае – только данные, приходящие с указанного IP-адреса;
- **RS_GOAL_IPADDR_EMULATE**=<IP-адрес> – IP-адрес, через который будут эмулироваться SMS.

Следующие ключи описаны в разделе **Пользовательские SMS**:

- **SMS_DLS**=<число> (**@RTM_Parameter** с **Параметр**=179);
- **SMS_TOR**=<число> (**@RTM_Parameter** с **Параметр**=180);
- **SMS_TCR**=<число> (**@RTM_Parameter** с **Параметр**=181);
- **SMS_RCN**=<> (**@RTM_Parameter** с **Параметр**=182);
- **SMS_FIND_DELIMITER**=ON (по умолчанию)/OFF;
- **SMS_DELIMITER_IS**=<один символ>;
- **SMS_ALL_INCOMING**=ON (по умолчанию)/OFF;

- **SMS_WRITE_INCOMMING=ON** (по умолчанию)/**OFF**;
- **SMS_READ_INCOMMING=ON** (по умолчанию)/**OFF**;
- **SMS_ACK=ON** (по умолчанию)/**OFF**;
- **SMS_ALARME_PHONE_LIST=<phone>;..<phone>;**;
- **SMS_CHECK_PHONE_LIST=<phone>; <phone>;**;
- **SMS_CHECK_PHONE_LIST_ADD_USER=ON** (по умолчанию)/**OFF**.

ODBC

- **SQLMANY** – отмена режима выполнения только одного SQL-запроса одновременно (режим по умолчанию).
- **SQLONCE** – задание режима выполнения только одного SQL-запроса одновременно.

RS

- **RS_FROM_TO_CHG=<число HEX>** – ключ для перехода с COM-порта, заданного байтом 0, на порт, заданный байтом 1;
- **RS_GOAL_EMULATE=<0 или 1>** – **@RS_on_off** с Параметр=15;
- **RS_OEM_COM_NUMBER=<номер COM-порта>** – этот ключ открывает указанный COM-порт (только для XPAC, назначение порта – OEM);
- **RSDAPI=<битовая маска>** – каждый бит соответствует RS. При работе с некоторыми виртуальными COM-портами в ряде случаев (например, при задании буфера) возникает ошибка. Если соответствующий бит установить, ошибка будет игнорироваться (однако работоспособность при этом не гарантируется!);
- **HNRDSLOW=OFF** и **HNRDRESP=OFF** – оба ключа по умолчанию имеют значение ON. Если в ответе по RS пришло 0 байт (**HNRDSLOW**) или неверное число байт (**HNRDRESP**), то:
 - если OFF – в каналах сбрасывается флаг ОТРАБОТАТЬ (и устанавливается признак аппаратной недостоверности);
 - если ON – в каналах сбрасывается флаг ОТРАБОТАТЬ в том случае, если в этих каналах признак аппаратной недостоверности был установлен ранее (в противном случае МРВ посылает запрос еще раз).
- **RS1TO_NODE_<номер узла>=<номер RS>** – первый COM-порт, через который можно обмениваться с узлом;
- **RS2TO_NODE_<номер узла>=<номер RS>** – второй COM-порт, через который можно обмениваться с узлом;
- следующие ключи описаны в разделе **Редактор параметров COM-порта**:

- **COM_SPEED_I00**=<baud rate>
- **COM_SPEED_I12**=<baud rate>
- **COM_SPEED_I13**=<baud rate>
- **COM_SPEED_I14**=<baud rate>
- **COM_SPEED_I15**=<baud rate>

T-FACTORY

- **TF_MONCALC**=<день месяца> – день месяца, в который производятся вычисления параметров T-FACTORY при изменении месяца (по умолчанию – 1).

Программы

- **VLVAIMP**=<impT> – константа **K** для расчета длины управляющего импульса в FBD-блоке **ZDV**, с;
- **VLVADDM**=<число секунд> – заданное время добавляется ко времени хода в блоке **ZDV**;
- **KLPADDM**=<число секунд> – заданное время добавляется ко времени хода в блоке **KLP**;
- **PIDLIM**<nn>=<число FLOAT> – константы **MAX_n** для FBD-блока **PID2** (<nn>= «00», «01»...«15»);
- **PIDDZN**<nn>=<число FLOAT> – коэффициенты **k_n** для вычисления зоны нечувствительности FBD-блока **PID2** (<nn>= «00», «01»...«15»);
- **IREGPR1** и **IREGPR2** – константы FBD-блока **IREG** (см. Раздел 'Управление');
- **IREGPR0** – зарезервировано;
- **IREGAG0**, **IREGAG1** и **IREGAG2** – переопределение алгоритма FBD-блока **IREG** (см. Раздел 'Управление');
- **VLVSTSM**=<число> – для задания режима FBD-блока **ZDV** (см. Раздел 'Управление');
- **KLPSTSM**=<число> – для задания режима FBD-блока **KLP** (см. Раздел 'Управление');
- **MDLSTSM**=<число> – для задания режима моделей задвижки и клапана (см. Группа 'Модели');
- **REGSTSM**= – зарезервировано;
- **PRISTSM**= – зарезервировано;
- **OTHSTSM**=<число> – для задания режима FBD-блока **SBRK** (см. Раздел 'Управление');
- **PMPSTSM**=<число> – для задания режима FBD-блока **MOTOR** (см. Раздел 'Управление').

Графика

- **MEMTGRP=<AR_Lines>** – управление числом строк ОТ, считываемых в графику:
 - AR_Lines=0 – число считываемых строк соответствует длине ОТ;
 - AR_Lines=1 – число считываемых строк соответствует размеру буфера чтения ОТ;
 - AR_Lines=2 – <длина ОТ>/4;
 - AR_Lines=3 – 256;
 - AR_Lines=4 – 1024;
 - AR_Lines=5 – 4096;
 - AR_Lines=6 – 65535;
- **GTHSPRE=<cMode>** – режим выполнения функций управления в точке перекрытия группы ГЭ, не содержащей оконных ГЭ:
 - GTHSPRE=0 – выполняются функции всех видимых ГЭ, начиная с верхнего (значение по умолчанию);
 - GTHSPRE=1 – выполняются функции только верхнего видимого ГЭ;
- **GRHLOOP=<gPeriod>** – период обновления графики в мс;
- **SCR_POPUP=<число>** – см. **Особенности вызова графического экрана**.
- **GRAPH_OPENGL=MEMBUF** – см. **Общие специфические атрибуты объемных ГЭ**.

Окна MPB

Ключи управления отображением окон MPB (см. **Профайлер с поддержкой графических экранов**), настройка протоколов и панели MPB:

- **GRAPH_STATUS_WND=OFF/ON** – окна **Системные_сообщения_1** и **Системные_сообщения_2**;
- **GRAPH_POPUP_TREE=OFF(по умолчанию)/ON** – дерево всплывающих экранов;
- **GRAPH_POPUP_MENU=OFF/ON(по умолчанию)** – меню экранов;
- **GRAPH_POPUP_ACTION=ON(по умолчанию)/OFF** –
- **GRAPH_STATIC_ACTION=ON(по умолчанию)/OFF** –
- **GRAPH_STATUS_DISABLE=LOG** – бит 0 переменной 14.14.2;
- **GRAPH_STATUS_DISABLE=PROF** – бит 1 переменной 14.14.2;
- **GRAPH_STATUS_DISABLE=** – все биты переменной 14.14.2;
- **EMB_FNTSIZE=<число>** – при старте заданная величина добав-

ляется к высоте шрифтов, загружаемых в панель;

- **ECUUEMB**=<число> – максимальное число динамических элементов панели, которые могут обновляться одновременно (1000 по умолчанию).

Резервирование

- **NET_SWTNTIME**=<aT> – таймаут переключения на другой сетевой адаптер в случае отсутствия обмена, с. Работает в случае задания несимметричной передачи (прием по двум адаптерам, отсылка – по одному). Если в течение заданного времени по используемому адаптеру обмена нет, а на другом адаптере зафиксирована некоторая активность, MPB переключается на другой адаптер;
- **RS_SWTI_TRACE**=<число> – OUTPUT **@RS_on_off** с **Параметр=6**;
- **SYNC_ECOUNTER_RS_COM**=DISABLE.ENABLE – в состоянии TRACE отключить/подключить RS с назначением **E_Meter** (см. **Редактор параметров COM-порта**);
- **SYNC_DUAL_RS_COM**=DISABLE/ENABLE – в состоянии TRACE отключить/подключить RS с назначением **DualRS**;
- **DISABLE_IN_TRACE**=<число HEX> – переменная **ModeMask-Trace** (переменная 15.16 **@Redundant** с **Параметр=20**).

Следующие ключи описаны в разделе **Синхронизация резервов**:

- **SYNC_MASK**
- **SYNC_BY_IP** и **SYNC_BY_TCP**
- **SYNC_ADJUST**
- **SYNC_DEBUG**
- **SYNC_EMUL_ERR**
- **SYNC_ARCH**
- **SYNC_HISTORY**
- **SYNC_SLOW_WORK**
- **SYNC_FAST_START**
- **SYNC_DISABLE_SYNC**
- **SYNC_DISABLE_OTHER_THEN**
- **SYNC_ENABLE_OTHER_THEN**
- **SYNC_UNIT_TCP_CONNECT**
- **SYNC_UNIT_RS_COM**
- **SYNC_CONTROLLER_OEM_RS_COM**
- **SYNC_EXTRA_DEBUG**
- **SYNC_FIND_ERROR**
- **SYNC_SWITCH_MODE**

- SYNC_ARCH_START_DEPTH
- TM_FT_IP_ADDR_<номер узла>
- SYNC_NO_NODE
- SYNC_ARCH_LAST_DEPTH=<число секунд>
- SYNC_ARCH_START_DEPTH=<число секунд>
- SYNC_ARCH_DISABLE_A1=ON
- SYNC_ARCH_DISABLE_A2=ON
- SYNC_ARCH_DISABLE_A3=ON
- SYNC_CPY_ARCH_BYFLAG=ON/OFF
- SYNC_HISTORY_TIME_LIVE=<число секунд>
- SYNC_CPY_ALA_BYFLAG=ON/OFF

Следующие ключи описаны в разделе **Редактор параметров COM-порта**:

- RS_DNODE_PERIOD_CHECK
- RS_DNODE_COUNT_CHAR_CHECK
- RS_DNODE_EXCHANGE_COUNT
- RS_DNODE_PERIOD_RESET
- SYNC_SWITCH_MODE_BY_COM
- RS_CHANGE_UNIT_BY_COM

Консоли

- **NLLREAD**=<nllt> – период (в секундах) чтения вспомогательной информации (например, значения атрибута удаленного канала, которым управляет консоль).

Индивидуальные архивы

- **ARCHT_NET**=<число секунд, DEC> – таймаут перехода к поиску следующего канала при запросе удаленных индивидуальных архивов по I-NET (значение по умолчанию – 60с, см. также **Запрос удаленного индивидуального архива**);
- **ARCHT_RS**=<число секунд, DEC> – таймаут перехода к поиску следующего канала при запросе удаленных индивидуальных архивов по M-LINK (значение по умолчанию – 60с).

Обмен по телефонным линиям

- **MODEM_REGIM**=<число HEX> – зарезервировано;
- **MODEMCNERR**=<Ncon> – после **Ncon** неудачных попыток соединения модем переинициализируется (т.к. соединение устанавливается для каждого канала обмена, то **Ncon** – это предельно допустимое число каналов узла MASTER, для которых не удалось

- установить соединение). По умолчанию **Ncon** = 10;
- **MODEM_REINT**=<Tcon> – если модем не удалось инициализировать, следующая попытка его инициализации предпринимается через **Tcon** мс. По умолчанию **Tcon** = 2000 мс;
 - **MODEM_SLOCK**=<Tfind> – если для узла задано удержание связи (см. описание **@Node_Lock** в разделе **Группа ДИАГНОСТИКА**), но каналы обмена с ним не обнаружены, соединение не разрывается, а следующая попытка поиска каналов обмена предпринимается через **Tfind** мс. По умолчанию **Tfind** = 250 мс;
 - **MODEM_GOLNK**=<Tfind1> – если каналы обмена по данному RS не обнаружены, следующая попытка их поиска предпринимается через **Tfind1** мс. По умолчанию **Tfind1** = 1000 мс;
 - **MODEM_TNOND**=<TlostM> – если <текущее время> – <время последнего приема от узла> > **TlostM**, считается, что узла нет. **TlostM** задается в секундах, значение по умолчанию – 3600 с;
 - **GSM_TNOND**=<TlostG> – аналог **MODEM_TNOND** для GSM-модема;
 - **MODEM_CREPT**=<Natt> – число попыток чтения ответа модема SLAVE в случае, когда соединение установлено и данные отосланы, но в течение соответствующего таймаута считано 0 байт. По умолчанию **Natt** = 2;
 - **MODEM_ERRRD**=<Nch> – если в ходе текущего сканирования базы каналов при обработке **Nch** каналов обмена с некоторым узлом **A** получены некорректные ответы (после каждого некорректного ответа соединение с узлом **A** разрывается), МРВ переходит к поиску и обработке каналов обмена со следующим узлом. При следующем сканировании каналы обмена с узлом **A** будут анализироваться. По умолчанию **Nch** = 4;
 - **MODEM_INLNK**=<Tnode> – максимальное время обмена с одним узлом. По истечении **Tnode** секунд МРВ прерывает обмен с узлом и переходит к поиску и обработке каналов обмена со следующим узлом. Ключ используется в отсутствие удержания связи. По умолчанию **Tnode** = 0.

Функции вывода

Ключи для подстановки строк (**nn**=00...15, см. **Номер SubNum**):

- **DOC_DEFLVLS**<nn>=<строка>
- **DOC_DEFLVLE**<nn>=<строка>
- **DOC_DEFENGS**<nn>=<строка>
- **DOC_DEFENGE**<nn>=<строка>
- **DOC_DEFALRS**<nn>=<строка>
- **DOC_DEFANYS**<nn>=<строка>
- **DOC_DEFANYE**<nn>=<строка>

- **DOC_DEFANYV**<nn>=<строка>

OPC

- **OPCSMSK=1** – этот ключ обеспечивает функционирование OPC-клиента TRACE MODE в узле **Logger**.

MPB как сервер протоколов поверх TCP/IP

- **SLAVE_TCPADDR**
- **SLAVE_TCPPORT**
- **SLVPUIP**=<число>

Данные ключи описаны в разделах **MPB как сервер протоколов поверх TCP/IP**.

MPB как сервер протокола МЭК 60870-104

- **SP104_CASDU_ADDR**=<адрес>;
- **IEC_SLAVE_LIST_ON_START**=ON/OFF;

- **_DONT_CLOSE_BY_ERROR**=ON/OFF
- **SP104_STOPDT**=CLOSE
- **SP104_TWO_PORT**=ON
- **SLAVE_TCPPORT**=<номер порта, DEC>
- **SP104_BLOCK_COT3**=<число HEX (0 по умолчанию)>
- **SP104_BLOCK_OUT_COT3**=<число HEX (0 по умолчанию)>
- **SP104_BLOCK_OUT_COT20**=<число HEX (0 по умолчанию)>
- **IEC_COUNTER_LIST_ON_START**=OFF/ON
- **SP104_IOA_CODE**=ON
- **SP104_START_IOA**=<число DEC>
- **SP104_MIN_IOA**=<число DEC>
- **SP104_MAX_IOA**=<число DEC>
- **SP104_USE_103**=ON
- **SP104_USE_103**=<число секунд>

- **IEC_TESTFR**=<число секунд>
- **IEC_SLAVE_TESTFR**=<n, DEC>

- **IEC_USE_PN**=OFF
- **IEC_SLAVE_KFROM**=
- **TCP_DISCONN00**=<число секунд, DEC>

- IEC_SLAVE_COT10=OFF/ON
- IEC_SLAVE_COT7=OFF/ON
- IEC_SLAVE_TESTFR(CON)=ON/OFF
- IEC_SLAVE_FIRST_TESTFR
- _SLAVE_CLOSE_SOC
- IEC_SLAVE_MANY=COT20
- IEC_SLAVE_MANY=COT3
- IEC_SLAVE_MANY=0

- IEC_ENABLE_HEX16

- SP104_USE_103

Данные ключи описаны в разделе **MPB как сервер протокола МЭК 60870-104**.

MPB как клиент протокола МЭК 60870-104

- IEC_TESTFR=<число секунд>
- IEC_FAOUT=<число секунд>
- IEC_HOST_ACK_COT=ANY
- IEC_KFROM=
- TCP_DISCONN07=<число секунд, DEC>

- IEC_HOST_SFRAME=OFF/ON
- IEC_HOST_FIRST_TESTFR
- IEC_HOST_CLOSE_SOC
- IEC_HOST_TESTFR(CON)=ON/OFF

- IEC_ENABLE_HEX16

Данные ключи описаны в разделе **Обмен по IEC 60870-104**.

Каналы CALL

Следующие ключи описаны в разделе **Канал CALL.Sum**:

- MASK_QDS_AND=qds_and_mask
- MASK_QDS_OR=qds_or_mask
- MASK_QDS_SET_FLOAT=qds_set_mask_f
- MASK_QDS_SET_HEX=qds_set_mask_hex

- **MASK_QDS_SET_DEC=qds_set_mask_dec**
- **MASK_QDS_SET_EVT=qds_set_mask_ev**
- **MASK_QDS_SET_ONE=qds_set_mask_one**
- **MASK_QDS_SET_DEF=qds_set_mask_def**

Атрибут 61

Следующие ключи описаны в разделе в разделе **Общие атрибуты каналов/Вкладка 'Основные' редактора канала:**

- **TM_FA_QDS=OFF/ON/DEF**
- **TM_FA_QDS_ACT=OFF**
- **TM_FA_QDS_REC=OFF**
- **TM_FA_QDS_TIME=ON**

Регистратор

Следующие ключи описаны в разделе **Архивирование в регистратор:**

- **USELOGGER=<DEC номер узла>;**
- **LOG_SEND=<протокол отправки данных в регистратор>;**
- **LOG_SENDTYPE=<1..7>;**
- **LOG_QSIZE=<DEC число блоков>;**
- **LOG_CHECK_TIME=<число секунд>;**
- **LOG_SEND_SLOW=<0 .. 15с>;**
- **LOG_STORY_TIME=<число секунд>;**
- **LOG_SEND_TIME=<число секунд>;**
- **LOG_SEND_OFFSET=<число секунд>;**
- **LOG_SAVE_FILE=ON/OFF**
- **LOG_USE_PORT=STD/LOG**
- **PERIOD_BLOCK_TIME_ADD**

Пользователи

- **USER_CFR=<число DEC>;**
- **EMAIL_ONLY.**

Эти ключи описаны в разделе **Канал класса ПОЛЬЗОВАТЕЛЬ.**

Электронная почта

- **EMAIL_SRV=<имя сервера исходящих сообщений (SMTP)>;**
- **EMAIL_DEF=<адрес Кому>;**
- **EMAIL_FROM=<адрес От кого>;**
- **EMAIL_LOGIN=<имя учетной записи>;**

- **EMAIL_PASSW**=<пароль учетной записи>.

Эти ключи описаны в разделе **Канал CALL.EMAIL**.

Интерпретация временных меток

- **UTC_OFFSET**=<число минут>;
- **ISDST=OFF**.

Эти ключи рассматриваются в описании ключа **UTC=ON** в разделе **Канал CALL.Vector**.

Типовые средства редактирования




В данном разделе описаны инструменты и операции редактирования, которые являются типовыми для различных редакторов. Специфические инструменты отдельных редакторов рассматриваются в разделах, посвященных работе в этих редакторах.

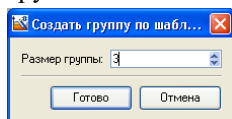
Типовые инструменты редактирования

К типовым инструментам редактирования относятся следующие:

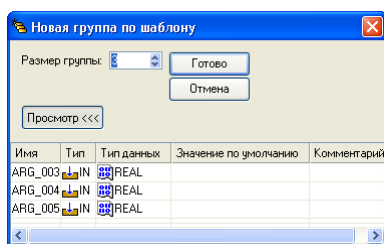


– создание новых объектов. При нажатии ЛК на иконке создается новый объект. При нажатии ЛК на стрелке открывается меню, включающее следующие команды:








-  **Создать** – добавить объект;
-  **Создать группу** – добавить группу объект;
-  **Создать по шаблону** – добавить группу объектов с характеристиками, аналогичными характеристикам выделенного объекта. При выполнении этой команды открывается диалог, в котором задается число объектов создаваемой группы:

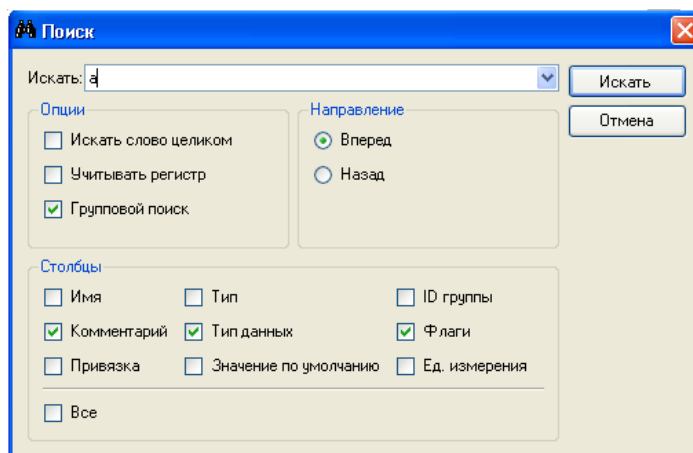


В ряде случаев данный диалог имеет окно предварительного просмотра объектов, которые будут созданы. Чтобы открыть/закрыть это окно, нужно нажать ЛК на кнопке **Просмотр**:



X (**Backspace** или **Del**) – удалить выделенные объекты. Удаление объекта, открытого в соответствующем редакторе, запрещено;

-  (Ctrl+Z) – отменить последнее действие;
-  (Ctrl+Y) – вернуть последнее отмененное действие;
-  (Ctrl+A) – выделить все;
-  (Ctrl+X) – вырезать выделенные объекты в буфер обмена;
-  (Ctrl+C) – копировать выделенные объекты в буфер обмена;
-  (Ctrl+V) – вставить содержимое буфера обмена в указанную позицию;
-  (Ctrl+F) – открыть диалог задания параметров поиска:



В общем случае поиск производится по подстроке и является нечувствительным к регистру.

В разделе **Опции** диалога могут быть заданы специальные параметры поиска:


искать только слово целиком;

учитывать регистр;


групповой поиск.



В разделе **Направление** задается направление поиска (**Назад** или **Вперед**).


В разделе **Столбцы** могут быть указаны столбцы таблицы, в которых будет производиться поиск (только для табличных редакторов).


 – окно выбора строки для поиска (аналог окна **Искать** диалога задания параметров поиска);

 – искать назад;


 – искать вперед;


 Arial  20 – выбор семейства и размера шрифта;


 **I** (Ctrl+I) – наклонный шрифт;


 **B** (Ctrl+B) – жирный шрифт;


 U (Ctrl+U) – подчеркнутый шрифт;


 – выровнять по левому краю;


 – выровнять по центру;


 – выровнять по правому краю;


 – выровнять по ширине;

 – нумерованный список;

 – маркированный список;

 – уменьшить отступ;

 – увеличить отступ;



 **A** – выбор цвета шрифта.

В редакторах имеются также контекстные меню, содержащие стандартные команды для создания и удаления элементов, работы с буфером обмена и т.п. Для вызова контекстного меню нужно нажать ПК в поле редактора.

Типовые операции редактирования

С помощью мыши:

- **Редактирование текстового поля таблицы.** Как правило, для перехода к редактированию текстового поля таблицы надо дважды нажать ЛК на этом поле. Далее нужно ввести новый текст с помощью клавиатуры и нажать ENTER.
- **Вызов контекстного меню.** Как правило, для вызова контекстного меню нужно нажать ПК на объекте (если объект имеет такое меню).
- **Выделение произвольного текста.** Для выделения произвольного текста нужно установить курсор в начальную позицию и, удерживая ЛК нажатой, переместить курсор в конечную позицию, после чего кнопку мыши отпустить.

- **Выделение слов.** Для выделения слова нужно дважды нажать на нем ЛК.
- **Выделение объекта.** Для выделения объекта (рисунка, таблицы, графического элемента и т.п.) нужно нажать на нем ЛК (при наведении на объект курсор принимает вид  или ). Как правило, при выделении на экране отображается прямоугольник, ограничивающий объект:




- **Выделение группы объектов.** Для выделения группы объектов нужно с помощью мыши обвести их контурным прямоугольником (нажать ЛК в некоторой точке рабочего поля редактора и, удерживая кнопку нажатой, переместить курсор в направлении диагонали будущего прямоугольника, после чего кнопку отпустить). Ограничивающий прямоугольник, как правило, отображается на экране.

В РПД: перемещение по диагонали вверх влево или вниз влево приводит к выделению ГЭ, частично захваченных прямоугольником, в противном случае – к выделению только полностью захваченных ГЭ.



Выделить группу объектов можно также последовательным нажатием на них ЛК с удержанием кнопки **Ctrl**. Для выделения группы объектов в списке можно нажать ЛК на некотором объекте **obj1**, после чего, удерживая клавишу **SHIFT**, нажать ЛК на некотором объекте **obj2** – по этой команде выделяются объекты, расположенные в списке между объектами **obj1** и **obj2** (объекты **obj1** и **obj2** также выделяются),



- **Перетаскивание выделенного текста, объекта или группы объектов (метод drag-and-drop).** Для использования данного метода нужно установить курсор на выделенный текст, объект или группу объектов, нажать ЛК и, удерживая кнопку нажатой, переместить курсор в нужную позицию, после чего кнопку мыши отпустить.

Методом drag-and-drop можно перемещать окна и панели инструментов редакторов интегрированной среды. Перед перемещением окна нужно установить курсор на его заголовок. Перед перемещением панели нужно установить курсор на ее левый край (курсор при этом принимает вид ).

- **Изменение размеров выделенного объекта или группы объектов.**

Для изменения размеров выделенного объекта нужно установить курсор в вершину прямоугольника, ограничивающего объект (кур-

сор при этом принимает вид  или ), нажать ЛК и, удерживая кнопку нажатой, переместить курсор в нужную точку, после чего кнопку мыши отпустить.

Для изменения высоты или ширины выделенного объекта нужно установить курсор в середину соответствующей стороны ограничивающего прямоугольника (курсор при этом принимает вид  или ) и выполнить аналогичные действия.

Аналогичным способом можно изменять размеры окон редакторов интегрированной среды.

В редакторе представления данных таким способом можно изменять размеры выделенной группы графических элементов.

С помощью клавиатуры:

- **Shift+→/←** – выделить один символ (букву, рисунок в тексте и т.п.) или снять выделение одного символа.
- **Ctrl+Shift+→/←** – выделить слово или снять выделение слова.
- **Shift+↓/↑** – выделить строку текста или снять выделение строки.
- **Shift+Home/End** – выделить текст от позиции курсора до начала/конца строки.
- **Shift+PgUp/PgDn** – выделить текст от позиции курсора до начала/конца окна.
- **Ctrl+Shift+Home/End** – выделить весь текст от позиции курсора до начала/конца.
- **Ctrl+A** – выделить весь текст.
- **Backspace** или **Del** – удалить выделенные элементы.
- **Ctrl+Z** – отменить последнее действие.
- **Ctrl+Y** – повторить последнее отмененное действие.
- **Ctrl+X** – вырезать выделенные элементы в буфер обмена.
- **Ctrl+C** – копировать выделенные элементы в буфер обмена.
- **Ctrl+V** – вставить содержимое буфера обмена в позицию курсора.
- **Ctrl+F** – открыть диалог задания параметров поиска.

Чтобы снять выделение текста или объекта, нужно нажать ЛК или любую кнопку перемещения на клавиатуре.

Сочетания клавиш в ИС

В ИС поддерживаются следующие сочетания клавиш:

- **ALT** – снять/установить фокус на главное меню ИС;
- **ENTER** – раскрыть выделенное меню (группу меню), выполнить выделенную команду; открыть выделенный компонент в редакторе;

- **ESC** – закрыть выделенное меню (группу меню); свернуть список combo-box;
- **ALT+SPACE** – открыть контекстное меню главного окна ИС;
- **TAB** – переключить фокус;
- **SPACE (ALT+ ↓)** – раскрыть список выделенного элемента combo-box;
- **→** – раскрыть выделенную группу дерева;
- **←** – свернуть выделенную группу дерева;
- **F1** – открыть контекстную справку;
- **CTRL+TAB** – переключить фокус на следующее открытое окно редактора (только для режима MDI);
- **ALT+ENTER** – открыть окно свойств выделенного объекта структуры проекта;
- **CTRL+ENTER** – перейти к редактированию имени выделенного объекта структуры;
- двойное нажатие ЛК на заголовке открытого редактора – выход из режима MDI до следующего открытия/закрытия любого редактора.

Форматы

Формат Си вывода чисел

Обозначению формата предшествует знак процента (%):

- **%d** или **%i** – вывод значения как целого со знаком в формате DEC;
- **%o** – вывод значения в восьмеричном формате без знака;
- **%u** – вывод значения как целого без знака в формате DEC;
- **%x** – вывод значения в формате HEX без знака с использованием нижнего регистра для букв;
- **%X** – вывод значения в формате HEX без знака с использованием верхнего регистра для букв;
- **%e** – вывод значения со знаком в форме [–]**D.mmmm e** [**sign**]**ddd**, где **D** – один десятичный знак, **mmmm** – один или более десятичных знаков, **ddd** – три десятичных знака, **sign** – «+» или «-»;
- **%E** – аналог **%e** с использованием **E** вместо **e**;
- **%f** – вывод значения со знаком в форме [–]**DDDD.mmmm**, где **DDDD** – один или более десятичных знаков. Число знаков перед десятичной точкой зависит от величины значения, число знаков после десятичной точки зависит от запрошенной точности. Число знаков после запятой (**k**) может быть задано при указании формата в виде **%.<k>f**;
- **%g** – вывод значения со знаком в **f** или **e** формате (в зависимости от того, в каком из этих двух форматов представление компактнее для данного числа и точности). Формат **e** используется тогда, когда показатель степени меньше -4 или больше или равен точности числа. Замыкающие нули удаляются, десятичная точка появляется только тогда, когда число дробно;
- **%G** – аналог **%g** с использованием **E** вместо **e**.

Формат Си вывода даты и времени

Обозначению формата предшествует знак процента (%), реальный вывод зависит от региональных настроек ОС:

- %a** – сокращенное наименование дня недели;
- %A** – полное наименование дня недели;
- %b** – сокращенное наименование месяца;
- %B** – полное наименование месяца;

- %c** – дата и время в соответствии с региональными настройками ОС;
- %d** – день месяца как целое число (01-31);
- %H** – часы в формате 00-23;
- %l** – часы в формате 01-12;
- %j** – день года как целое число (001–366);
- %m** – месяц как целое число (01–12);
- %M** – минуты как целое число (00–59);
- %p** – индикатор AM/PM для часов в формате 01-12;
- %S** – секунды как целое число (00–59);
- %U** – неделя года как целое число (00–53), первый день недели – воскресенье;
- %w** – день недели как целое число (0–6, воскресенье – 0);
- %W** – неделя года как целое число (00–53), первый день недели – понедельник;
- %x** – дата в соответствии с региональными настройками ОС;
- %X** – время в соответствии с региональными настройками ОС;
- %y** – год без века как целое число (00–99);
- %Y** – год с веком как целое число (YYYY);
- %z, %Z** – наименование или аббревиатура временной зоны, в зависимости от настроек реестра. Пусто, если зона неизвестна.

Любому из форматов может предшествовать знак **#**. Для различных форматов этот знак имеет различный смысл:

- ##a, ##A, ##b, ##B, ##p, ##X, ##z, ##Z, ##%** – знак **#** игнорируется;
- ##c** – длинное представление даты и времени в соответствии с региональными настройками ОС (например, Tuesday, March 14, 1995, 12:41:29);
- ##x** – длинное представление даты в соответствии с региональными настройками ОС (например, Tuesday, March 14, 1995);
- ##d, ##H, ##l, ##j, ##m, ##M, ##S, ##U, ##w, ##W, ##y, ##Y** – удаление лидирующих нулей.

Формат IP-адреса

В редакторах ИС IP-адрес задается в виде четырех числовых полей, разделенных точками.

Если число в поле начинается с отличной от 0 цифры, оно интерпретируется как десятичное, если с нуля – как восьмеричное, если с префикса **0x** – как шестнадцатеричное.

Функции общего назначения

Копирование архивов и отчета тревог

При создании резервной копии отчета тревог или архива монитор использует функцию COPY_ARCH, которая имеет аргумент DESTINATION. Значение, переданное в этот аргумент, задает имя файла резервной копии и его размещение:

- 1 – в текущей директории ОС с именем канала;
- 2 – в текущей директории ОС с именем канала, к которому через знак подчеркивания добавлено время создания копии (как число секунд с 01.01.70);
- 3 – в текущей директории ОС с именем HH_DD_MM_YY;
- 4 – зарезервировано;
- 5 – в директории размещения основного файла архива с именем канала;
- ≥ 65 – такие значения интерпретируются как ANSI-коды (например, 65 – это прописная латинская буква A). В этом случае к символу, соответствующему заданному коду, добавляется двоеточие, слеш и имя файла (HH_DD_MM_YY) и таким образом указывается путь к резервной копии архива.

Если значения DESTINATION, перечисленные выше, увеличить на 16, имя файла не меняется, но сохраняется файл в поддиректории, заданной атрибутом (80, **CMNT**) канала.

Если при конфигурировании для файла архива было указано расширение (см. **Задание параметров узла**), оно добавляется к имени файла резервной копии, в противном случае добавляется расширение, используемое в TRACE MODE по умолчанию, – т.е. **rep** для SIAD и **evn** для OT.

Номер SubNum

SubNum задает функции вывода.

Описание или цветовое обозначение кодов

- **300** – значение атрибута (0,**R**) канала, привязанного к аргументу; цвет зависит от атрибута (4,**I**) канала:
 - I=0 – **цвет1**;
 - I=2 – **цвет5**;
 - I=1 или 3 – **цвет2**;

- **301** – «OFF», если аргумент равен 0; в противном случае – «ON».

Подстановка строк

Следующие значения **SubNum** выводят строку, заданную соответствующим ключом в файле *.cnf (**nn**=00...15 – значение аргумента, см. **Задавание параметров работы мониторов**):

- **302** – DOC_DEFVLVS<nn>
- **303** – DOC_DEFVLVE<nn>
- **304** – DOC_DEFENGS<nn>
- **305** – DOC_DEFENGE<nn>
- **306** – DOC_DEFALRS<nn>
- **307** – DOC_DEFANYS<nn>
- **308** – DOC_DEFANYE<nn>
- **309** – DOC_DEFANYV<nn>
- **402–409** – см. **Запись длинных строк в канал CALL**.

Вывод времени

Следующие значения **SubNum** выводят в соответствующем формате значение аргумента с типом данных DATE_AND_TIME, к которому может быть привязан канал TIME, атрибут (45, T) или аргумент с типом данных DATE_AND_TIME:

- **445** – дата в формате **DD.MM.YY**;
- **446** – время в формате **HH:MM:SS**;
- **447** – дата в формате **DD <полное наименование месяца> YYYY**;
- **448** – полное наименование дня недели;
- **449** – полное наименование месяца;
- **450** – полное наименование предыдущего месяца;
- **451 – 447** с добавлением «Г.»;
- **452** – длинное представление даты и времени (**6 Март 2008 г. 17:36:00**);
- **453** – дата с сокращенным наименованием месяца (**06 мар 2008**);
- **454** – сокращенное наименование дня недели.
- **455** –
- **456** –
- **457** –

Вывод канала CALL

Значения **SubNum**, перечисленные в этом разделе, выводят в виде таблицы канал CALL одной из следующих конфигураций:

- канал CALL с данными (далее – **call**, таким каналом может быть CALL.TVC, CALL.ChGroupReq или CALL.AS_DATA);
- канал CALL, к аргументам которого привязаны каналы CALL с данными (далее такая конфигурация обозначается как **root-call**);
- канал CALL, к аргументам которого привязаны каналы CALL, к аргументам которых привязаны каналы CALL с данными (далее такая конфигурация обозначается как **root-subroot-call**).

Если к **root/subroot** привязан **call.R**, то выводятся данные **call**, а если **call.A**, то выводятся данные и, дополнительно, строка, содержащая аппаратное значение **call**.

Каналом **root/subroot** может быть CALL.ChGroupReq, CALL.AS_DOCUMENT или канал выборки из SIAD, а также канал CALL.ROOT (см. **Канал CALL.ROOT**).

Конфигурацию «канал CALL.ChGroupReq, к аргументам которого привязаны каналы CALL.ChGroupReq» использовать не рекомендуется.

Формат таблицы, а также форматы вывода значений и меток времени могут быть заданы (см. описание атрибута 83, **SGNL** в разделе **Атрибуты каналов, отображаемые профайлером**). Если в **call** форматы значений и меток времени не заданы, они наследуются от **root (subroot)**.

SubNum = 327-330

Используются для генерации столбцов, заголовков которых разбит на две части А и В (в шаблоне задается **@<аргумент>.327**):

| А | | | | |
|---|---|---|---|---|
| В | В | В | В | В |
| | | | | |

- А – имя канала, привязанного к аргументу;
- В – число таких столбцов равно числу кривых в CALL.TVC (в этом случае В – это номер кривой) или числу привязок в CALL.ROOT (в этом случае В – это имя привязки (канала)).

Помимо задания вывода данных, в столбце шаблона должны задаваться

управляющие SubNum:

- **328** – признак начала таблицы;
- **329** – признак окончания текущей ячейки и начала новой ячейки;
- **330** – признак окончания текущей ячейки.

Для вывода архивных данных в шаблоне нужно указать **@<аргумент>.R**, при этом к аргументу должен быть привязан атрибут 124, **ArgSize** канала.

Подобные столбцы не суммируются в строке ИТОГО.

SubNum = 337

Вывод HTML-тега `
` (перенос строки).

SubNum = 520-528

Варианты **SubNum = 520-528** не предназначены для конфигураций **root-call** и **root-subroot-call**.

SubNum = 520-524 – таблицы аргументов по строкам (горизонтальные).

- **520** – заголовок (1 строка, содержит имя и комментарий канала CALL) + (время + значение) 2 столбца (для TVC – 1 + число столбцов-кривых).
- **521** – (время + значение) 2 столбца (для TVC – 1+число столбцов-кривых).
- **522** – аналог **520**, но без заголовка и без времени.
- **523** – (значение) 1 столбец (для TVC – число столбцов-кривых).
- **524** – (значение) 1 столбец (для TVC – число столбцов-кривых) без сетки.

SubNum = 525-528 – таблицы аргументов по столбцам (вертикальные).

- **525** – заголовок (1 строка) + (время + значение) 2 строки (для TVC – 1 + число строк-кривых).
- **526** – (время + значение) 2 строки (для TVC – 1 + число строк-кривых).
- **527** – (значение) 1 строка (для TVC – число строк-кривых).
- **528** – (значение) 1 строка (для TVC – число строк-кривых) без сетки.

Если задано время, а его нет, выводится номер аргумента (для CALL.ChGroupReq, CALL.Vector и CALL.AS_DATA).

Для вариантов **520**, **521** и **524** в файле `../<папка узла>/col_name.txt` можно задать имена столбцов (под имена столбцов выделяется массив 8x8). В файле допускаются два варианта задания элементов массива:

- задание одного элемента (**n**, **m**):
[n, m]

```

<строка>
• задание восьми элементов ((n,1)...(n,8)):
[n]
<строка1>
...
<строка8>

```

Все строки в файле должны начинаться с нулевого знакоместа. Если первый символ строки – точка с запятой, строка интерпретируется как комментарий. Последняя строка файла должна быть пустой.

SubNum = 536-543 и SubNum = 552-559

В новых проектах не рекомендуется использовать данные SubNum, т.к. в последующих релизах они не будут поддерживаться.

SubNum = 536-543 – вертикальные таблицы, **SubNum = 552-559** – горизонтальные таблицы для **root-call** и **root-subroot-call** (**root/subroot** – CALL.ChGroupReq или CALL.AS_DOCUMENT; для нечетных **SubNum** выводится таблица без заголовка). Для горизонтальных таблиц число столбцов определяется по каналу с максимальным числом аргументов. При некорректной привязке вместо таблицы выводится слово «ERROR». Для **call** данные значения **SubNum** аналогичны **520**. Для **root-call** и **root-subroot-call** – вывод привязанных каналов CALL в виде подтаблиц в основной таблице. Форматирование основной таблицы и подтаблиц аналогично **520**. Для привязанного CALL.TVC:

- в случае **root-call**: если привязан атрибут **N** ($N < 140$), выводятся все кривые (значения и их времена); если атрибут 142, **ARG02** и т.д., выводится только соответствующая кривая (значения и их времена), если атрибут 140, **ARG00**, выводится только столбец времени;
- в случае **root-subroot-call**: при привязке атрибута 142, **ARG02** и т.д. выводится соответствующая кривая.

Если к аргументам (REAL) **root** привязаны атрибуты 45, **T** каналов CALL.ChGroupReq или CALL.AS_DATA, в таблицу выводятся времена, рассчитанные по атрибутам 59 и 252. Если аргументы **root** имеют тип данных DATE_AND_TIME, вместо времен выводятся интервалы.

Если к аргументам **root** привязаны атрибуты ChGroupReq.45 или TVC.140, в таблицу не выводятся имена каналов.

Существует 3 типа вывода информации в заголовок таблицы:

- обычный (далее – **вывод_0**);
- расширенный 1 (при **SubNum = 538, 539, 542, 543, 554, 555, 558 и 559**, далее – **вывод_1**);

- расширенный 2 (при **root** = CALL.AS_DOCUMENT, далее – **вывод_2**).

Для **root** в заголовок таблицы выводится:

- **вывод_0** – имя и комментарий;
- **вывод_1** – имя, кодировка и комментарий;
- **вывод_2** – имя, кодировка и комментарий.

Для **subroot** в заголовок таблицы выводится:

- **вывод_0** – имя;
- **вывод_1** –
- **вывод_2** – 2 строки, первая содержит имя и кодировку, вторая – комментарий.

Для **call** в заголовок таблицы выводится (имена столбцов/строк):

- **вывод_0** – имя;
- **вывод_1** – имя и комментарий;
- **вывод_2** – имя, кодировка и комментарий.

SubNum = 544-549

Допускается конфигурация **root-subroot.124-subroot.124-call**.

Допускается также конфигурация CALL.ROOT-CGR_main (см. **Универсальный механизм обмена с электросчетчиками**).

Канал **call** может быть любым каналом класса CALL без привязки (10.0), обычно привязывается к аргументу INPUT **root/subroot..**

Если **call** – CALL.TVC, то при привязке **call.0** выводятся все кривые, при привязке **call.142**, **call.143** и т.п. – только соответствующая кривая, при привязке **call.59** – времена.

Если **call** – CALL.ChGroupReq, также допускается привязка к **call.59**.

Вывод аргументов INPUT без привязки каналов **root/subroot** зависит от типа данных аргумента:

- REAL, LREAL – вывод значения аргумента в строку после основной таблицы;
- UINT – в ячейку пишется номер строки;
- INT, DINT – в ячейку пишется номер строки и значение аргумента;
- BOOL – вывод пустого столбца;
- временной – вывод значения аргумента, преобразованного во временной формат;
- SINT, USINT – вывод N знаков подчеркивания (N – значение ар-

гумента).

Аргументы OUTPUT каналов **root/subroot** используются следующим образом:

- к аргументу привязан этот же **root/subroot** – вывод аппаратных значений **call** (предшествующих **subroot-call**);
- к аргументу привязан **call** (CALL.ChGroupReq) – вывод данных **call** в строку после основной таблицы; вывод зависит от привязанного атрибута **call**:
 - 0, **R** – вывод всех аргументов;
 - 2, **In** – вывод аргументов с привязкой;
 - 9, **Q** – вывод аргументов без привязки;
 - 142, **ARG02** – вывод аргументов с привязкой к аргументам;
- к аргументу **root** привязан CALL.STRING.In (CALL.STRING.R) – строки CALL.STRING задают имена столбцов (строк).

Биты атрибута **Параметр** (34, **FPrnt**) канала **root** используются для управления выводом (1 – запрет указанной функции вывода):

- бит 0 (0x1) – нумерация столбцов вверх;
- бит 1 (0x2) – нумерация столбцов вниз;
- бит 2 (0x4) – именование столбцов вверх;
- бит 3 (0x8) – именование столбцов вниз;
- бит 4 (0x10) – вставка пустых строк (если запрещено, пустые строки сжаты);
- бит 5 (0x20) – добавление размерности (82, **DIM**) к имени столбца данных **call**;
- бит 6 (0x40) – зарезервировано;
- бит 7 (0x80) – вывод в столбец, соответствующий аргументу INPUT, имени канала **call**, привязанного к последующему аргументу OUTPUT (данные такого **call** выводятся в строку).

Вывод информации о резервах

Используется **SubNum=644** (root-html-column).

Вывод информации об OPC-серверах

SubNum=645 выводит таблицу, которая содержит следующую информацию об OPC-серверах: номер, имя, статус, время последнего подключения, время последнего обмена и общее число ошибок (root-html-row).

Вывод информации о проблемах

Используется **SubNum=646** (root-xml).

Вывод статусов узлов

Используется **SubNum=647** (root-xml).

Вывод информации об обмене по RS

Используется **SubNum=648**.

Вывод информации об обмене по TCP

Используется **SubNum=649**.

Вывод протокола действий пользователей

Используется **SubNum=560** (см. Канал класса ПОЛЬЗОВАТЕЛЬ).

Вывод другой информации

SubNum=701 – последнее имя файла, сгенерированного этим каналом.

Подтипы каналов

Подтип и **дополнение к подтипу** являются характеристиками каналов, номера этих параметров отображаются профайлером (атрибут 126, TsT – см. **Атрибуты каналов, отображаемые профайлером**).

Подтип 0

Этот подтип имеют каналы всех классов (кроме CALL и каналов T-FACTORY) с ненастроенным свойством **СВЯЗЬ**.

Подтип 1

Этот подтип имеют каналы обмена с платами аналогового ввода/вывода (см. **Группа 'Платы ввода-вывода'**):

- **AI_Kruiz** – ввод в контроллере КРУИЗ (дополнение к подтипу – 0);
- **AO_Kruiz** – вывод в контроллере КРУИЗ (дополнение к подтипу – 0);
- **AI_MFC** – ввод в контроллере МФК (дополнение к подтипу – 1);
- **AO_MFC** – вывод в контроллере МФК (дополнение к подтипу – 1);
- **AI_5710** – плата 5710 фирмы Octagon Systems (дополнение к подтипу – 2);
- **AI_5720** – плата 5720 фирмы Octagon Systems (дополнение к подтипу – 3);
- **AI_5700** – плата 5700 фирмы Octagon Systems (дополнение к подтипу – 4);
- **AI_5648** – плата 5648 фирмы Octagon Systems (дополнение к подтипу – 5);
- **AI_PCI_Advantech** – платы PCI аналогового ввода фирмы ADVANTECH (дополнение к подтипу – 6);
- **ISO_AD32** – плата AD32 фирмы ICP-DAS (дополнение к подтипу – 7);
- **In_8254_Timer** – чтение текущего значения таймеров и счетчиков на базе микросхем I8253, I8254 и их аналогов (дополнение к подтипу – 8);
- **Out_8254_Timer** – задание уставки таймеров и счетчиков на базе микросхем I8253, I8254 и их аналогов (дополнение к подтипу – 8);
- **OEM_AI** – аналоговый ввод в контроллере ADAM5510 (дополнение к подтипу – 9);
- **LA_AI16** – плата AI16 фирмы LanAutomatic (дополнение к подтипу – 10);
- **LA_UNI24/48** – платы UNI24 и UNI48 фирмы LanAutomatic (дополнение к подтипу – 11);

- **LA_AI8S** – плата AI8S фирмы LanAutomatic (дополнение к подтипу – 11);
- **PCL_channel** – платы PCL711 и PCL813 фирмы Advantech (дополнение к подтипу – 12);
- **PCL_AI** – платы контроллера MIC-2000 и PCL818 фирмы Advantech. Мультиплексированием входов платы PCL-818 управляет настройка **MUX** этого канала. Сигнал на управление мультиплексором подается в порт **BASE+3** (дополнение к подтипу – 13);
- **A_I/O** – платы аналогового ввода/вывода контроллера LOMICONT (дополнение к подтипу – 14);
- **AI_RWH** –
чтение аналоговых данных через драйвер (см. **Драйверы t11 и t12** и **Каналы для вызова драйвера RWH**) (дополнение к подтипу – 15);
- **AOutput** – плата аналогового вывода 5750 фирмы Octagon Systems (дополнение к подтипу – 18)
- **PS-IO** – аналоговый ввод/вывод в контроллере FESTO (дополнение к подтипу – 19);
- **ISO_DA_16/8** – платы DA16, DA8 фирмы ICP-DAS. В платах можно использовать 2 порта **DI** и 2 порта **DO**. Базовые адреса – (**BASE+0**) и (**BASE+1**). Для канала **DO** с адресом (**BASE+0**) следует в настройке **STATE** установить (**+2<-FF**), для канала **DO** с адресом (**BASE+1**) – (**+1<-FF**) (дополнение к подтипу – 20);
- **INT86_in/out** – ввод/вывод аналоговых сигналов на платах 60xx фирмы Octagon Systems (ввод при **BASE = F800**, **MUX = 2**; вывод – при **BASE = F801**, **MUX = 2**) (дополнение к подтипу – 22);
- **OEM_AO** – аналоговый вывод в контроллере ADAM5510 (дополнение к подтипу – 23);
- **AIO_Trei** – аналоговый ввод-вывод в контроллере Trei (дополнение к подтипу – 25);
- **OEM_Register** – плата PCL-839 (запись/чтение текущих значений регистров R0...R7) (дополнение к подтипу – 26);
- **AO(W)** – платы аналогового вывода с передачей слов (дополнение к подтипу – 27);
- **LA_AO16** – плата аналогового вывода AO16 фирмы LanAutomatic (дополнение к подтипу – 28);
- **AO(L,H)_PCL** – платы аналогового вывода фирмы Advantech с прямой последовательностью передачи байтов (дополнение к подтипу – 29);
- **AO(H,L)_PCL** – платы аналогового вывода фирмы Advantech с обратной последовательностью передачи байтов (дополнение к подтипу – 30);
- **AO_RWH** –
запись аналоговых данных через драйвер (см. **Драйверы t11 и**

t12 и **Каналы для вызова драйвера RWH**) (дополнение к подтипу – 31);

- **WinCon_AI** – аналоговый ввод в контроллере WinCON/WinPAC/XPAC (дополнение к подтипу – 32);
- **WinCon_AO** – аналоговый вывод в контроллере WinCON/WinPAC/XPAC (дополнение к подтипу – 33);
- **ADAM5510_AI** – аналоговый ввод в контроллере ADAM 5510 (дополнение к подтипу – 40);
- **ADAM5510_AO** – аналоговый вывод в контроллере ADAM 5510 (дополнение к подтипу – 41).

Для конфигурирования шаблонов каналов подтипа 1 в ИС встроен редактор.

Кроме атрибутов, общих для всех источников/приемников (см. **Редакторы источников (приемников)**), а также **Шаблоны каналов обмена**), в редакторе задаются следующие атрибуты:

- **Базовый адрес (BASE)** – базовый адрес платы (номер слота для МФК). Базовый адрес УСО должен быть кратным 32;
- **Канал (CH)** – номер канала на плате;
- **Мультиплексор (MUX)** – номер канала на мультиплексоре;
- **GAIN/REG** – параметр усиления (целое число в формате DEC);
- **ALT/REG** – сдвиг в битах считанного значения перед записью в канал.

При использовании плат PCI аналогового ввода фирмы Advantech, обменивающихся по каналам с дополнением **AI_PCI_Advantech**, в операционной системе, не поддерживающей Plug&Play и не позволяющей автоматически выделить ресурсы для устройств на шине PCI, может возникнуть необходимость использовать утилиты от производителя, позволяющие производить инициализацию этой платы с целью определения ее базового адреса. Полученный таким образом адрес используется при настройке каналов обмена с платой. Процедуру инициализации в таких ОС необходимо осуществлять перед каждым запуском MicroRTM.

Для каналов с дополнениями **In_8254_Timer** и **Out_8254_Timer** атрибуты имеют следующее значение:

- **BASE** – базовый адрес микросхемы;
- **CH** – номер счетчика;
- **MUX** – тип микросхемы:
 - 0 – микросхема I8254;
 - 1 – микросхема I8253;
- **GAIN/REG** – опрашиваемая информация (только для типа INPUT):
 - 0 – текущее значение;

- 1 – статус;
- 2 – текущее значение с последующим сбросом;
- **ALT/REG** – режим работы таймера/счетчика (для типов OUTPUT и INPUT при значении **GAIN/REG** = 2).

При программировании таймеров ТМС-10 и АСL-8454 атрибут **MUX** используется для задания номера микросхемы. Значение старших 4 битов этого атрибута равно величине (**номер микросхемы +1**).

Для канала **AI_MFC** атрибут **GAIN/REG** имеет следующие значения для плат аналогового ввода:

- 0 или 2 – считывание двуполярного значения напряжения;
- 1 – считывание однополярного напряжения.

Кроме того, атрибуты каналов данного подтипа имеют специфическое назначение для отдельных видов устройств (см. раздел **Особенности устройств**).

Обмен с i-8093

Поддерживается только в WinPAC и XPAC.

Используются каналы 1.26 **OEM_Register**.

4-канальная плата, может быть сконфигурирована работа с 3 каналами. Канал задается младшими битами параметра **Канал** (ia.c[2]).

Режим (тип счета) задается параметром **GAIN/REG** (ia.c[4]) (см. документацию платы):

- 1 – CW/CCW counting mode
- 2 – Pulse/Direction counting mode
- 3 – AB Phase (Quadrant counting) mode

Если **GAIN/REG** = 0x80, режим не устанавливается, а считывается текущий режим.

В зависимости от старших битов C2, из канала считывается:

- C2 (старшие биты равны 0) – 32-bit encoder;
- C2+8 – статус линий (бит 0 – линия А, бит 1 – линия В, бит 2 – линия С);
- C2+16 – частота;
- C2+32 – getIndex;
- C2+40 – getXOR.

Запись:

- C2 (старшие биты равны 0) – сброс 32-bit encoder;
- C2+32 – setXOR;
- C2+48 – запись в encoder.

Подтип 2

Этот подтип имеют каналы обмена с платами дискретного ввода/вывода (см. **Группа 'Платы ввода-вывода'**):

- **DI_Kruiz** – ввод в контроллере КРУИЗ (дополнение к подтипу – 0);
- **DO_Kruiz** – вывод в контроллере КРУИЗ (дополнение к подтипу – 0);
- **DI_MFC** – ввод в контроллере МФК (дополнение к подтипу – 1);
- **DO_MFC** – вывод в контроллере МФК (дополнение к подтипу – 1);
- **DI_port** – чтение из порта ввода/вывода (микросхема 8255) (дополнение к подтипу – 2);
- **DO_port** – запись в порт ввода/вывода (микросхема 8255) (дополнение к подтипу – 2);
- **DI_word** – ввод, 16 бит (дополнение к подтипу – 3);
- **DO_word** – вывод, 16 бит (дополнение к подтипу – 3);
- **DI_Trei** – ввод в контроллере Trei (дополнение к подтипу – 4);
- **DI** – ввод в контроллере Lomicont (дополнение к подтипу – 5);
- **DO** – вывод в контроллере Lomicont (дополнение к подтипу – 5);
- **OEM_DI_8** – ввод, 8 бит (дополнение к подтипу – 6);
- **OEM_DI_16** – ввод, 16 бит (дополнение к подтипу – 7);
- **OEM_DI_32** – ввод, 32 бита (дополнение к подтипу – 8);
- **OEM_Status** – чтение регистра статуса канала платы PCL-839. Тип канала должен быть INPUT (дополнение к подтипу – 9);
- **OEM_SysKey** –
- **In_EZ** – ввод на платах 60xx фирмы OCTAGON SYSTEMS (дополнение к подтипу – 13);
- **Out_EZ** – вывод на платах 60xx фирмы OCTAGON SYSTEMS (дополнение к подтипу – 13);
- **DI_RWH** –
чтение данных через драйвер (см. **Драйверы t11 и t12** и **Каналы для вызова драйвера RWH**) (дополнение к подтипу – 15);
- **DO_Trei** – вывод в контроллере Trei (дополнение к подтипу – 15);
- **OEM_DO_8** – вывод, 8 бит (дополнение к подтипу – 22);
- **OEM_DO_16** – вывод, 16 бит (дополнение к подтипу – 23);
- **OEM_DO_32** – вывод, 32 бита (дополнение к подтипу – 24);
- **OEM_Command** – взаимодействие с командным буфером платы PCL-839. Тип канала должен быть INPUT (дополнение к подтипу – 25);
- **OEM_Led** – дополнение к подтипу – 26. Если значение канала **OEM_Led** не равно 0, на экране появляется окно с сообщением **<ИМЯ канала OEM_Led>:<значение канала OEM_Led>**. В заголовке отображается время открытия окна;

- **DO_RWH** – вывод данных через драйвер (см. **Драйверы t11 и t12** и **Каналы для вызова драйвера RWH**) (дополнение к подтипу – 31);
- **WinCon_DI_8** – дискретный ввод (8 бит) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 32);
- **WinCon_DO_8** – дискретный вывод (8 бит) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 33);
- **WinCon_DI_16** – дискретный ввод (16 бит) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 34);
- **WinCon_DO_16** – дискретный вывод (16 бит) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 35);
- **WinCon_DI_32** – дискретный ввод (32 бита) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 36);
- **WinCon_DO_32** – дискретный вывод (32 бита) в контроллере WinCon/WinPAC/XPAC (дополнение к подтипу – 37);
- **ADAM5510_DI_8** – дискретный ввод (8 бит) в контроллере ADAM 5510 (дополнение к подтипу – 40);
- **ADAM5510_DO_8** – дискретный вывод (8 бит) в контроллере ADAM 5510 (дополнение к подтипу – 41);
- **ADAM5510_DI_16** – дискретный ввод (16 бит) в контроллере ADAM 5510 (дополнение к подтипу – 42);
- **ADAM5510_DO_16** – дискретный вывод (16 бит) в контроллере ADAM 5510 (дополнение к подтипу – 43).

Для конфигурирования шаблонов каналов подтипа 2 в ИС встроен редактор.

Кроме атрибутов, общих для всех источников/приемников (см. **Редакторы источников (приемников)**), а также **Шаблоны каналов обмена**), в редакторе задаются следующие атрибуты:

- **Базовый адрес (BASE)** – базовый адрес платы (номер слота для МФК). Базовый адрес УСО должен быть кратным 32;
- **Канал (CH)** – номер канала на плате;
- **Мультиплексор (MUX)** – номер канала на мультиплексоре;
- **Состояние (STATE)** – тип инициализации портов микросхемы 8255 при старте.

Настройки каналов данного подтипа имеют специфическое назначение для отдельных видов устройств (см. раздел **Особенности устройств**).

Подтип 3

Этот подтип имеют каналы обмена по OPC (см. **Группа 'OPC-группа', Обмен по OPC**).

Дополнение к подтипу указывает интерфейс: 1 – OPC, 2 – OPC HDA.

Подтип 8

Этот подтип имеют каналы обмена с модулями распределенного УСО (см. Группа ‘Распределенные УСО’):

- **AI** – аналоговый ввод с модулей с одним аналоговым входом (дополнение к подтипу – 0);
- **CJC** – температура холодных спаев модулей опроса термопар (дополнение к подтипу – 1);
- **D_DI/O/A** – дискретные входы модулей аналогового ввода (дополнение к подтипу – 2);
- **rHAlarm** – чтение верхней аварийной границы модулей аналогового ввода (дополнение к подтипу – 3);
- **rLAlarm** – чтение нижней аварийной границы модулей аналогового ввода (дополнение к подтипу – 4);
- **ECount** – значение счетчика событий (дополнение к подтипу – 5);
- **ReadBack** – мониторинг аналогового выхода модулей аналогового управления (дополнение к подтипу – 6);
- **DI** – значения дискретных входов. Для модуля I-7052 (8-разрядного) ответ размещается в старшем байте канала (дополнение к подтипу – 7);

Если **Канал**=7, канал 8.7 генерирует команду @aa и ожидает !DDDD. Ответ обрабатывается как число HEX.

- **Ain** – значение аналогового ввода модулей, имеющих несколько аналоговых сигналов (номер входа задается с помощью настройки **CH**) (дополнение к подтипу – 8);
- **Frq/Counter** – величина частоты/счетчика (дополнение к подтипу – 9);
- **rMaxCounter** – величина уставки счетчика (дополнение к подтипу – 10);
- **rOverflow Flag** – мониторинг флага переполнения счетчика (дополнение к подтипу – 11);
- **Ai8** – то же, что и **Ain**, но реализуется групповой запрос (дополнение к подтипу – 12). Запрос осуществляется первым из каналов (с настройкой **CH**=0), относящихся к одному модулю. Остальные каналы этого модуля должны иметь такое же дополнение и размещаться за первым каналом. Для самого первого канала в групповом запросе параметр **CH** может принимать следующие значения:
 - 0, 32, 64 (DEC) – при этом запрос к модулю будет следующего вида: #NN, где NN – адрес DCS-модуля;
 - 16, 48 (DEC) – при этом запрос к модулю будет следующего вида: \$NNA, где NN - адрес DCS-модуля;

- **AO** – управление величиной аналогового выхода (дополнение к подтипу – 16);
- **DO** – управление дискретными выходами (дополнение к подтипу – 17);
- **DO/AI** – управление дискретными выходами модулей аналогового ввода (дополнение к подтипу – 18);
- **SetHAlarm** – управление значением верхней аварийной границы модулей аналогового ввода (дополнение к подтипу – 19);
- **SetLAlarm** – управление значением нижней аварийной границы модулей аналогового ввода (дополнение к подтипу – 20);
- **Start/Stop_Counter** – запуск/остановка счетчика (дополнение к подтипу – 21):
 - 0 – остановить;
 - 1 – запустить;
- **Clear_Counter** – сброс счетчика (дополнение к подтипу – 22);
- **wMaxCounter** – управление уставкой счетчика (дополнение к подтипу – 23). Если данный канал задает значение аварийной границы для канала 0 счетчика (в случае разрешенного в модуле аварийного режима 0), настройка СН должна быть равна 16 (DEC); если для канала 0 счетчика задается максимальная величина счета, настройка СН должна быть равна 0. Соответственно, настройки СН для канала 1 счетчика должны быть увеличенными на 1. Допустимый диапазон значений каналов с данным дополнением к подтипу составляет 0...65535;
- **CJC_Calibrate** – калибровка компенсации температуры холодных спаев термопар (дополнение к подтипу – 24). Посылаемое значение должно быть целым числом в диапазоне от -1000 до +1000. Оно умножается в модуле на 0.01, и результат добавляется к измеренной величине CJC;
- **Offset_Calibrate** – калибровка смещения нуля (дополнение к подтипу – 25). Любое изменение значения канала приведет к передаче команды выполнить калибровку, алгоритм и условия выполнения которой задаются фирменными утилитами при настройке модуля;
- **DOutput** – управление 13-битовым дискретным выходом контроллера I-7042 (дополнение к подтипу – 26). Значения 0 и 1 настройки **SLOT** этого канала задают соответственно регистры A и B на модуле I-7042, 2 и 3 – соответственно регистры L и H на модуле NU6056, 4 – выбор третьего регистра на модуле с 24 дискретными выходами;
- **DO16** – дискретный выход, 16 бит (дополнение к подтипу – 108).

Для конфигурирования шаблонов каналов подтипа 8 в ИС встроен редактор.

Кроме атрибутов, общих для всех источников/приемников (см. **Редакторы источников (приемников)**), а также **Шаблоны каналов обмена**), в редакторе задаются следующие атрибуты:

- **Номер порта (RS)** – номер последовательного интерфейса (0 – COM1, ..., 31 – COM32);

- **Адрес (ADDR)** – адрес устройства (0...255). Для внешних модулей DCS, подключенных по RS, адрес не может быть равен 0;
- **Канал (CH)** – номер канала на модулях группового ввода и вывода аналоговых сигналов:
 - для модулей ввода – от 0 до 7,
 - для модуля I-7024 – от 1 до 4,
 - для модуля I-7022 – 16 и 17 (DEC),
 - для модуля I-87057 – 0 и 1,
 - в остальных случаях, кроме модуля I-7016, - всегда 0.
- **Слот (SLOT)** – номер посадочного места для контроллеров ADAM-5000 (от 1 до 4). Модуль ADAM-5080 должен быть настроен на выдачу данных в десятичном формате. Для модулей I-7042, I-7043, NuDAM-6056, 6058 значение (0, 1, 2, 3, 4) этой настройки определяет порт записи в модуле (соответственно А, В, L, Н, С). Для других модулей эта настройка должна быть равна 0;
- **Контрольная сумма** – признак вычисления контрольной суммы.

Каналы DCS/AI (все варианты) и DCS/AO обмениваются данными с модулями аналогового ввода/вывода в формате инженерных единиц.

Настройки каналов данного подтипа имеют специфическое назначение для отдельных видов устройств (см. раздел **Особенности устройств**).

Каналы данного подтипа используются для обмена с модулями с помощью ASCII-команд (**c** – I1 – канал; **s** – C4 – слот).

| Дополнение к подтипу | ASCII-команда | Число байт в команде и ответе (в скобках – число байт, извлекаемых из ответа) | Прим. |
|----------------------|---------------|---|-------|
| 125 ((I1 & 0x80)==0) | \$aa3c | 5, 4 | |
| 0 | #aa | 3, 9 | |
| | #aa | 3. 6 (4) | I1=5 |
| | #aa | 3. 10 (8) | I1=6 |
| 1 | \$aa3 | 4, 9 | |
| | \$aaSs3 | 6, 9 | |
| 2 | @aaDI | 5. 9 | I1=0 |
| | @aaDI | 5, 6 | I1=1 |
| | \$aaB | 4, 6 | I1=2 |
| | \$aaSsCcS | 8, 6 | |

| Дополнение к подтипу | ASCII-команда | Число байт в команде и ответе (в скобках – число байт, извлекаемых из ответа) | Прим. |
|----------------------|---------------|---|--------|
| | | | |
| 3 | @aaRH | 5, 11 | I1=0 |
| | @aaRA | 5, 12 | I1=1 |
| | \$aaSsCcRHU | 10, 11 | |
| | | | |
| 4 | @aaRL | 5, 11 | I1=0 |
| | @aaRP | 5, 12 | I1=1 |
| | \$aaSsCcRLU | 10, 11 | |
| | | | |
| 5 | @aaRE | 5, 9 | |
| | | | |
| 6 | \$aa8 | 4, 10 | I1=0 |
| | \$aa8c | 5, 10 | I1>0 |
| | \$aaSsCc6 | 8, 10 | |
| | | | |
| 7 | \$aa6 | 4, 8 (4) | I1=0 |
| | \$aa6 | 4, 10 (8) | I1=2 |
| | \$aaSs6 | 6, 10 | |
| | | | |
| 8 | #aac | 4, 9 | I1<16 |
| | #aac | 4, 9 | I1>=32 |
| | #aac | 4, 12 | I1>=64 |
| | #aaSsCc | 7, 9 | |
| | \$aaSsCc | 7, 9 | I1>=32 |
| | | | |
| 9 | #aac | 4, 10 | |
| | \$aaSsCc | 7, 10 | |
| | | | |
| 99 | #aaSsCc | 7, 12 | |
| | | | |
| 10 | \$aa3c | 5, 12 | |
| | | | |
| 11 | \$aa7c | 5, 5 | |
| | \$aaSs7 | 6, 12 | |
| | | | |

| Дополнение к подтипу | ASCII-команда | Число байт в команде и ответе (в скобках – число байт, извлекаемых из ответа) | Прим. |
|----------------------|----------------|---|-------------|
| 12 | \$aaA | 4, 34 | I1=16...31 |
| | #aa | 3, 58 | I1=0...15 |
| | #aa | 3, 82 | I1=64 |
| | \$aaSs | 5, 58 | I1>=32 |
| | #aaSs | 5, 58 | I1<16 |
| | | | |
| 16 | #aaDD.DDD | 9, 2 | I1=0 |
| | #aac+DD.DDD | 11 | I1=1...8 |
| | #aacDD.DDD | 10 | I1=16...23 |
| | \$aa7+DD.DDD | 11 | I1=32 |
| | #aaCc+DD.DDD | 12 | I1=49...55 |
| | #aaCcDD.DDD | 11 | I1=56...63 |
| | \$aaSsCcDDD.DD | 13 | I1>=4, s<>0 |
| | \$aaSsCcDD.DDD | 13 | I1<4, s<>0 |
| | | | |
| 17 | #aa00XX | 7 | I1=0 |
| | #aa0cXX | 7 | c='A'+I1 |
| | \$aaSs00XX | 9 | |
| | | | |
| 108 117 | #aa00XXXX | 9 | I1=0 |
| | #aaXXXXXXXX | 11 | I1=1 |
| | @aaXXXXXXXX | 11 | I1=2 |
| | #aaSc | | s<>1 |
| | \$aaSs00XXXX | 11 | |
| | #AA0DDDD | 9 | |
| | #AASi0DDDD | 11 | |
| | | | |
| 18 | @aaD0sD | 7, 4 | s='0'+C4 |
| | | | |
| 21 | \$aa5c | 6, 4 | |
| | \$aaSsCc5 | 9, 4 | |
| | | | |
| 22 | \$aaCc | 5 | I1=32 |
| | \$aa6c | 5 | |

| Дополнение к подтипу | ASCII-команда | Число байт в команде и ответе (в скобках – число байт, извлекаемых из ответа) | Прим. |
|----------------------|------------------|---|-----------|
| | @aaCc | | I1=64 |
| | \$aaSsCc | 8 | |
| 23 | @aaSAXXXXXXXXXX | 13 | I1=17 |
| | @aaPAXXXXXXXXXX | 13 | I1=16 |
| | \$aa3cXXXXXXXXXX | 13 | I1<16 |
| 26 | #aa0AXX | 7, 2 | I1=0 |
| | #aa0BXX | | I1=1 |
| | #aa0LXX | | I1=2 |
| | #aa0HXX | | I1=3 |
| | #aa0CXX | | I1=4 |
| | @aa | 3, 6 (4) | I1=3 |
| | @aa | 3, 10 (8) | I1=4 |
| | \$aaS<c-8>6 | 6, 12 (8) | I1=8...16 |

Если комментарий (атрибут 80, **CMNT**) канала данного подтипа начинается с ";**xx**" (**xx** – число байт для приема), то далее следует ASCII строка, которая будет передана в COM порт (к этой строке будет добавлена контрольная сумма).

Обмен с i-87040 в WinCon

WinCon: для взаимодействия с платой **87040** (32 DI), установленной в основной крейт контроллера, в ИС нужно создать шаблон 8.7 (**Подтип 8, DI**), задать в нем **Канал=2** и привязать к каналу HEX32 типа INPUT (см. также **Контроллеры XPAC, WinPac, LinPac и WinCon**).

Обмен с i-87041 в WinCon

WinCon: для взаимодействия с платой **87041** (32 DO), установленной в основной крейт контроллера, в ИС нужно создать шаблон 8.108 (**Подтип 8, DO16**), задать в нем **Канал=2** и привязать к каналу HEX32 типа OUTPUT (см. также **Контроллеры XPAC, WinPac, LinPac и WinCon**).

Обмен с i-70xx

Для аналоговых входов **AI**, **AIn**, **AI8** номер канала надо задавать со смещением на 16 (DEC) для значений в HEX формате и со смещением на 64 (DEC) для 12-битных чисел.

Для обмена с модулем **I-7016** используются следующие каналы (см. **Подтип 8**):

- аналоговый выход – **AO**, Канал (CH)=20, Слот (SLOT)=0;
- дискретный выход 1 – **DO/AI**, Канал (CH)=0, Слот (SLOT)=0;
- дискретный выход 2 – **DO/AI**, Канал (CH)=0, Слот (SLOT)=1;
- дискретный вход – **D_I/O/A**, Канал (CH)=0, Слот (SLOT)=0;
- аналоговый вход 1 – **AI**, Канал (CH)=1, Слот (SLOT)=0, тип периода пересчета - F1;
- аналоговый вход 2 – **AI**, Канал (CH)=2, Слот (SLOT)=0, тип периода пересчета - F3;
- чтение счетчика – **ECount**, Канал (CH)=0, Слот (SLOT)=0.

Верхняя и нижняя границы относятся к тому каналу, который считывается в данный момент времени. Поэтому при выборе разных аналоговых каналов необходимо синхронно менять границы командами **SetHAlarm** и **SetLAlarm** (если они разные).

При организации опроса аналоговых входов необходимо учитывать, что в модулях существует задержка на переключение аналогового входа (около 300-350 мс).

В модулях дискретного ввода **I7000** имеются команды считывания и обнуления счетчиков, подключенных к каждому дискретному входу. Считывание значения счетчика осуществляется каналом **Ain** при смещении номера канала на 32 (DEC; канал 3 задается как 35, DEC). Обнуление счетчика осуществляется каналом **Clear_Counter** с таким же смещением номера канала.

Управление WATCHDOG в I-7000:

- Надо создать источник **Распределенные УСО/DCS/DI**, задать ему COM-порт и адрес устройства и установить настройку **Канал=128**.
- Канал, привязанный к этому источнику, будет со своим периодом генерировать широковещательную команду **~**** – обнуление таймера Watchdog.
- Если есть необходимость постоянно контролировать статус устройства и перезапускать Watchdog, надо создать канал, связанный с аналогичным источником с настройкой **Канал=130**, и в атрибут **C4** канала внести величину таймаута для Watchdog (целое число с ценой младшего разряда 0.1 с). Этот канал со своим периодом будет генерировать команду **~aa0** – проверка статуса.

- При получении ответа, не равного 0, атрибут **I1** (97) канала автоматически будет установлен равным 131, и будет генерироваться команда **~aa1** – отмена режима Watchdog, а затем атрибут **I1** (97) канала автоматически будет установлен равным 132, и будет генерироваться команда **~aa31 + C4** – активизация Watchdog с таймаутом, равным **C4**.
- Затем канал снова возвращается к режиму контроля статуса.

Подтипы 9 и 100

Эти подтипы имеют каналы обмена соответственно по MODBUS RTU и MODBUS TCP/IP.

Дополнение к подтипу задает код команды в запросе (номер функции в стандартном протоколе MODBUS RTU):

- **Rout_Byte(1)** – считать байт данных типа **out** (номер дополнения к подтипу – 1);
- **Rin_Byte(2)** – считать байт данных типа **in** (номер дополнения к подтипу – 2);
- **Rout_Word(3)** – считать слово данных типа **out** (номер дополнения к подтипу – 3);
- **Rin_Word(4)** – считать слово данных типа **in** (номер дополнения к подтипу – 4);
- **W_SingleCoil(5)** – передать значение одного дискретного сигнала (номер дополнения к подтипу – 5);
- **W_Word(6)** – передать слово данных (номер дополнения к подтипу – 6);
- **R_Exception(7)** – считать статус контроллера (номер дополнения к подтипу – 7);
- **Rout_Float(3)** – считать 4-байтовую переменную с плавающей точкой типа **out** (номер дополнения к подтипу – 8);
- **Rin_Float(4)** – считать 4-байтовую переменную с плавающей точкой типа **in** (номер дополнения к подтипу – 9);
- **W_Float(16)** – передать 4-байтовую переменную с плавающей точкой типа **out** (номер дополнения к подтипу – 10);
- **W_Word(16)** – передать слово данных (номер дополнения к подтипу – 11);
- **W_Float(16)_wait** – передать 4-байтовую переменную с плавающей точкой типа **out** с последующей задержкой на выполнение других команд (номер дополнения к подтипу – 12);
- **W_Word(6)_wait** – передать слово данных с последующей задержкой на выполнение других команд (номер дополнения к подтипу – 13);
- **W_Byte(15)** – для каналов HEX 16 – передать 1 байт данных, для каналов HEX32 – передать 4 байта (номер дополнения к подтипу – 14);

- **Vzlet1(2,3)** –

эти каналы (FLOAT) служат для копирования архивов из прибора **ВЗЛЕТ МР** (см. **Прибор ‘Взлет МР’**) (номер дополнения к подтипу – 15);

- **W_MaskReg** – функция маскированной записи. Переменная должна быть привязана к каналу HEX32 OUTPUT; два младших байта канала задают And_Mask, а два старших байта – Or_Mask. Номер дополнения к подтипу – 22;

- **R_FIFO_Queue** –

см. **Канал CALL.ChGroupReq**. Номер дополнения к подтипу – 24.

Шаблоны каналов обмена по MODBUS конфигурируются в специальном редакторе – см. **Обмен по MODBUS**.

Подтип 10

Этот подтип имеют каналы класса CALL. Номер дополнения к подтипу соответствует номеру типа вызова (см. **Атрибуты канала класса CALL**).

Подтипы 11 и 12

Эти подтипы имеют каналы вызова драйверов (соответственно t11 и t12 – см. **Драйверы t11 и t12**). Номера дополнений к подтипу соответствуют указанным в файле %Trace Mode%\Drivers_with_Setup\drivers.txt.

Подтип 13

Этот подтип имеют каналы обмена по DDE/NetDDE (см. **Группа ‘DDE-группа’, Редактор переменной DDE, Обмен MPB как DDE-клиент - приложение, Обмен между MPB по NetDDE**).

Подтип 14

Этот подтип имеют в ИС системные переменные TRACE MODE группы СИСТЕМНЫЕ (см. **Группа ‘Диагностика и сервис’ и Редактор системных переменных TRACE MODE**).

Описание этих переменных приведено в разделе **Группа СИСТЕМНЫЕ**.

Подтип 15

Этот подтип имеют в ИС системные переменные TRACE MODE группы ДИАГНОСТИКА (см. **Группа ‘Диагностика и сервис’ и Редактор**

системных переменных TRACE MODE).

Описание этих переменных приведено в разделе **Группа ДИАГНОСТИКА**.

Подтипы 22-26

Эти подтипы имеют каналы T-FACTORY:

22 – M-РЕСУРС (см. **Канал класса M-РЕСУРС**);

23 – M-РЕСУРС (сумматор);

24 – D-РЕСУРС (см. **Канал класса D-РЕСУРС**);

25 – ПЕРСОНАЛ (см. **Канал класса ПЕРСОНАЛ**);

26 – ЕДИНИЦА ОБОРУДОВАНИЯ (см. **Канал класса ЕДИНИЦА ОБОРУДОВАНИЯ**).

Подтип 64

Этот подтип имеют каналы, связанные со встроенными генераторами и моделями (см. **Группа ‘Генераторы’** и **Группа ‘Модели’**). Дополнение к подтипу указывает привязанный источник/приемник:

- 1 – генератор пилообразного сигнала;
- 2 – генератор треугольного сигнала;
- 3 – генератор синусоидального сигнала;
- 4 – генератор «бегущая единица»;
- 5 – генератор «битовый меандр»;
- 6 – генератор случайных чисел;
- 7 – модель объекта первого порядка;
- 8 – модель клапана;
- 9 – модель задвижки;
- 10 – модель мотора;
- 11 – модель резервуара;
- 12 – модель печи;
- 13 – пустой источник.

Подтип 65

Этот подтип имеют каналы, связанные с каналами другого узла по последовательному интерфейсу по протоколу M-LINK (см. также **Связь канал-канал**).

Подтип 66

Этот подтип имеют каналы связи с терминалами и клавиатурами (см. **Группа 'Терминалы'**):

- **Terminal** – обмен с терминалом (дополнение к подтипу – 0);
- **Keyboard** – чтение клавиатуры (дополнение к подтипу – 1).

*Редактор шаблона канала **TERMINAL***

Кроме атрибутов, общих для всех источников/приемников (см. **Редакторы источников (приемников)**), а также **Шаблоны каналов обмена**), в редакторе задаются следующие атрибуты:

- **Номер порта** – номер последовательного порта обмена (0 – COM1, 1 – COM2...);
- **Адрес** – адрес терминала;
- **X-координата** – смещение вывода по X (число знакомест);
- **Y-координата** – смещение вывода по Y (число строк);
- **Параметр** – параметр (см. **Мнемосхемы**);
- **Размер строки** – максимальное число знакомест в строке.

*Редактор шаблона канала **KEYBOARD***

Кроме атрибутов, общих для всех источников/приемников (см. **Редакторы источников (приемников)**), а также **Шаблоны каналов обмена**), в редакторе задаются следующие атрибуты:

- **Номер порта** – номер последовательного порта обмена (0 – COM1, 1 – COM2...);
- **Адрес** – адрес клавиатуры.

Использование данной переменной описано в разделе **Мнемосхемы**.

Подтип 67

Этот подтип имеют каналы, связанные с каналами другого узла по GPRS (см. также **Связь канал-канал**).

Подтип 69

Пока монитор не установил интерфейс взаимодействия с удаленным узлом, этот подтип имеют каналы, для которых в ИС задан интерфейс **Auto** (см. **Связь канал-канал**). После установки интерфейса подтип таких каналов изменяется (например, при установке сетевого обмена – на **подтип 71**).

Подтип 70

Этот подтип имеют каналы, связанные с каналами этого же узла.

Подтип 71

Этот подтип имеют каналы, связанные с каналами другого узла по сети (см. также **Связь канал-канал**). Дополнение к подтипу указывает протокол:

- 0 – UDP
- 20 – TCP

Подтипы 102 и 103

Такой подтип имеют каналы универсального обмена с электросчетчиками (см. **Универсальный механизм обмена с электросчетчиками**).

Подтип 108

Такой подтип имеют каналы обмена по TCP/IP общего назначения.

Подтип 109

Такой подтип имеют каналы обмена по RS общего назначения.

Подтип 254

Этот подтип имеют каналы, которые не пересчитываются монитором. Такая ситуация может возникнуть, например, в случае отказа оборудования, с которым связан канал.

Библиотеки компонентов

Поставляемая пользовательская библиотека компонентов

При установке TRACE MODE 6 в поддиректорию **%TRACE MODE%\LIB** копируется файл пользовательской библиотеки компонентов **tmdeenv.tmul**. Для использования библиотеки файл необходимо скопировать в директорию **%TRACE MODE%**. Библиотека загружается автоматически при загрузке проекта в ИС.

Начиная с релиза 6.08, для использования библиотеки файл необходимо скопировать в директорию C:\Users\All Users\AdAstra\Trace Mode IDE 6\ (в Windows 7) или в C:\Documents and Settings\All Users\Application Data\AdAstra\Trace Mode IDE 6\ (в Windows XP).

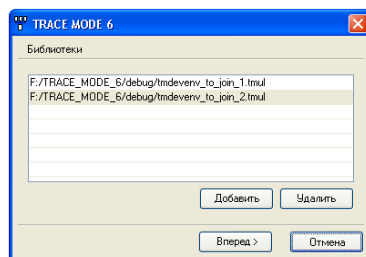
Группа **Ресурсы** библиотеки содержит графические изображения различных объектов (моторов, резервуаров и т.п.), а также обозначения различных устройств на схемах.

Операции с библиотеками компонентов описаны в разделах **Копирование и вставка объекта структуры** и **Объединение пользовательских библиотек компонентов**.

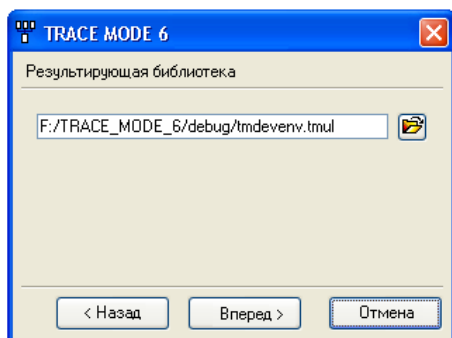
Объединение пользовательских библиотек компонентов

Для объединения нескольких файлов пользовательских библиотек компонентов в один используется утилита **tmde_ulj.exe** или команда ИС (см. **Меню 'Файл' и главная панель инструментов ИС**).

В первом диалоге утилиты с помощью кнопок **Добавить** и **Удалить** нужно сформировать список файлов ***.tmul** (не менее двух), которые требуется объединить:

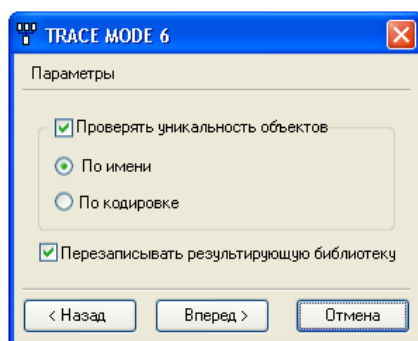


В последующем диалоге необходимо выбрать имеющийся файл *.tmul, в который будет сохранен результат объединения:



При выполнении команды ИС библиотеки всегда объединяются в файл **tmdevenv.tmul** (если файл отсутствует, он создается), поэтому в диалоге отсутствует выбор файла.

В последующем диалоге задаются параметры объединения:



Если флаги **Проверять уникальность объектов** и **Перезаписывать результирующую библиотеку** не установлены, к собственным библиотекам результирующего файла добавляются библиотеки из указанных файлов.

Если флаг **Проверять уникальность объектов** не установлен, а флаг **Перезаписывать результирующую библиотеку** установлен, результирующий файл будет содержать только библиотеки из указанных файлов (собственные библиотеки результирующего файла будут удалены).

Если флаг **Проверять уникальность объектов** установлен и выбрана опция проверки по имени/кодировке, выполняется последовательное сравнение объектов из указанных файлов (по списку) с объектами результирующего файла. Если структурный путь и имя/кодировка анализируемого объекта совпадают с аналогичными параметрами объекта в результирующем файле, анализируемый объект не добавляется в результирующую

щий файл. При несовпадении параметров объект добавляется в результирующий файл.

В данном контексте под объектом понимается структурная группа или компонент.

Сравнение не выполняется для шаблонов и каналов с настроенным свойством **ВЫЗОВ** – такие объекты считаются уникальными и всегда добавляются в результирующий файл.

На каждом шаге процедуры сравнение производится, в том числе, с объектами, добавленными в результирующий файл на предыдущих шагах.

Если при заданной проверке по имени/кодировке установлен флаг **Пере-записывать результирующую библиотеку**, собственные объекты результирующего файла уничтожаются при старте процедуры, и сравнение с ними не производится.

AdAstra Research Group, Ltd

107076 Россия, Москва, а/я 38

Тел. (495) 771-71-74

Факс (495) 518-98-46

E-mail: adastra@adastra.ru

<http://www.adastra.ru>