Пакет поддержки “РИТМ”

Комплекс полунатурного моделирования “РИТМ” и пакет поддержки “РИТМ” в Simulink представляют собой программно-аппаратное решение “под ключ”, предоставляющее возможности тестирования в реальном времени.

Дополнительная информация о КПМ “РИТМ” и пакете поддержки “РИТМ” доступна в разделе Введение.

Содержание

Содержание 2
Пакет поддержки “РИТМ” 27
Установка 28
  Содержание 28
Предварительные требования 28
Установка пакета поддержки “РИТМ” 28
Настройка соединения между хост-компьютером и машиной “РИТМ” 28
  Настройка хоста 28
  Настройка “РИТМ” 29
Обновление пакета поддержки “РИТМ” 29
Обновление прошивки “РИТМ” 29
Введение 30
  Содержание 30
Комплекс полунатурного моделирования “РИТМ” 30
ОСРВ “РИТМ.Реальное время” 30
  Технические возможности ОСРВ “РИТМ.Реальное время” 30
  Техническая информация 31
Пакет поддержки “РИТМ” 31
  Технические возможности пакета поддержки “РИТМ” 31
Общая информация 31
РИТМ.Управление машинами 33
  Содержание 33
Назначение программы 33
Запуск 33
Описание программы 33
Главное меню 34
  Обновление 34
  Справка 35
Настройка подключения 35
Работа с машиной 35
  Управление подключением к машине 36
  Управление приложениями 36
  Настройки машины 37
    Настройка сетевого интерфейса хост-таргет на машине 37
    Настройка сетевых интерфейсов 38
Начало работы с пакетом поддержки “РИТМ” 39
  Содержание 39
Пример простой модели 39
Пример работы с модулями ввода-вывода 43
Настройки модели 44
  Содержание 44
Панель настроек модели 44
Описание настроек 44
  Target computer for this model 44
  Enable overrun detection 45
  Action on overrun 45
  Terminate model after this number of overruns 45
  Enable Task Execution Time calculation 45
  Enable execution profiling 46
  Number of profiling points to log 46
См. также 46
Режимы выполнения модели 47

Содержание 47
Режим независимого выполнения 47
Область применения 47
Режим интерактивного выполнения 47
Область применения 48
Переполнение шага расчета 49
Содержание 49
Что такое переполнение шага расчета? 49
Методы диагностики 50
Обработка переполнения шага расчета в КПМ “РИТМ” 50
Пример 50
Принцип работы 50
Режим уведомления пользователя 52
Режим останова выполнения модели 52
Рекомендации производителя 53
Профилирование выполнения 54
Содержание 54
Общие сведения 54
Получение результатов профилирования 54
Просмотр и интерпретация результатов 54
Дополнительные возможности 57
Пример работы 59
Многозадачное выполнение 60
Содержание 60
Введение 60
Описание технологии 60
Информация в документации Simulink 60
Дополнительная информация 60
Пример многозадачного выполнения 61
Ограничения 61
Примеры и демонстрации многозадачного выполнения 62
Содержание 62
Предварительные требования 62
Разбиение модели на задачи 62
Включение Concurrent Execution 62
Настройка Concurrent Execution 63
Запуск модели 65
Интерфейсы пользователя в App Designer 66
Содержание 66
Предварительные требования 66
Запуск и остановка модели 66
Настройка параметров модели 67
Если параметры заданы в виде чисел в блоках Simulink 67
Если параметры заданы в виде названий переменных 67
Отображение сигналов модели 67
Пример графического интерфейса пользователя 68
Запуск нейросетей на РИТМе 70
Содержание 70
Предварительные требования 70
Общая информация 70
Пример генерации независимого C/C++ кода 70
Пример генерации C++ кода на oneDNN 70
Функции и API РИТМ 71

Вспомогательная функция ritm.util.getLog 72
Вспомогательная функция ritm.util.setupEthernet 73
Вспомогательная функция ritm.util.setupTargetGui 74
Вспомогательная функция ritm.util.getMatFile 77
Вспомогательная функция ritm.util.setTime 78
Вспомогательная функция ritm.util.isRunning 79
Вспомогательная функция ritm.util.isConnected 80
Программное управление моделями на РИТМ 81
Вспомогательная функция ritm.util.buildModel 82
Вспомогательная функция ritm.util.runModel 83
Вспомогательная функция ritm.util.stopModel 84
Блоки для работы с графическим интерфейсом 85
Содержание 85
Библиотека блоков РИТМ График 85
Библиотека блоков РИТМ График 86
Содержание 86
Описание 86
Библиотека блоков 86
Блок РИТМ График 86
Маска блока 86
Входы блока 86
Параметры блока 86
Примечания об использовании 87
Примеры и демонстрации РИТМ График 88
Блоки для работы с ОСРВ “РИТМ” 89
Содержание 89
Библиотека блоков РИТМ Время выполнения 89
Библиотека блоков РИТМ Unix Время 89
Библиотека блоков РИТМ Задержка 89
Библиотека блоков РИТМ Задача ОСРВ 89
Библиотека блоков сериализации шин Simulink 89
Библиотека блоков РИТМ Время выполнения 90
Содержание 90
Описание 90
Библиотека блоков 90
Блок РИТМ Время выполнения 90
Маска блока 90
Выходы блока 90
Параметры блока 90
См. также 90
Примеры и демонстрации РИТМ Время выполнения 92
Библиотека блоков РИТМ Unix Время 93
Содержание 93
Описание 93
Библиотека блоков 93
Блок РИТМ Unix Время 93
Маска блока 93
Выходы блока 93
Параметры блока 93
Примеры и демонстрации РИТМ Unix Время 94
Библиотека блоков РИТМ Задержка 95
Содержание 95
Библиотека блоков 95

Блок РИТМ Задержка 95
Маска блока 95
Параметры блока 95
Примеры и демонстрации РИТМ Задержка 96
Библиотека блоков РИТМ Задача ОСРВ 97
Содержание 97
Описание 97
Область применения 97
Библиотека блоков 97
Блок РИТМ Задача ОСРВ 97
Маска блока 97
Входы блока 98
Выходы блока 98
Параметры блока 98
Указания по моделированию 98
Примеры и демонстрации РИТМ Задача ОСРВ 101
Содержание 101
Прием и передача данных по протоколу UDP 101
Конвейеризация расчетов 101
Замечания по работе 102
Библиотека блоков сериализации шин Simulink 105
Содержание 105
Описание 105
Область применения 105
Библиотека блоков 105
Блок Сериализация шины 106
Маска блока 106
Входы блока 106
Выходы блока 106
Параметры блока 106
Блок Десериализация шины 106
Маска блока 106
Входы блока 106
Выходы блока 106
Параметры блока 107
Пример использования блоков сериализации шин Simulink 108
Библиотека блоков RITM From File 109
Содержание 109
Библиотека блоков 109
Блок RITM From File 109
Маска блока 109
Выходы блока 111
Параметры блока 111
Примеры и демонстрации RITM From File 113
Модули общего назначения 114
Содержание 114
Модуль RITMeX GP-LC-45 114
Модуль RITMeX GP-ID-4X 114
Модуль RITMeX GP-AD-24 114
Модуль RITMeX GP-ID-24 114
Модуль RITMeX GP-ID-34 114
Модуль RITMeX GP-EL-64 115
Модуль RITMeX GP-LC-45 116
Содержание 116

Описание 116
Библиотека блоков 116
Аналоговые входы 116
Характеристики АЦП 116
Блок АЦП в Simulink 116
Маска блока 116
Параметры блока 117
Выходы блока 117
Аналоговые выходы 117
Характеристики ЦАП 117
Блок ЦАП в Simulink 117
Маска блока 117
Параметры блока 117
Входы блока 117
Цифровые входы 117
Характеристики цифровых входов 117
Блок цифровых входов в Simulink 118
Маска блока 118
Параметры блока 118
Выходы блока 118
Цифровые выходы 118
Характеристики цифровых выходов 118
Блок цифровых выходов в Simulink 118
Маска блока 118
Параметры блока 118
Входы блока 118
Генерация высокочастотных периодических сигналов 118
Блок генерации синусоидального сигнала в Simulink 118
Маска блока 119
Параметры блока 119
Входы блока 119
Блок генерации ШИМ в Simulink 119
Маска блока 119
Параметры блока 119
Входы блока 120
Блок генерации пилообразного сигнала в Simulink 120
Маска блока 120
Параметры блока 120
Входы блока 120
Блок генерации произвольного периодического сигнала в Simulink 120
Маска блока 120
Параметры блока 121
Входы блока 121
Блок генерации ШИМ на дискретных выходах в Simulink 121
Маска блока 121
Параметры блока 121
Входы блока 121
Примеры и демонстрации GP-LC-45 123
Назначение пинов GP-LC-45 126
Содержание 126
Аналоговые входы и выходы 126
Цифровые входы и выходы 129
Модули RITMeX GP-ID-45/GP-ID-46 131
Содержание 131
Описание 131

Библиотека блоков 131
Аналоговые выходы 131
Характеристики ЦАП 131
Блок “Аналоговый выход” в Simulink 131
Маска блока 132
Параметры блока 132
Вход блока 132
Цифровые входы 132
Характеристики цифровых входов 132
Блок “Цифровые входы” в Simulink 132
Маска блока 132
Параметры блока 133
Цифровые выходы 133
Характеристики цифровых выходов 133
Блок “Цифровые выходы” в Simulink 133
Маска блока 133
Параметры блока 134
Примеры и демонстрации GP-ID-4X 135
Назначение пинов GP-ID-4X 136
Содержание 136
Аналоговые выходы 136
Схема подключения аналогового выхода как источника тока 136
Схема подключения аналогового выхода как источника напряжения 137
Цифровые выходы 137
Цифровые входы 138
Модуль RITMeX GP-AD-24 140
Содержание 140
Описание 140
Библиотека блоков 140
Аналоговые выходы 141
Характеристики ЦАП 141
Блок “Аналоговый выход” многоканальный в Simulink 141
Маска блока 142
Параметры блока 142
Вход блока 142
Блок “Аналоговый выход” одноканальный в Simulink 142
Маска блока 142
Параметры блока 143
Вход блока 143
Примеры и демонстрации GP-AD-24 144
Назначение пинов GP-AD-24 145
Содержание 145
Аналоговые выходы 145
Схема подключения аналогового выхода в режиме источника напряжения 146
Схема подключения аналогового выхода в режиме источника тока 146
Модуль RITMeX GP-ID-24 148
Содержание 148
Описание 148
Характеристики модуля 148
Библиотека блоков 148
Блок “Цифровые входы” в Simulink 148
Маска блока 149
Параметры блока 149
Блок “Цифровые выходы” в Simulink 149
Маска блока 149

Параметры блока 150
Примеры и демонстрации GP-ID-24 151
Назначение пинов GP-ID-24 152
Содержание 152
Цифровые входы и цифровые выходы 152
Модуль RITMeX GP-ID-34 153
Содержание 153
Описание 153
Библиотека блоков 153
Цифровые входы 153
Характеристики цифровых входов 153
Блок “Цифровые входы” в Simulink 153
Маска блока 154
Параметры блока 154
Цифровые выходы 154
Характеристики цифровых выходов 154
Блок “Цифровые выходы” в Simulink 154
Маска блока 154
Параметры блока. 155
Примеры и демонстрации GP-ID-34 156
Назначение пинов GP-ID-34 157
Содержание 157
Цифровые входы и цифровые выходы 157
Схема подключения входного сигнала 158
Схема подключения выходного сигнала 158
Модуль RITMeX GP-EL-64 160
Содержание 160
Описание 160
Библиотека блоков 160
Цифровые входы 160
Характеристики цифровых входов с открытым коллектором 160
Блок “Цифровые входы открытый коллектор” в Simulink 160
Маска блока 161
Параметры блока 161
Характеристики цифровых входов TTL уровня 161
Блок “Цифровые входы TTL” в Simulink 161
Маска блока 161
Параметры блока 162
Цифровые выходы 162
Характеристики цифровых выходов с открытым коллектором 162
Блок “Цифровые выходы открытый коллектор” в Simulink 162
Маска блока 162
Параметры блока. 163
Характеристики цифровых выходов TTL уровня 163
Блок “Цифровые выходы открытый коллектор” в Simulink 163
Маска блока 163
Параметры блока. 163
Примеры и демонстрации GP-EL-64 164
Назначение пинов GP-EL-64 165
Содержание 165
Цифровые входы и цифровые выходы 165
Схема каналов с открытым коллектором 165
Схема каналов TTL уровня 166
Модули RS-232/RS-422/RS-485 167
Содержание 167

Модули RITMeX RS-MO-45 и RITMeX RS-MO-35 167
Библиотека блоков RITM-COM 167
Модули RITMeX RS-MO-45 и RITMeX RS-MO-35 168
Содержание 168
Описание 168
Библиотека блоков 168
Характеристики модуля 169
Блок Последовательный порт 169
Маска блока 169
Параметры блока 170
Блок Передатчик 170
Маска блока 171
Входы блока 171
Параметры блока 171
Блок Приёмник 171
Маска блока 171
Выходы блока 172
Параметры блока 172
Примеры и демонстрации RS-MO-45 и RS-MO-35 173
Назначение пинов RS-MO-45 и RS-MO-35 174
Библиотека блоков RITM-COM 175
Содержание 175
Описание 175
Библиотека блоков 175
Блок Последовательный порт 176
Маска блока 176
Параметры блока 177
Блок Передатчик 177
Маска блока 177
Входы блока 178
Параметры блока 178
Блок Приёмник 178
Маска блока 178
Выходы блока 178
Параметры блока 179
Примеры и демонстрации RITM-COM 180
Блоки поддержки Ethernet 181
Содержание 181
Библиотека UDP 181
Библиотека сырого Ethernet 181
Блоки и API для работы с PTP на РИТМ 181
Библиотека блоков RITM-UDP 182
Содержание 182
Описание 182
Библиотека блоков 182
Блок UDP-Передатчик 182
Маска блока 182
Входы блока 183
Параметры блока 183
Блок UDP-Приёмник 183
Маска блока 183
Выходы блока 183
Параметры блока 183
Примеры и демонстрации RITM-UDP 184

Библиотека блоков RITM-TCP 185
Содержание 185
Описание 185
Библиотека блоков 185
Блок TCP Сервер 186
Маска блока 186
Выходы блока 187
Параметры блока 187
Блок TCP Клиент 187
Маска блока 187
Выходы блока 187
Параметры блока 188
Блок TCP Отправка 188
Маска блока 188
Входы блока 188
Выходы блока 188
Параметры блока 188
Блок TCP Приём 188
Маска блока 189
Входы блока 189
Выходы блока 189
Параметры блока 189
Примеры и демонстрации RITM-TCP 190
Библиотека блоков RITM Raw Ethernet 193
Содержание 193
Описание 193
Библиотека блоков 193
Блок Raw Ethernet Передатчик 194
Маска блока 194
Входы блока 194
Параметры блока 194
Блок Raw Ethernet Передатчик Многопакетный 194
Маска блока 195
Входы блока 195
Параметры блока 195
Блок Raw Ethernet Приёмник 195
Маска блока 195
Выходы блока 195
Параметры блока 195
Блок Raw Ethernet MAC 196
Выходы блока 196
Параметры блока 196
Блок Raw Ethernet Упаковка пакета 196
Входы блока 196
Выходы блока 196
Параметры блока 196
Блок Raw Ethernet Распаковка пакета 196
Входы блока 196
Выходы блока 196
Параметры блока 197
Фильтр сообщений в блоке Raw Ethernet Приёмник 197
Примеры и демонстрации RITM-Raw-Ethernet 198
Блоки и API для работы с PTP на РИТМ 201
Библиотека блоков РИТМ PTP 202
Содержание 202

Библиотека блоков 202
Блок РИТМ PTP 202
Маска блока 202
Параметры блока 202
Выходы блока 202
Набор функции для настройки PTP соединения 204
Содержание 204
Функция PTP.setupEthPtp 204
Функция PTP.startPtp 204
Функция PTP.stopPtp 205
Функция PTP.resetSettings 205
Функция PTP.getState 205
Функция PTP.getSettings 206
Сетевые карты для КПМ РИТМ 208
Модули 1 Гбит Ethernet 209
Содержание 209
Сетевой адаптер RITMeX ET-IN-55 209
Сетевой адаптер RITMeX ET-IN-65 209
Поддерживаемые блоки 209
Модуль RITMeX ET-MX-X5 210
Содержание 210
Описание 210
Поддерживаемые блоки 210
Блоки поддержки RDMA 211
Содержание 211
Библиотека RDMA TCP 211
Библиотека RDMA Буфер 211
Библиотека блоков RITM-RDMA-TCP 212
Содержание 212
Описание 212
Библиотека блоков 212
Блок RDMA TCP Сервер 213
Маска блока 213
Выходы блока 213
Параметры блока 213
Блок RDMA TCP Клиент 214
Маска блока 214
Выходы блока 214
Параметры блока 214
Блок RDMA TCP Отправка 214
Маска блока 214
Входы блока 215
Выходы блока 215
Параметры блока 215
Блок RDMA TCP Приём 215
Маска блока 215
Входы блока 216
Выходы блока 216
Параметры блока 216
Примеры и демонстрации RITM-RDMA-TCP 217
Библиотека блоков RITM-RDMA-BUF 218
Содержание 218
Описание 218
Библиотека блоков 218

Блок RDMA BUF Входной буфер 219
Маска блока 219
Выходы блока 219
Параметры блока 219
Блок RDMA BUF Выходной буфер 219
Маска блока 219
Входы блока 220
Выходы блока 220
Параметры блока 220
Тип ritm_rdma_buf_status 220
Примеры и демонстрации RITM-RDMA-BUF 221
Блоки поддержки Modbus 222
Содержание 222
Общие сведения 222
Особенности работы 222
Поддерживаемые функции протокола Modbus 222
Блоки для работы с Modbus 222
Руководство по использованию библиотеки 222
Блок настройки соединения Modbus TCP 224
Содержание 224
Описание 224
Маска блока 224
Параметры блока 224
Блок настройки соединения Modbus RTU 226
Содержание 226
Описание 226
Маска блока 226
Параметры блока 226
Блоки приема данных 228
Содержание 228
Блок Modbus Read Registers 228
Описание 228
Маска блока 228
Параметры блока 229
Выходы блока 229
Блок Modbus Read Coils 229
Описание 229
Маска блока 229
Параметры блока 230
Выходы блока 230
Блоки отправки данных 231
Содержание 231
Блок Modbus Write Registers 231
Описание 232
Маска блока 232
Параметры блока 232
Входы блока 232
Выходы блока 232
Блок Modbus Write Coils 232
Описание 232
Маска блока 232
Параметры блока 233
Входы блока 233
Выходы блока 233

Блок Modbus Write Register 233
Описание 233
Маска блока 233
Параметры блока 234
Входы блока 234
Выходы блока 234
Блок Modbus Write Coil 234
Описание 234
Маска блока 234
Параметры блока 235
Входы блока 235
Выходы блока 235
Блоки для обработки данных 236
Содержание 236
Блок Modbus Bytes to single 236
Описание 236
Маска блока 236
Параметры блока 236
Блок Modbus Single to bytes 237
Описание 237
Маска блока 237
Параметры блока 237
Работа с библиотекой блоков Modbus TCP 238
Содержание 238
Рекомендуемое ПО для ознакомления с библиотекой 238
Особенности работы с симулятором ModRSsim2 238
Особенности работы с симулятором ModbusPal 238
Особенности работы библиотеки 238
Карта смещений 238
Начало работы 238
Минимальное необходимое ПО 238
Простой обмен данными 239
Рекомендуемые приемы моделирования 241
Запись множественных значений 241
Работа с библиотекой блоков Modbus RTU 242
Содержание 242
Модель 242
Моделирование 243
Библиотека источников питания 245
Содержание 245
Общие сведения 245
Особенности работы 245
Блоки для работы с источниками питания 245
Руководство по использованию библиотеки 245
Блок управления источником питания MAISHENG WSD-6050 246
Содержание 246
Описание 246
Маска блока 246
Параметры блока 246
Входы блока 247
Выходы блока 247
Работа с блоком управления источника питания MAISHENG WSD-6050 248
Содержание 248
Включение дистанционного управления 248

Демонстрация работы 248
Моделирование 249
Блок управления источником питания OWON OWP3006H 251
Содержание 251
Описание 251
Маска блока 251
Параметры блока 252
Входы блока 252
Выходы блока 252
Работа с блоком управления источника питания OWON OWP3006H 253
Содержание 253
Включение дистанционного управления 253
Демонстрация работы 253
Моделирование 254
Библиотека контроллеров управления 256
Содержание 256
Блок для работы с рулевым управлением Logitech G29 256
Блоки для работы джойстиком и блоком РУД Thrustmaster Hotas Warthog 256
Блок для работы с рулевым управлением Logitech G29 257
Содержание 257
Библиотека блоков 257
Блок RITM G29 WHEEL 257
Маска блока 257
Выход блока 257
Параметры блока 258
Пример работы блока 258
Блоки для работы джойстиком и блоком РУД Thrustmaster Hotas Warthog 259
Содержание 259
Библиотека блоков 259
Блок RITM TM JOYSTICK 259
Маска блока 259
Выход блока 260
Параметры блока 260
Блок RITM TM TROTTLE 260
Маска блока 260
Выход блока 260
Параметры блока 261
Пример работы блока 261
Модули поддержки MIL-STD-1553 262
Содержание 262
Модуль RITMeX MI-EL-45 262
Модули RITMeX MI-EL-X5 263
Содержание 263
Описание 263
Библиотека блоков 263
Настройка канала 264
Блок “Устройство МКИО: настройка” в Simulink 264
Маска блока 264
Параметры блока 264
Контроллер шины 265
Блок “МКИО контроллер шины: передача сообщения” в Simulink 265
Маска блока 265
Параметры блока 265
Входы блока 265

Выходы блока 265
Блок “МКИО контроллер шины: получение сообщения” в Simulink 265
Маска блока 266
Параметры блока 266
Входы блока 266
Выходы блока 266
Блок “МКИО контроллер шины: передача сообщения с ОУ1 на ОУ2” в Simulink 266
Маска блока 266
Параметры блока 267
Входы блока 267
Выходы блока 267
Расшифровка статуса ошибки в блоках контроллера шины 267
Оконечное устройство 267
Блок “МКИО оконечное устройство: запись в подадрес” в Simulink 267
Маска блока 267
Параметры блока 268
Входы блока 268
Блок “МКИО оконечное устройство: чтение из подадреса” в Simulink 268
Маска блока 268
Параметры блока 268
Входы блока 268
Выходы блока 269
Блок “МКИО оконечное устройство: получить событие” в Simulink 269
Маска блока 269
Параметры блока 269
Выходы блока 269
Монитор шины 269
Блок “МКИО монитор шины: получить сообщения” в Simulink 269
Маска блока 269
Параметры блока 270
Входы блока 270
Выходы блока 270
Блок “МКИО монитор шины: декодировать сообщения” в Simulink 270
Вид блока 270
Входы блока 271
Выходы блока 271
Примеры и демонстрации RITMeX MI-EL-45/MI-EL-55 273
Назначение пинов MI-EL-45 274
Содержание 274
Внешние разъемы 274
Назначение контактов 274
Тестовый жгут проводов 274
Тип подключения 275
Внешние разъемы 275
Тестовый жгут проводов 277
Модули поддержки CAN 278
Содержание 278
Библиотека блоков Simulink для работы с CAN 278
Примеры работы с библиотекой блоков CAN 278
Модуль RITMeX CA-MA-45 278
Модуль RITMeX CA-MA-55 278
Модули RITMeX CA-ID-45 и RITMeX CA-ID-55 278
Библиотека блоков RITM-CAN 279
Содержание 279
Описание 279

Библиотека блоков 279
Блок CAN Настройка 279
Маска блока 279
Параметры блока 279
Блок CAN Статус 280
Маска блока 280
Параметры блока 280
Входы блока 280
Блок CAN Передатчик 280
Маска блока 280
Входы блока 280
Параметры блока 281
Блок CAN Приёмник 281
Маска блока 281
Выходы блока 281
Параметры блока 281
Фильтр сообщений в блоке CAN Приёмник 281
Пример работы фильтра 282
Блок CAN Перезапуск 282
Маска блока 282
Параметры блока 282
Шаг расчёта 282
Примеры и демонстрации работы с CAN блоками КПМ “РИТМ”. 283
Назначение пинов CA-MA-45 285
Назначение пинов CA-MA-55 286
Назначение пинов CA-ID-45 и CA-ID-55 287
Модули поддержки ARINC 429 288
Содержание 288
Блоки кодирования/декодирования слов ПК ARINC 429 288
Модуль RITMeX AR-EL-45 288
Модуль RITMeX AR-EL-55 288
Блоки кодирования/декодирования слов ПК ARINC 429 289
Содержание 289
Блок ARINC 429 Раскодировать слово ПК 289
Маска блока 289
Входы блока 289
Выходы блока 289
Блок ARINC 429 Закодировать слово ПК 289
Маска блока 289
Входы блока 290
Выходы блока 290
Модуль RITMeX AR-EL-45 291
Содержание 291
Описание 291
Библиотека блоков 291
Блок ARINC 429 Настройка 291
Маска блока 291
Параметры блока 292
Блок ARINC 429 Передатчик 292
Маска блока 292
Параметры блока 293
Входы блока 293
Блок ARINC 429 Приёмник 293
Маска блока 293
Параметры блока 294

Входы блока 294
Выходы блока 294
Примеры и демонстрации AR-EL-45 295
Назначение пинов AR-EL-45 296
Модуль RITMeX AR-EL-55 297
Содержание 297
Описание 297
Библиотека блоков 297
Блок ARINC 429 Настройка 297
Маска блока 297
Параметры блока 298
Блок ARINC 429 Передатчик 298
Маска блока 298
Параметры блока 299
Входы блока 299
Блок ARINC 429 Приёмник 299
Маска блока 299
Параметры блока 300
Выходы блока 300
Примеры и демонстрации AR-EL-55 301
Назначение пинов AR-EL-55 302
Запуск алгоритмов на ПЛИС 303
Содержание 303
Общие сведения о запуске алгоритмов на ПЛИС 303
Подключение цифровых входов-выходов к алгоритму на ПЛИС 303
Подключение аналоговых входов-выходов к алгоритму на ПЛИС 303
Подключение аналоговых входов-выходов RF1 к алгоритму на ПЛИС 303
Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС 303
Моделирование силовой электроники в реальном времени на ПЛИС 303
Общие сведения о запуске алгоритмов на ПЛИС 304
Содержание 304
Общая информация 304
Как это работает 304
С точки зрения аппаратуры 305
С точки зрения пользователя 305
С точки зрения программного обеспечения 305
Установка 305
Дополнительная информация 306
Подключение цифровых входов-выходов к алгоритму на ПЛИС 307
Содержание 307
Требования к аппаратному обеспечению 307
Требования к программному обеспечению 307
Общая структура системы 307
Пример использования 307
Демо модель 307
Генерация прошивки для ПЛИС 309
Назначение интерфейсов DUT к интерфейсам на ПЛИС 309
Генерация HDL кода и интеграция с проектом Xilinx Vivado 311
Обзор программной модели 313
Подключение аналоговых входов-выходов к алгоритму на ПЛИС 316
Содержание 316
Требования к аппаратному обеспечению 316
Требования к программному обеспечению 316

Общая структура системы 316
Пример использования 316
Демо модель 316
Генерация прошивки для ПЛИС 318
Назначение интерфейсов DUT к интерфейсам на ПЛИС 318
Генерация HDL кода и интеграция с проектом Xilinx Vivado 323
Обзор программной модели 325
Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС 328
Содержание 328
Модели для цифровых входов-выходов 328
Демо модель default 328
Демо модель pwm_cap 329
Демо модель gpiod64_l1 329
Демо модель gpiod64_l1l2 329
Модели для аналоговых входов-выходов 330
Демо модель analogio_gpio 330
Демо модель analogio_gpiod64 330
Модель передачи типов данных для входов-выходов подсистемы DUT 331
Модели для прямого доступа к памяти DMA 331
Демо модель dma 331
Модели смешанного типа 332
Демо модель dma/аналог/цифра 332
Многочастотные модели 332
Демо модель multirate_pid 332
Моделирование силовой электроники в реальном времени на ПЛИС 333
Содержание 333
Требования к аппаратному обеспечению 333
Требования к программному обеспечению 333
Общие положения 333
Создание и подготовка Simscape-модели 333
Генерация модели в пространстве состояний 335
Этапы перевода модели Simscape в модель реализации HDL: 335
Работа модели реализации HDL 338
Валидация модели реализации HDL 339
Генерация HDL-кода 339
Настраиваемые модули на базе ПЛИС 344
Модуль аналоговых выходов 346
Содержание 346
Описание 346
Библиотека блоков 346
Маска блока 346
Параметры блока 346
Входы блока 347
Примеры и демонстрации BA-IS-DAC/BA-IS-ADC 348
Назначение пинов BA-IS-DAC 349
Модуль аналоговых входов 350
Содержание 350
Описание 350
Библиотека блоков 350
Маска блока 350
Параметры блока 351
Выходы блока 351
Назначение пинов BA-IS-ADC 352

Модуль генератор частоты на аналоговые выходы 353
Содержание 353
Описание 353
Библиотека блоков 353
Блок генератор частоты на аналоговые выходы 353
Маска блока 353
Параметры блока 354
Входы блока 354
Примеры и демонстрации BA-IS-GEN-FREQ-DAC 355
Модуль генератор синуса на аналоговые выходы 356
Содержание 356
Описание 356
Библиотека блоков 356
Блок генератор синуса на аналоговые выходы 356
Маска блока 356
Параметры блока 357
Входы блока 357
Примеры и демонстрации BA-IS-GEN-SINE-DAC 358
Модуль усреднения с аналоговых входов 359
Содержание 359
Описание 359
Библиотека блоков 359
Блок BA-IS-ADC-RMS-SMA 359
Маска блока 359
Параметры блока 360
Входы блока 360
Выходы блока 360
Примеры и демонстрации BA-IS-ADC-RMS-SMA 361
Назначение пинов BA-IS-ADC-RMS-SMA 362
Модули на основе Fibre Channel 363
Содержание 363
Модуль аналоговых выходов через протокол Аврора 363
Модуль аналоговых входов-выходов через протокол Аврора 363
Модули приемопередачи через интерфейс Aurora 8b10b 363
Модуль Fibre Channel стандарта FC-AE-ASM 363
Модуль аналоговых выходов через протокол Аврора 364
Содержание 364
Описание 364
Требования к оборудованию 364
Библиотека блоков 364
Блок Аврора ЦАП 364
Маска блока 364
Требования к блоку 365
Параметры блока 365
Входы блока 365
Примеры и демонстрации BA-IS-AURORA-DAC 366
Назначение пинов BA-IS-AURORA-DAC 367
Модули приемопередачи через интерфейс Aurora 8b10b 368
Содержание 368
Описание 368
Требования к оборудованию 368
Библиотека блоков 368
Блок Aurora 8b10b TX 368
Маска блока 369

Параметры блока 369
Вход блока 369
Блок Aurora 8b10b RX 369
Маска блока 369
Параметры блока 370
Выход блока 370
Примечание: 370
Примеры и демонстрации BA-IS-AURORA-TX и BA-IS-AURORA-RX 372
Назначение пинов BA-IS-AURORA-TX и BA-IS-AURORA-RX 373
Модуль Fibre Channel стандарта FC-AE-ASM 374
Содержание 374
Описание 374
Примеры и демонстрации BA-IS-DMA-TX/BA-IS-DMA-RX 375
Назначение пинов Fibre Channel 376
Модули прямого доступа к памяти DMA 377
Содержание 377
DMA для записи данных в ПЛИС 377
DMA для чтения данных из ПЛИС 377
DMA для записи данных в ПЛИС 378
Содержание 378
Описание 378
Библиотека блоков 378
Блок запись DMA 378
Маска блока 378
Требования к блоку 379
Параметры блока 379
Входы блока 379
DMA для чтения данных из ПЛИС 380
Содержание 380
Описание 380
Библиотека блоков 380
Блок чтение DMA 380
Маска блока 380
Требования к блоку 381
Параметры блока 381
Входы блока 381
ПРИМЕЧАНИЕ 1: 381
ПРИМЕЧАНИЕ 2: 382
Модуль Цифровых входов-выходов 383
Содержание 383
Описание 383
Библиотека блоков 383
Блок Цифровых входов 383
Маска блока 383
Параметры блока 384
Выходы блока 384
Блок Цифровых выходов 384
Маска блока 384
Параметры блока 384
Входы блока 385
Примеры и демонстрации BA-IS-GPIO 386
Назначение пинов BA-IS-GPIO 387
Модуль Генерации ШИМ 388
Содержание 388
Описание 388

Библиотека блоков 388
Блок Генерации ШИМ 388
Маска блока 388
Параметры блока 388
Входы блока 389
Примеры и демонстрации BA-IS-PWM/BA-IS-CAP 390
Назначение пинов BA-IS-PWM 391
Модуль Захвата ШИМ 392
Содержание 392
Описание 392
Библиотека блоков 392
Блок Захвата ШИМ 392
Маска блока 392
Параметры блока 392
Выходы блока 393
Назначение пинов BA-IS-CAP 394
Модуль SPI 395
Содержание 395
Описание 395
Библиотека блоков 395
Блок Работа с SPI 395
Маска блока 395
Параметры блока 395
Входы блока 396
Выходы блока 396
Фаза и полярность 396
Примеры и демонстрации BA-IS-SPI 397
Назначение пинов BA-IS-SPI 398
Модуль I2C 399
Содержание 399
Описание 399
Библиотека блоков 399
Блок Запись в I2C 399
Маска Блока 399
Параметры блока 400
Входы блока 400
Блок Чтение с I2C 400
Маска Блока 400
Параметры блока 401
Выходы блока 401
Примеры и демонстрации BA-IS-I2C 402
Назначение пинов BA-IS-I2C 403
Модуль Квадратурного декодера и Квадратурного энкодера 404
Содержание 404
Описание 404
Заметки об использовании 404
Библиотека блоков 405
Блок Квадратурный декодер 405
Маска блока 405
Параметры блока 406
Выходы блока 406
Блок Квадратурный энкодер 406
Маска блока 406
Параметры блока 407
Входы блока 407

Примеры и демонстрации BA-IS-QAD/BA-IS-QAE 408
Назначение пинов BA-IS-QAD/BA-IS-QAE 409
Модуль приемопередатчика UART 410
Содержание 410
Описание 410
Библиотека блоков 410
Блок Настройка UART 410
Маска блока 410
Параметры блока 411
Блок Передача UART 411
Маска блока 411
Параметры блока 412
Входы блока 412
Блок Приём UART 412
Маска блока 412
Параметры блока 413
Выходы блока 413
Замечания об использовании и рекомендации 413
Примеры и демонстрации BA-IS-UART 415
Назначение пинов BA-IS-UART 416
Модуль захвата ШИМ через UART 417
Содержание 417
Описание 417
Библиотека блоков 417
Блок Захват ШИМ через UART 417
Маска блока 418
Параметры блока 418
Выходы блока 418
Детали реализации 418
Примеры и демонстрации BA-IS-CAPAURT 420
Назначение пинов BA-IS-CAPAURT 421
Модули FMC-RF1 аналоговых входов/выходов 422
Содержание 422
Модуль FMC-RF1 АЦП в коде 422
Модуль FMC-RF1 АЦП в вольтах 422
Модуль FMC-RF1 DUT АЦП инициализация 422
Модуль FMC-RF1 ЦАП одноканальный 422
Модуль FMC-RF1 ЦАП двухканальный 422
Модуль FMC-RF1 DUT ЦАП инициализация 422
Модуль FMC-RF1 АЦП в коде 423
Содержание 423
Описание 423
Библиотека блоков 423
Маска блока 423
Параметры блока 423
Выходы блока 424
Примеры и демонстрации FMC-RF1 АЦП 425
Назначение пинов FMC-RF1 АЦП 426
Модуль FMC-RF1 АЦП в вольтах 427
Содержание 427
Описание 427
Библиотека блоков 427
Маска блока 427
Параметры блока 427

Выходы блока 428
Модуль FMC-RF1 DUT АЦП инициализация 429
Содержание 429
Описание 429
Библиотека блоков 429
Маска блока 429
Параметры блока 429
Модуль FMC-RF1 ЦАП одноканальный 431
Содержание 431
Описание 431
Библиотека блоков 431
Маска блока 431
Параметры блока 431
Входы блока 432
Примеры и демонстрации FMC-RF1 ЦАП одноканальный 433
Назначение пинов FMC-RF1 ЦАП 434
Модуль FMC-RF1 ЦАП двухканальный 435
Содержание 435
Описание 435
Библиотека блоков 435
Маска блока 435
Параметры блока 435
Входы блока 436
Примеры и демонстрации FMC-RF1 ЦАП двухканальный 437
Модуль FMC-RF1 DUT ЦАП инициализация 438
Содержание 438
Описание 438
Библиотека блоков 438
Маска блока 438
Параметры блока 438
Настройка модуля ввода-вывода BA-IS-XX 440
Содержание 440
Описание 440
Библиотека блока 440
Маска блока 440
Параметры блока 441
Электроэнергетические системы 442
Содержание 442
Блоки поддержки IEC 61850 442
Блоки поддержки PRP 442
Блоки поддержки C37.118 442
Блоки для моделирования электрических машин 442
Вспомогательные функции 442
Блоки поддержки IEC 61850 443
Содержание 443
Библиотека блоков для работы с SV по IEC 61850 443
Библиотека блоков для работы с GOOSE по IEC 61850 443
Библиотека блоков для работы с SV по IEC 61850 444
Содержание 444
Описание 444
Библиотека блоков 444
Блок IEC 61850 SV Передатчик 445
Маска блока 445
Входы блока 446

Выходы блока 446
Параметры блока 446
Панель “Параметры заголовка и приоритета” 446
Панель “Параметры полей SV” 446
Блок IEC 61850 SV Передатчик 8 ASDU 446
Маска блока 446
Входы блока 447
Выходы блока 447
Параметры блока 447
Панель “Параметры заголовка и приоритета” 447
Панель “Параметры полей SV” 448
Блок IEC 61850 SV Передатчик настраиваемый 448
Маска блока 448
Входы блока 449
Выходы блока 450
Параметры блока 450
Панель “Параметры заголовка и приоритета” 450
Панель “Параметры полей SV” 450
Блок IEC 61850 SV Приёмник 450
Маска блока 450
Параметры блока 451
Параметры аналоговых сигналов 451
Распределение масштабирующих коэффициентов 451
Опции для распаковки аналоговых сигналов 451
Входы блока 451
Выход блока 451
Блок IEC 61850 SV Приёмник 8 ASDU 451
Маска блока 451
Параметры блока 451
Параметры аналоговых сигналов 451
Распределение масштабирующих коэффициентов 451
Опции для распаковки аналоговых сигналов 452
Входы блока 452
Выход блока 452
Блок IEC 61850 SV Реконструктор сигнала 452
Маска блока 452
Входы блока 452
Выход блока 452
Параметры блока 452
Блок IEC 61850 Параметр качества SV 452
Маска блока 452
Входы блока 453
Выход блока 453
Параметры блока 453
Атрибуты качества 453
Атрибут Detail quality 453
Примеры и демонстрации RITM IEC 61850 SV 455
Содержание 455
SV-поток с 80 выборками на период 455
SV-поток с 256 выборками на период 455
Настраиваемый SV-поток 456
Библиотека блоков для работы с GOOSE по IEC 61850 458
Содержание 458
Описание 458
Библиотека блоков 458

Блок IEC 61850 GOOSE Передатчик 459
Маска блока 459
Входы блока 460
Выходы блока 460
Параметры блока 460
Панель “Параметры заголовка и приоритета” 460
Панель “Параметры полей GOOSE” 460
Панель “Параметры allData” 460
Панель “Интервал передачи сообщения” 461
Блок IEC 61850 GOOSE Передатчик Спуфинг 461
Маска блока 462
Входы блока 463
Выходы блока 463
Параметры блока 464
Панель “Параметры работы спуфинга” 464
Панель “Параметры заголовка и приоритета” 464
Панель “Начальные параметры полей GOOSE” 464
Панель “Параметры allData” 464
Панель “Интервал передачи сообщения” 465
Блок IEC 61850 GOOSE Приёмник 465
Маска блока 465
Параметры блока 465
Вход блока 466
Выход блока 466
Блок IEC 61850 Параметр качества GOOSE 466
Маска блока 466
Входы блока 466
Выход блока 466
Параметры блока 466
Атрибуты качества 466
Атрибут Detail quality 467
Примеры и демонстрации RITM IEC 61850 GOOSE 468
Содержание 468
Передача сигналов любых типов данных по GOOSE-сообщениям 468
Пример моделирования GOOSE-спуфинга 468
Блоки поддержки PRP 470
Содержание 470
Библиотека блоков для работы с PRP по IEC 62439-3-2016 470
Библиотека блоков для работы с PRP по IEC 62439-3-2016 471
Содержание 471
Описание 471
Библиотека блоков 471
Блок Формирователь PRP-кадров 471
Маска блока 471
Входы блока 471
Выходы блока 471
Блок Отбрасыватель дубликатов PRP-кадров 472
Маска блока 472
Входы блока 472
Выходы блока 472
Примеры и демонстрации RITM PRP 473
Блоки поддержки C37.118 474
Содержание 474
Библиотека блоков для работы с C37.118 474
Библиотека блоков для работы с C37.118 475

Содержание 475
Описание 475
Библиотека блоков 479
Блок С37.118 Передатчик 480
Маска блока 480
Входы блока 481
Параметры блока 481
Блок С37.118 УСВИ 482
Маска блока 482
Входы блока 483
Выходы блока 483
Параметры блока 484
Блок С37.118 Приемник 485
Маска блока 485
Входы блока 486
Выходы блока 486
Параметры блока 487
Блок С37.118 Опорный сигнал 489
Маска блока 489
Входы блока 489
Выходы блока 489
Параметры блока 489
Блок С37.118 Измеритель СВИ 489
Маска блока 489
Входы блока 490
Выходы блока 490
Параметры блока 490
Примеры и демонстрации РИТМ С37.118 491
Содержание 491
Передача и прием синхронизированных векторных измерений (СВИ) по UDP 491
Блоки для моделирования электрических машин 492
Содержание 492
Библиотека блоков для моделирования электрических машин 492
Блоки для моделирования электрических машин 493
Содержание 493
Описание 493
Библиотека блоков 493
Блок RITM Phase-Domain Synchronous Machine 493
Маска блока 493
Порты 495
Параметры блока 495
Параметры вкладки “Main”: 495
Параметры вкладки “Impedances”: 496
Параметры вкладки “Time Constants”: 496
Параметры вкладки “Mechanical”: 496
Параметры вкладки “Faulted Windings”: 496
Параметры вкладки “Saturation”: 497
Параметры вкладки “Initial Conditions”: 497
Список литературы 497
Примеры и демонстрации RITM Phase-Domain Synchronous Machine 498
Вспомогательные функции 499
Вспомогательная функция ritm.energy.convertToComtrade 500

Пакет поддержки “РИТМ”

Комплекс полунатурного моделирования “РИТМ” и пакет поддержки “РИТМ” в Simulink представляют собой программно-аппаратное решение “под ключ”, предоставляющее возможности тестирования в реальном времени.

Дополнительная информация о КПМ “РИТМ” и пакете поддержки “РИТМ” доступна в разделе Введение.

Установка

Содержание

- Предварительные требования
- Установка пакета поддержки “РИТМ”
- Настройка соединения между хост-компьютером и машиной “РИТМ”
  - Настройка хоста
  - Настройка “РИТМ”
- Обновление пакета поддержки “РИТМ”
- Обновление прошивки “РИТМ”

Предварительные требования

- Версия MATLAB: R2022a и старше (поддерживаются Windows и Linux в качестве хоста)
- Любой из поддерживаемых хост-компиляторов: https://www.mathworks.com/support/compilers.html
- Необходимые тулбоксы: MATLAB, Simulink, Simulink Coder

Установка пакета поддержки “РИТМ”

- Скачайте архив с пакетом поддержки (файл вида RITM-library-vX.Y.Z.zip)
- Распакуйте архив в директорию с доступом для записи. Не используйте пробелы и не-ASCII символы в пути и имени директории
- Запустите MATLAB скрипт installRITM, который находится в директории с распакованными файлами

Во время установки будут скомпилированы блоки Simulink (S-функции) и прописаны необходимые пути в MATLAB.

Если директорию с распакованными файлами потребуется перенести в другое место, то скрипт installRITM нужно будут запустить заново.

Настройка соединения между хост-компьютером и машиной “РИТМ”

Рекомендуется соединить хост (то есть компьютер разработчика) и “РИТМ” напрямую при помощи сетевого патч-корда.
Производить подключение необходимо через Ethernet порт “РИТМ” с обозначением **Host**.

Настройка хоста

По умолчанию IP адрес “РИТМ” 192.168.56.3. Для первоначальной настройки следует выбрать IP адрес для хоста из этой же подсети.

Используя средства вашей операционной системы, настройте IP адрес для хоста, например, следующим образом:
IP адрес: 192.168.56.1
Маска подсети: 255.255.255.0

Настройка “РИТМ”

Чтобы изменить IP адрес РИТМ, используйте приложение РИТМ.Управление машинами, как описано в разделе РИТМ.Управление машинами -> Настройка сетевого интерфейса хост-таргет на машине.
После изменения IP адреса РИТМа назначьте для хоста IP адрес из такой же подсети.

Обновление пакета поддержки “РИТМ”

Используя приложение РИТМ.Управление машинами можно автоматически обновить версию пакета поддержки, если доступна более новая версия.

Обновление прошивки “РИТМ”

Используя приложение РИТМ.Управление машинами можно автоматически обновить прошивку машины “РИТМ”, если доступна более новая версия.

Введение

Содержание

- Комплекс полунатурного моделирования “РИТМ”
- ОСРВ “РИТМ.Реальное время”
  - Технические возможности ОСРВ “РИТМ.Реальное время”
  - Техническая информация
- Пакет поддержки “РИТМ”
  - Технические возможности пакета поддержки “РИТМ”
- Общая информация

В данном разделе приводится информация о комплексе полунатурного моделирования “РИТМ” (КПМ “РИТМ”) и пакете поддержки “РИТМ” в Simulink.

Комплекс полунатурного моделирования “РИТМ”

Комплекс полунатурного моделирования “РИТМ” представляет собой интегрированное, протестированное аппаратное решение. КПМ “РИТМ” собирается под конкретные требования заказчика и поставляется с необходимыми модулями ввода-вывода.

Функционирование КПМ “РИТМ” обеспечивается операционной системой реального времени “РИТМ.Реальное время”.

ОСРВ “РИТМ.Реальное время”

Операционная система “РИТМ.Реальное время” предназначена для запуска приложений пользователя в реальном времени с подключением к внешнему оборудованию через интерфейсные модули ввода-вывода.

“РИТМ.Реальное время” позволяет запускать детерминированные симуляции с гарантированным временем отклика приложения реального времени. Для этого операционная система оснащена ядром реального времени, предоставляющим такие гарантии.

Частью операционной системы также являются управляющие драйверы для интерфейсных модулей ввода-вывода и низкоуровневые библиотеки для работы с этими модулями ввода-вывода. Это позволяет подключать комплексы под управлением “РИТМ.Реальное время” к внешнему оборудованию, такому, как исполнительные механизмы (актуаторы), измерительные устройства (датчики).

Поддержка разнообразных интерфейсных модулей ввода-вывода через предоставляемые драйвера, интегрированные с ядром реального времени, а также низкоуровневая библиотека функций для работы с модулями ввода-вывода позволяет разработчикам приложений реального времени экономить время и фокусироваться на алгоритмических аспектах реализации системы.

Технические возможности ОСРВ “РИТМ.Реальное время”

- Запуск и работа на аппаратных комплексах полунатурного моделирования “РИТМ”
- Инструментарий для сборки пользовательских приложений реального времени

- Запуск приложений в реальном времени с детерминированным вызовом расчета алгоритма и гарантиями времени отклика
- Драйверы и низкоуровневые библиотеки для управления интерфейсными модулями ввода-вывода
- Управление, запуск, остановка и мониторинг выполнения приложения реального времени
- Поддержка распространенных интерфейсов и протоколов ввода-вывода:
  - Аналого-цифровые преобразователи
  - Цифро-аналоговые преобразователи
  - Цифровые входы и выходы
  - MIL-STD-1553
  - ARINC 429
  - CAN
  - Цифровые интерфейсы: UART, I2C, SPI, PWM, QAD, QAE
  - Квадратурные энкодеры и декодеры
  - Многие другие

Техническая информация

- Базовая операционная система: GNU/Linux (Arch Linux) с ядром Linux RT-PREEMPT
- Инструментарий для сборки: GNU GCC
- Минимальный гарантированный шаг расчета: от 10 до 100 мкс (в зависимости от аппаратной конфигурации комплекса полунатурного моделирования “РИТМ”)

Пакет поддержки “РИТМ”

Пакет поддержки “РИТМ” реализует интеграцию между средой модельно-ориентированного проектирования MATLAB/Simulink и КПМ “РИТМ”.

Технические возможности пакета поддержки “РИТМ”

- Автоматическая сборка, загрузка и запуск моделей Simulink на КПМ “РИТМ”
- Блоки Simulink для работы с модулями ввода-вывода на КПМ РИТМ
- Графическое приложение РИТМ.Управление машинами для управления машинами “РИТМ” и моделями
- Поддержка режима External Mode (Tune & Monitor) в Simulink
- Независимое выполнение модели на целевой системе
- Поддержка многозадачного, многочастотного выполнения моделей в реальном времени
- Поддержка нативных средств Simulink: Simulation Data Inspector, To Workspace, Scope
- Автоматический установщик пакета поддержки
- Автоматическое обновление пакета поддержки и прошивки КПМ “РИТМ”
- Поддержка запуска пользовательских алгоритмов на модулях ввода-вывода с ПЛИС

Общая информация

Операционная система реального времени “РИТМ.Реальное время” позволяет создавать современные, высокопроизводительные комплексы полунатурного моделирования, стенды, тренажеры и осуществлять испытания встраиваемых алгоритмов в режиме быстрого прототипирования с подключением к реальному оборудованию через интерфейсные модули ввода-вывода.

Тестирование в реальном времени - это исследование функционирования системы (или её компонентов) в реальных рабочих условиях - при заданной частоте дискретизации, с эффектами, возникающими при работе с реальным оборудованием и интерфейсами, с учетом синхронизации различных выполняющихся задач. Тестирование в реальном времени применяется в технологиях быстрого прототипирования алгоритма управления (или Rapid Prototyping), полунатурного моделирования объекта управления (также известное как HIL, или Hardware-in-the-Loop), а также в применении к сбору данных и обработки сигналов.

Быстрое прототипирование - это внесение изменений в поведение алгоритма управления и исследование работы алгоритма с использованием реального железа. Быстрое прототипирование означает, что разрабатываемый алгоритм управления загружается на систему реального времени и подключается к реальному объекту управления через модули ввода-вывода. Быстрое прототипирование позволяет тестировать алгоритмы управления с подключением к реальному оборудованию и объекту управления “в железе”, отрабатывая детали, которые могут быть недоступны во время симуляции на обычных пользовательских операционных системах. При этом пользователю не надо иметь навыки низкоуровневого программирования. Быстрое прототипирование позволяет на ранних стадиях изучать работу системы с использованием реального оборудования и настоящего объекта управления. Такое тестирование повышает уверенность, что конечная система будет корректно работать в окончательной реализации и позволяет осуществлять верификацию системы, наиболее приближенной к реальным условиям.

Полунатурное моделирование - это имитация объекта управления в реальном времени на отдельном компьютере (или компьютерах) под управлением операционной системы реального времени. Полунатурное моделирование позволяет запускать модель объекта управления в реальном времени с подключением к реальным исполнительным механизмам, датчикам и блоку управления. Это позволяет получить полноценную имитацию объекта управления. Полунатурное моделирование может применяться в широком спектре задач. Включая создание симуляторов различных условий эксплуатации, тренажеров для обучения персонала - например, пилотов или технологов и создания полунатурных испытательных стендов для отладки системы управления. Это позволяет осуществлять систематическое, полноценное тестирование системы.

Быстрое прототипирование и полунатурное моделирование существенно сокращает циклы разработки и испытаний систем и повышает надежность разрабатываемой системы.

Комплекс полунатурного моделирования “РИТМ” и операционная система “РИТМ.Реальное время” специально разработаны для совместной бесшовной работы. Аппаратное обеспечение “РИТМ” и программное обеспечение “РИТМ.Реальное время” совместно позволяют разработчикам сложных современных систем решать самые амбициозные проектные задачи.

РИТМ.Управление машинами

Содержание

- Назначение программы
- Запуск
- Описание программы
- Главное меню
  - Обновление
  - Справка
- Настройка подключения
- Работа с машиной
  - Управление подключением к машине
  - Управление приложениями
  - Настройки машины

Назначение программы

Программа «РИТМ.Управление машинами» предназначена для решения следующих задач:

- Подключение и настройка машин реального времени КПМ «РИТМ»
- Управление приложениями реального времени, сгенерированными из моделей Simulink

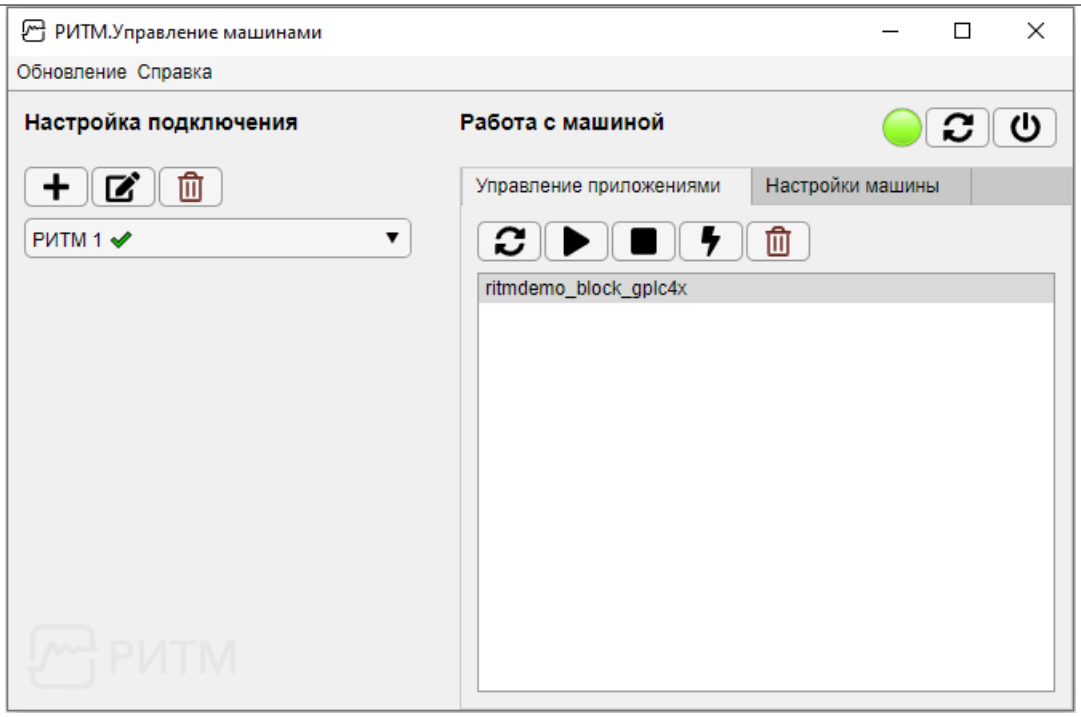
Запуск

Приложение РИТМ.Управление машинами можно запустить одним из двух способов:

1. В командном окне MATLAB выполнить команду **ritm.Manager**
2. В главном меню Matlab *APPS -> RITM Manager*

Описание программы

Интерфейс программы представлен на рисунке:

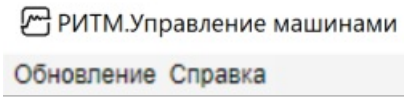


Интерфейс разбит на несколько областей:

- Главное меню
  Включает дополнительные возможности по работе с машинами, документацией и пакетом поддержки в целом.
- Настройка подключения
  Позволяет установить соединение с заданной машиной, управлять профилями различных машин.
- Работа с машиной
  Содержит инструменты для управления приложениями реального времени, загруженными на машину, также позволяет изменять настройки машины и управлять подключением к ней.

Главное меню

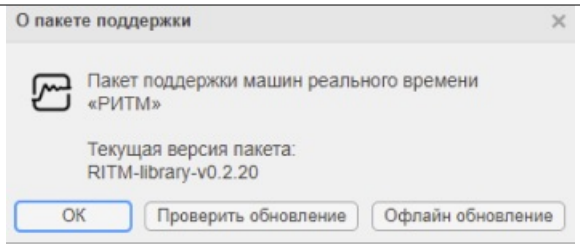
Главное меню содержит дополнительные функции приложения:



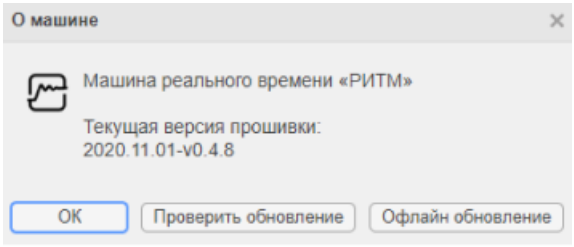
Обновление

Подменю «Обновление» позволяет узнать версию используемых инструментов и обновить их до последней версии:

- **Пакет поддержки** - позволяет проверить версию и обновить пакет поддержки



- **Прошивка машины** - позволяет проверить версию и обновить прошивку машины



Для обновления инструмента через интернет необходимо использовать кнопку “Проверить обновление”, которая проверяет текущую версию инструмента с актуальной и предлагает его обновить, если версия устаревшая.
Кнопка “Офлайн обновление” позволяет обновить инструмент с помощью доступных на хосте файлов обновления.

Справка

Подменю «Справка» дает пользователю доступ к полезной информации:

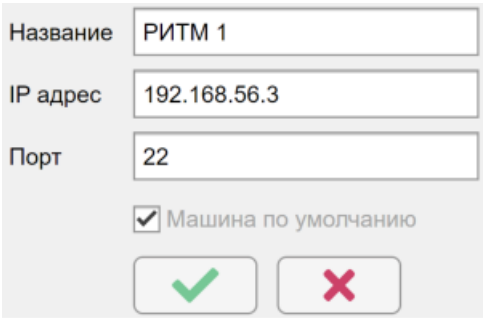
- **Документация** - открывает документацию «РИТМ.Управление машинами»
- **О программе** - показывает сведения о программе

Настройка подключения

Панель «Настройка подключения» предоставляет следующие возможности:

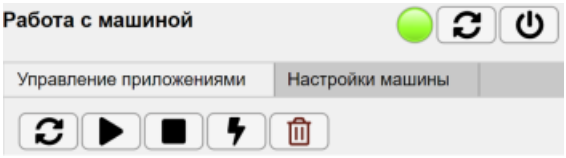
- Добавление/удаление профиля машины
- Задание параметров подключения машины (IP адрес, порт)
- Переключение между машинами
- Выбор машины, которая используется по умолчанию

Пример настройки представлен на рисунке:



Работа с машиной

Панель «Работа с машиной» содержит инструменты для управления подключенной машиной и состоит из нескольких областей:



- Управление подключением к машине
- Управление приложениями
- Настройки машины

Управление подключением к машине

Управление подключением осуществляется с помощью кнопок, также отображается статус подключения к текущей машине:

- **Статус соединения с машиной**
  Отображает текущий статус соединения с машиной.
- **Переподключение к машине** Осуществляет повторное подключение к машине, если соединение было нарушено.
- **Управление питанием машины**
  Позволяет выключить или перезагрузить машину.

Управление приложениями

На вкладке «Управление приложениями» выводится список всех приложений реального времени, загруженных на машину. Отображается, какое приложение запущено в данный момент.

Управление приложениями осуществляется с помощью кнопок:

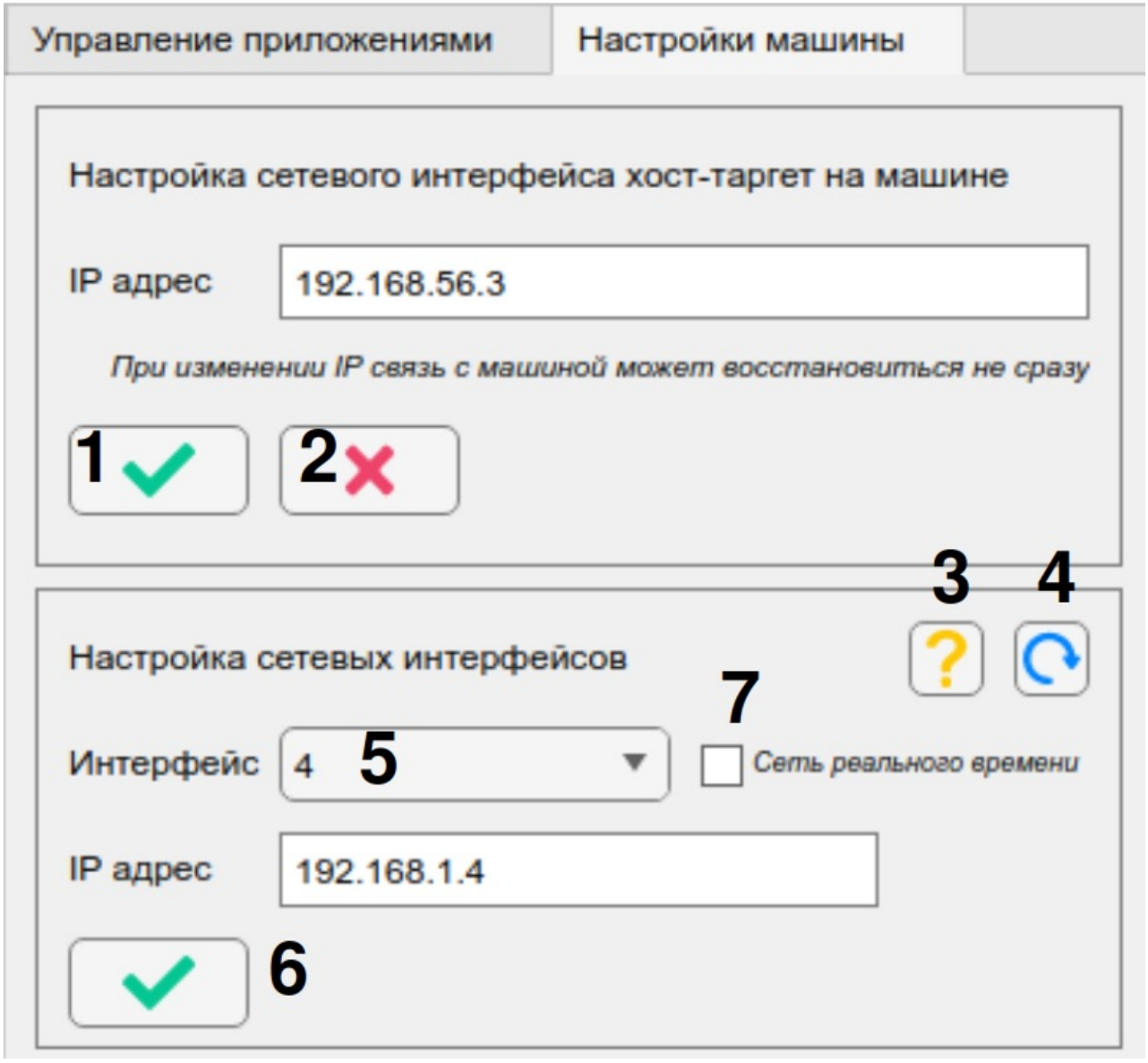
- **Обновление списка приложений**
  При этом происходит повторное подключение к машине и получение ее параметров.
- **Запуск приложения**
  Текущее запущенное приложение принудительно останавливается и запускается выбранное приложение.
- **Остановка приложения**
  Текущее запущенное приложение принудительно останавливается.
- **Автозапуск приложения**
  Выбранное приложение будет помещено в автозапуск, таким образом оно будет

автоматически запускаться при включении машины.

- **Удаление приложения с машины**
  Если выбранное приложение запущено, оно принудительно останавливается, а затем удаляется с машины.

Настройки машины

Панель “Настройки машины” позволяет настроить сетевой интерфейс хост-таргет, а также отобразить информацию о других доступных интерфейсах:



Данная панель разбита на две области, отвечающие за отдельные функции настройки сетевых интерфейсов:

Настройка сетевого интерфейса хост-таргет на машине
Данная панель позволяет сменить IPv4 адрес на машине. Также автоматически меняется IP адрес в профиле подключения. Если соединение не восстановилось сразу после смены адреса, используйте кнопку Переподключение к машине для восстановления подключения.

Чтобы сменить IPv4 адрес на желаемый, необходимо нажать на кнопку **1**. Также есть возможность вернуть исходный IPv4 адрес машины, если изменения не были приняты

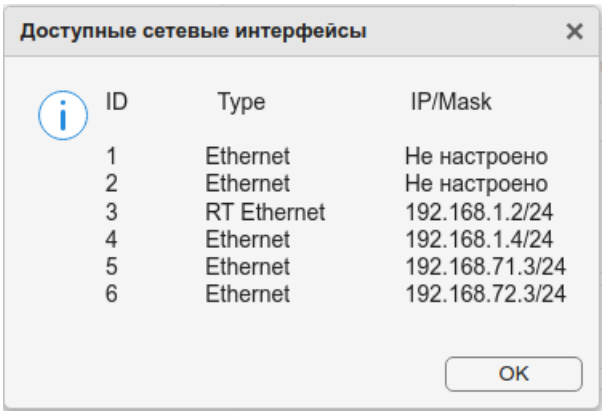
(не была нажата кнопка **1**), нажатием кнопки **2**.

Обратите внимание, что IP адрес хоста (заданный в настройках вашей операционной системы) должен находиться в одной подсети с IP адресом “РИТМ”.

Настройка сетевых интерфейсов

Данная панель позволяет настроить доступные на машине сетевые интерфейсы. Если при настройке сетевого интерфейса была выбрана опция “Сеть реального времени” (сетевой интерфейс RT Ethernet), то для работы с ним нужно использовать сетевые блоки “реального времени” с префиксом RT.

Список доступных интерфейсов можно получить, нажав на кнопку **3**. При этом откроется диалоговое окно, отображающее все доступные интерфейсы, как настроенные, так и ненастроенные:



Можно настроить интерфейсы, выбрав идентификатор интерфейса с помощью выпадающего меню **5** и назначив желаемый IP адрес интерфейса. Для использования сетевых блоков реального времени (с префиксом RT) на выбранном интерфейсе необходимо перед применением изменений нажать галочку **7**. После этого необходимо нажать на кнопку **6**, чтобы применить изменения.

Нажатием на кнопку **4** сбрасываются настройки всех сетевых интерфейсов, кроме интерфейса хост-таргет.

Не рекомендуется одновременно задействовать несколько сетевых интерфейсов с IP адресами, принадлежащими одной сети.
Это замечание относится также и к интерфейсу хост-таргет. IP адреса из одной подсети на нескольких сетевых интерфейсах могут приводить к ошибкам маршрутизации и, например, невозможности подключиться к машине по интерфейсу хост-таргет.
В случае возникновения такой ситуации следует отключить сетевые кабели от всех портов, кроме порта, отмеченного как **Host** на РИТМе, переподключиться к машине и корректно настроить сетевые интерфейсы.

Начало работы с пакетом поддержки “РИТМ”

Содержание

- Пример простой модели
- Пример работы с модулями ввода-вывода

Если вы раньше не работали с Simulink, мы рекомендуем пройти Интерактивный вводный курс по Simulink.

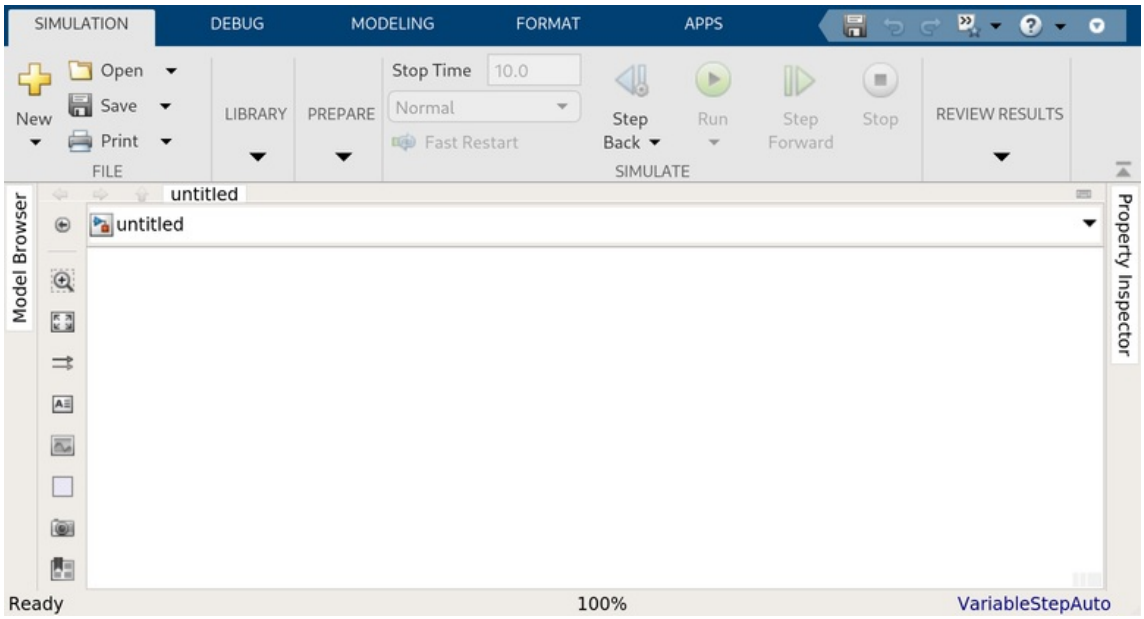
Дополнительная техническая информация, демонстрации и примеры доступны в документации:

- Введение
- Установка

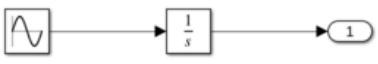
Пример простой модели

В этом примере показано, как создать с нуля модель Simulink и запустить её на КПМ “РИТМ”.

- Создайте новую модель Simulink:

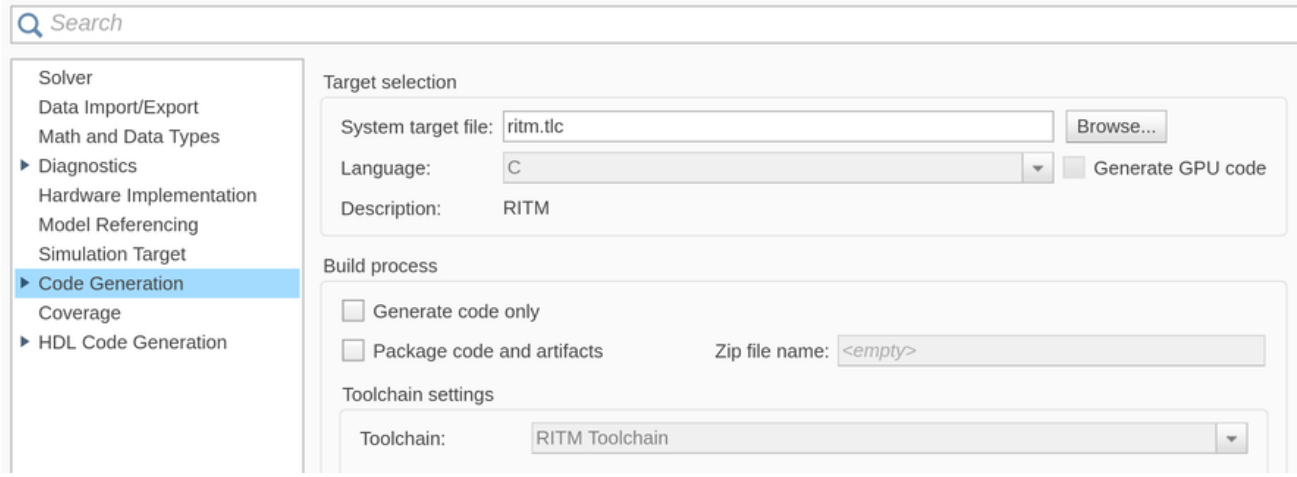


- Добавьте в модель стандартные блоки Simulink Sine Wave, Integrator и Out и соедините их:

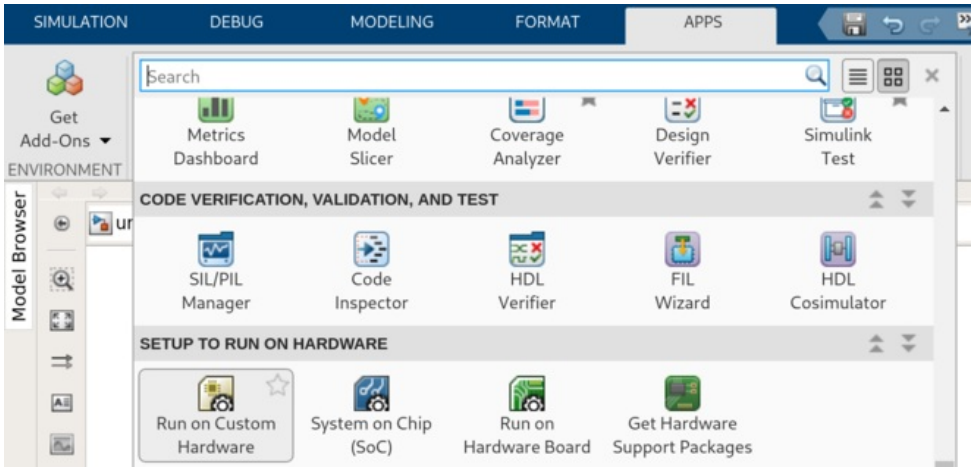


- Включите запись сигналов для выхода блока Integrator:

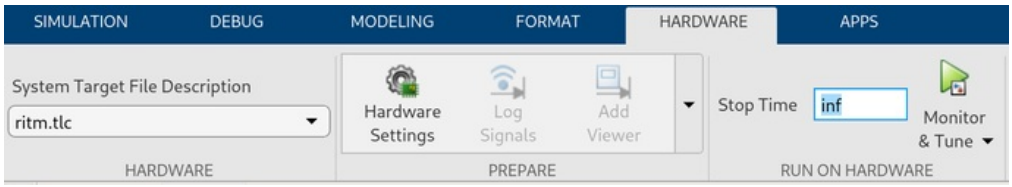
- Зайдите в настройки модели (Ctrl+E) и в панели Code Generation выберите ritm.tlc как System target file:



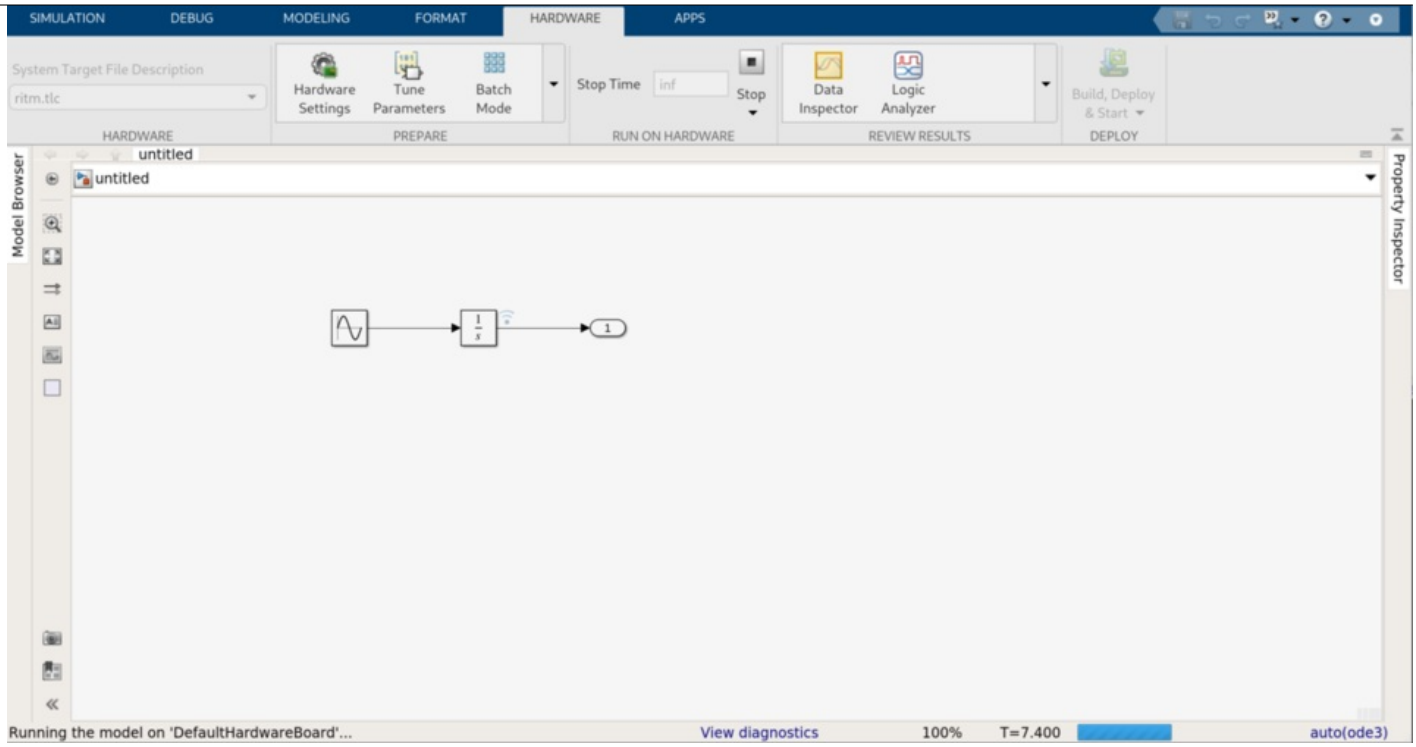
- Во вкладке Apps модели выберите Run on Custom Hardware:



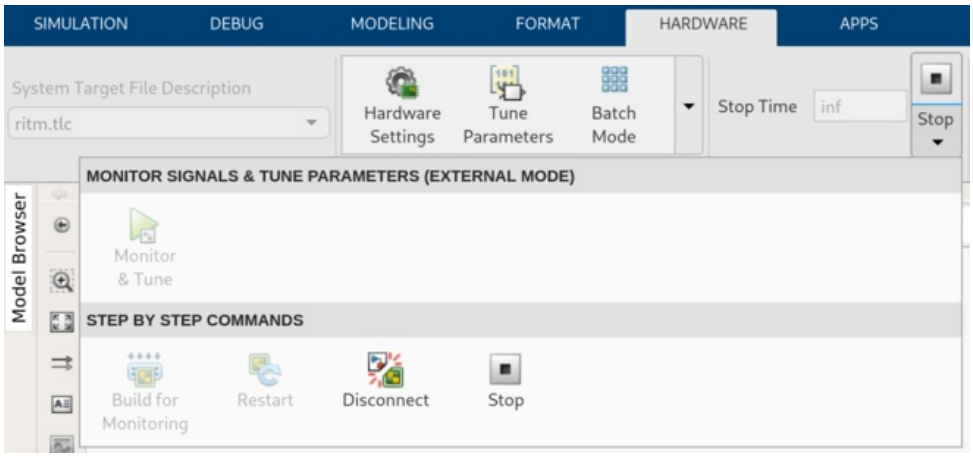
- Выставьте бесконечное время симуляции и нажмите кнопку Monitor & Tune:



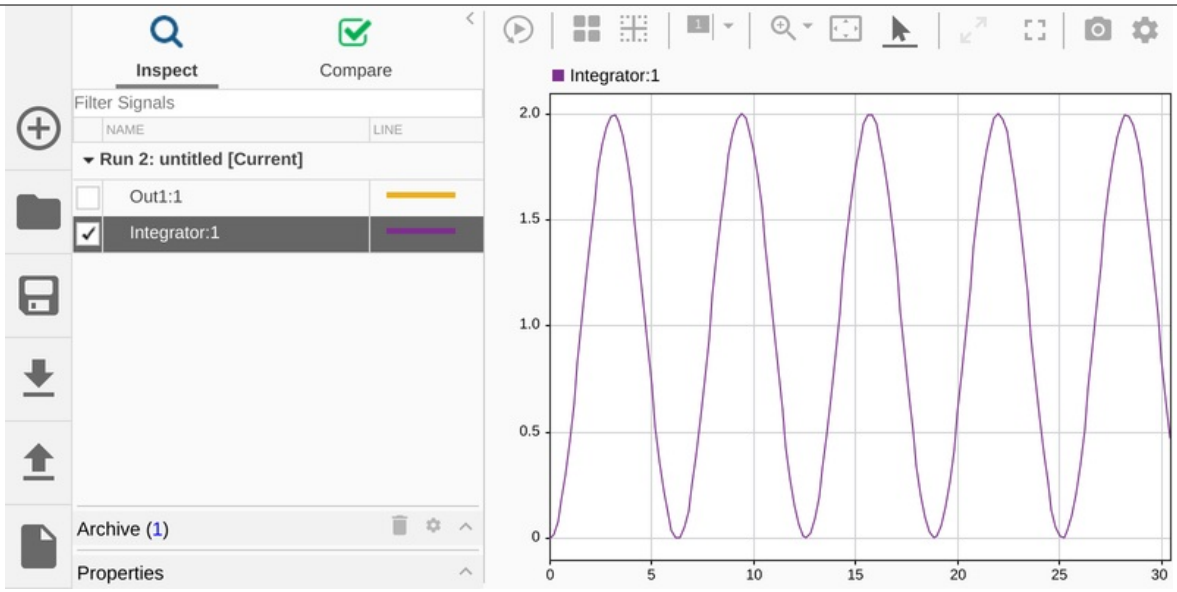
- Из модели генерируется код и запускается на машине “РИТМ” по умолчанию:



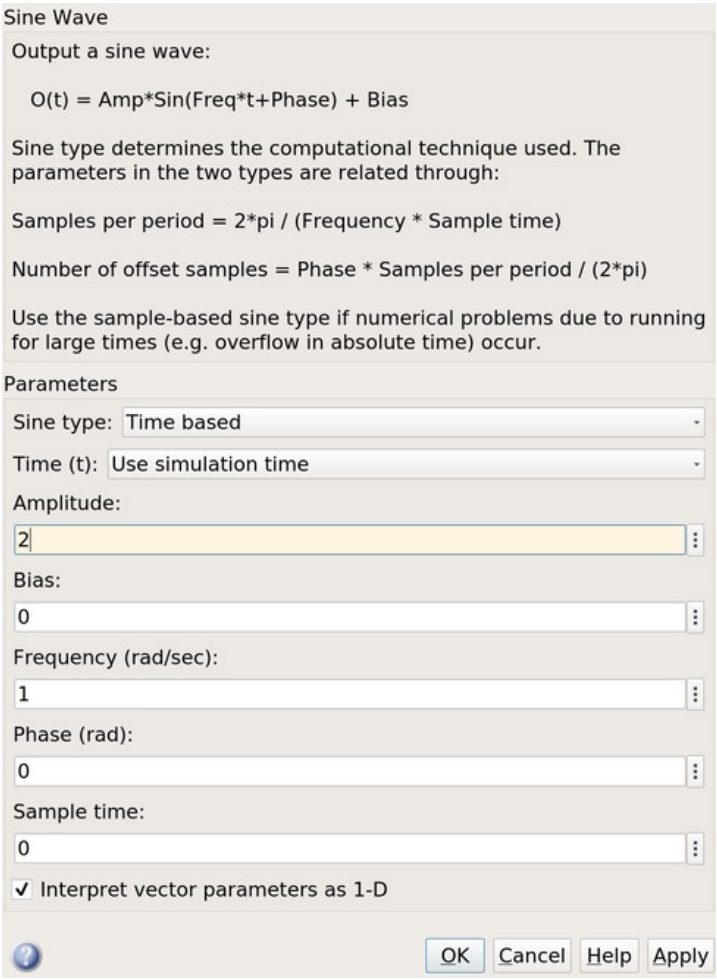
- Можно остановить выполнение модели на “РИТМ”, нажав Stop или отключить модель от “РИТМ” (приложение реального времени продолжит выполняться) нажав Disconnect в ниспадающем меню:

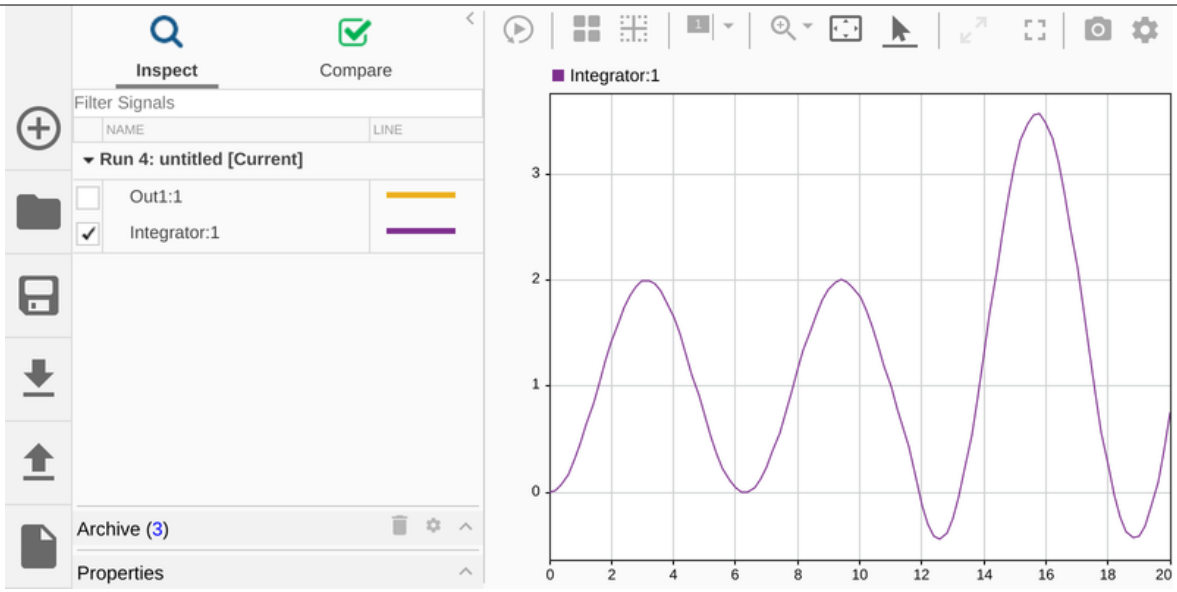


- Во время выполнения можно наблюдать за записанными сигналами при помощи Data Inspector:



- А также менять настраиваемые параметры модели “на лету” и видеть изменения. Например, изменить амплитуду синусоиды с 1 на 2 и наблюдать изменения в Data Inspector:





Пример работы с модулями ввода-вывода

В документации к каждому модулю ввода-вывода приводится информация о назначении пинов и пример использования блоков Simulink для работы с модулем ввода-вывода.

Настройки модели

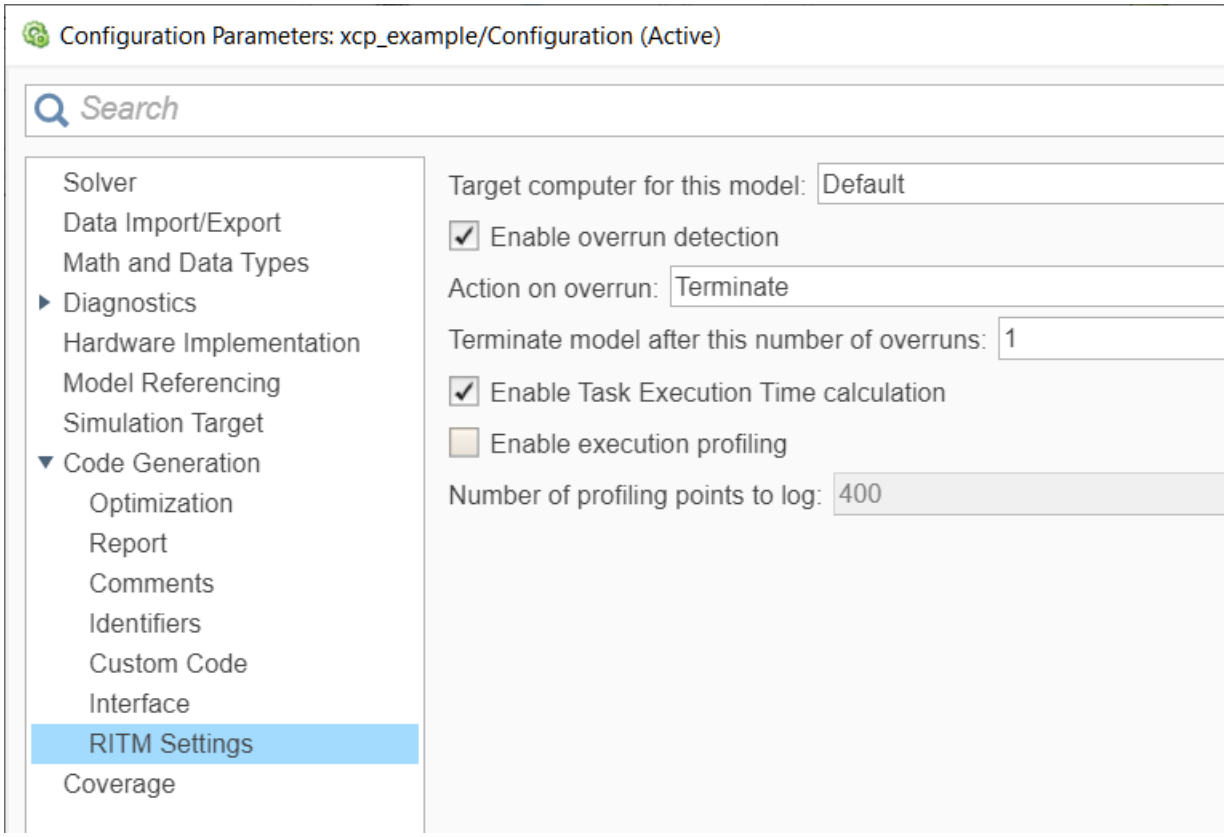
Содержание

- Панель настроек модели
- Описание настроек
  - Target computer for this model
  - Enable overrun detection
  - Action on overrun
  - Terminate model after this number of overruns
  - Enable Task Execution Time calculation
  - Enable execution profiling
  - Number of profiling points to log
- См. также

После выбора ritm.tlc в настройках System Target File пользователю становится доступна панель настроек модели для исполнения на КПМ “РИТМ”. Панель настроек расположена в категории **Code Generation**.

Панель настроек модели

Общий вид панели настроек приведен на рисунке:



Ниже перечислены настройки, доступные пользователю и дано их краткое описание.

Описание настроек

Target computer for this model

Описание

Выбор машины КПМ РИТМ на которой будет запущена модель.

Значение по умолчанию

Default - машина, выбранная по умолчанию в РИТМ.Управление машинами.

Примечания

Настройки машин рассмотрены в разделе документации РИТМ.Управление машинами.

Enable overrun detection

Описание

Включение или отключение обнаружения выхода за шаг расчета.

Значение по умолчанию

on - По умолчанию обнаружение выхода за шаг расчета **включено**.

Примечания

Рекомендуется не отключать данную настройку.

Action on overrun

Описание

Выбор действия при обнаружении выхода за шаг расчета. Поддерживаются следующие действия:

- *Notify* - вывод сообщения о выходе за шаг расчета.
- *Terminate* - остановка модели.

Значение по умолчанию

Terminate

Terminate model after this number of overruns

Описание

Количество обнаруженных выходов за шаг расчета, после которых требуется остановить исполнение модели.

Значение по умолчанию

1

Enable Task Execution Time calculation

Описание

Включение или выключения режима расчета времени, затраченного на исполнение каждого шага расчета в модели (TET).

Значение по умолчанию

on

Enable execution profiling

Описание

Включение или отключение режима профилирования модели.

Значение по умолчанию

off - По умолчанию, профилирование модели отключено.

Примечания

Режим профилирования может быть включен только при включенном режиме расчета TET.

Number of profiling points to log

Описание

Количество точек профилирования для анализа.

Значение по умолчанию

400

См. также

Начало работы

Обнаружение выхода за шаг расчета

Профилирование модели

Режимы выполнения модели

Содержание

- Режим независимого выполнения
  - Область применения
- Режим интерактивного выполнения
  - Область применения

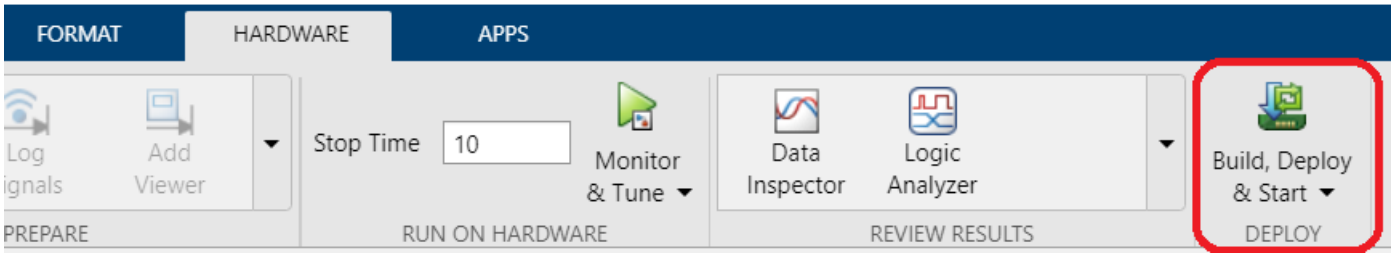
Поддерживаются следующие режимы выполнения модели на КПМ “РИТМ”:

- Выполнение в виде **независимого** приложения реального времени (*Build, Deploy & Start*)
- Выполнение в виде **интерактивного** приложения реального времени (*Monitor & Tune*)

Каждый из приведенных выше режимов предназначен для разных сценариев использования КПМ “РИТМ”.

Режим независимого выполнения

Модель запускается на КПМ “РИТМ” нажатием кнопки **Build, Deploy & Start**:



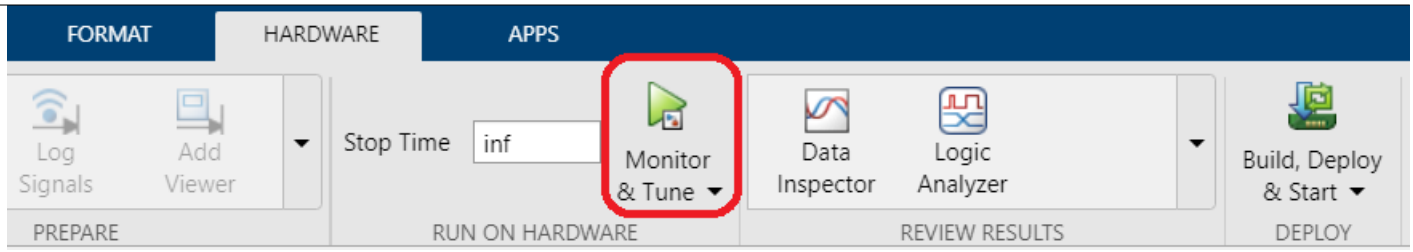
При этом из модели генерируется независимое от хост-компьютера приложение реального времени, которое начинает работать на КПМ “РИТМ” без привязки к хост-компьютеру. Наблюдать за выполнением модели можно на мониторе, подключенном к КПМ “РИТМ”. Управлять выполнением модели можно при помощи РИТМ.Управление машинами.

Область применения

Этот режим выполнения предназначен для неинтерактивного, независимого от хост-компьютера выполнения модели на КПМ “РИТМ”. Данный режим может применяться для симуляции объекта управления, когда настройка параметров модели на лету не требуется. Основной сценарий применения - полунатурное моделирование (HIL).

Режим интерактивного выполнения

Модель запускается на КПМ “РИТМ” нажатием кнопки **Monitor & Tune**:



При этом из модели генерируется интерактивное приложение реального времени, которое начинает работать на КПМ “РИТМ”. Во время работы приложения реального времени можно менять параметры самой модели на лету, записывать сигналы модели и просматривать их в *Simulation Data Inspector* на хост-компьютере. Так же, как для независимого выполнения модели, в этом режиме можно наблюдать за выполнением модели на мониторе, подключенном к КПМ “РИТМ”. Управление выполнением приложения реального времени осуществлятся из самой модели Simulink.

Область применения

Этот режим выполнения предназначен для исследования поведения алгоритмов управления и интерактивной настройки приложения реального времени через модель Simulink. Основной сценарий применения - быстрое прототипирование алгоритмов управления.

Переполнение шага расчета

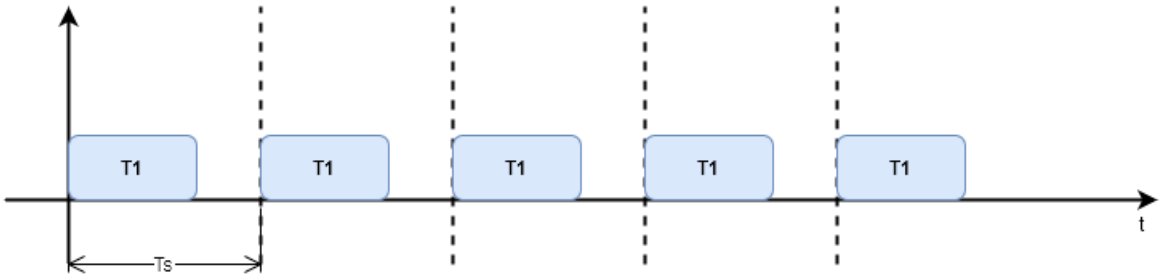
Содержание

- Что такое переполнение шага расчета?
- Методы диагностики
- Обработка переполнения шага расчета в КПМ “РИТМ”
- Пример
  - Принцип работы
  - Режим уведомления пользователя
  - Режим останова выполнения модели
- Рекомендации производителя

Что такое переполнение шага расчета?

Переполнение шага расчета - это ситуация, возникающая в системах жесткого реального времени, при которой выполнение расчета занимает больше времени, чем отведено на расчет одного шага.

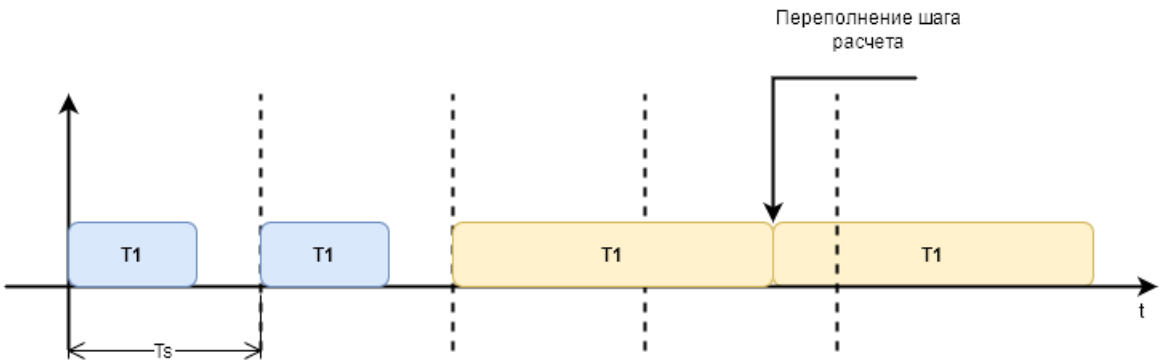
Циклограмма работы одной задачи без переполнения шага расчета:



Где:

- T1 - время, необходимое для вычислений
- Ts - время шага расчета

Циклограмма работы задачи с переполнением шага расчета:



Переполнение шага расчета может быть вызвано несколькими факторами:

- Вычислительная сложность задачи - задача не может быть расчитана за время шага расчета
- Задержки при работе с оборудованием

Переполнение шага расчета ведет к нарушению режима жесткого реального времени.

Методы диагностики

Для оценки возможности возникновения переполнения шага расчета требуется воспользоваться режимом профилирования модели.

Обработка переполнения шага расчета в КПМ “РИТМ”

Пакет поддержки КПМ “РИТМ” позволяет обнаруживать и обрабатывать переполнение шага расчета. Поддерживаются следующие методы обработки:

- Уведомление пользователя, без остановки выполнения модели
- Остановка выполнения модели после обнаружения определенного количества переполнений шага расчета. Количество задается пользователем.

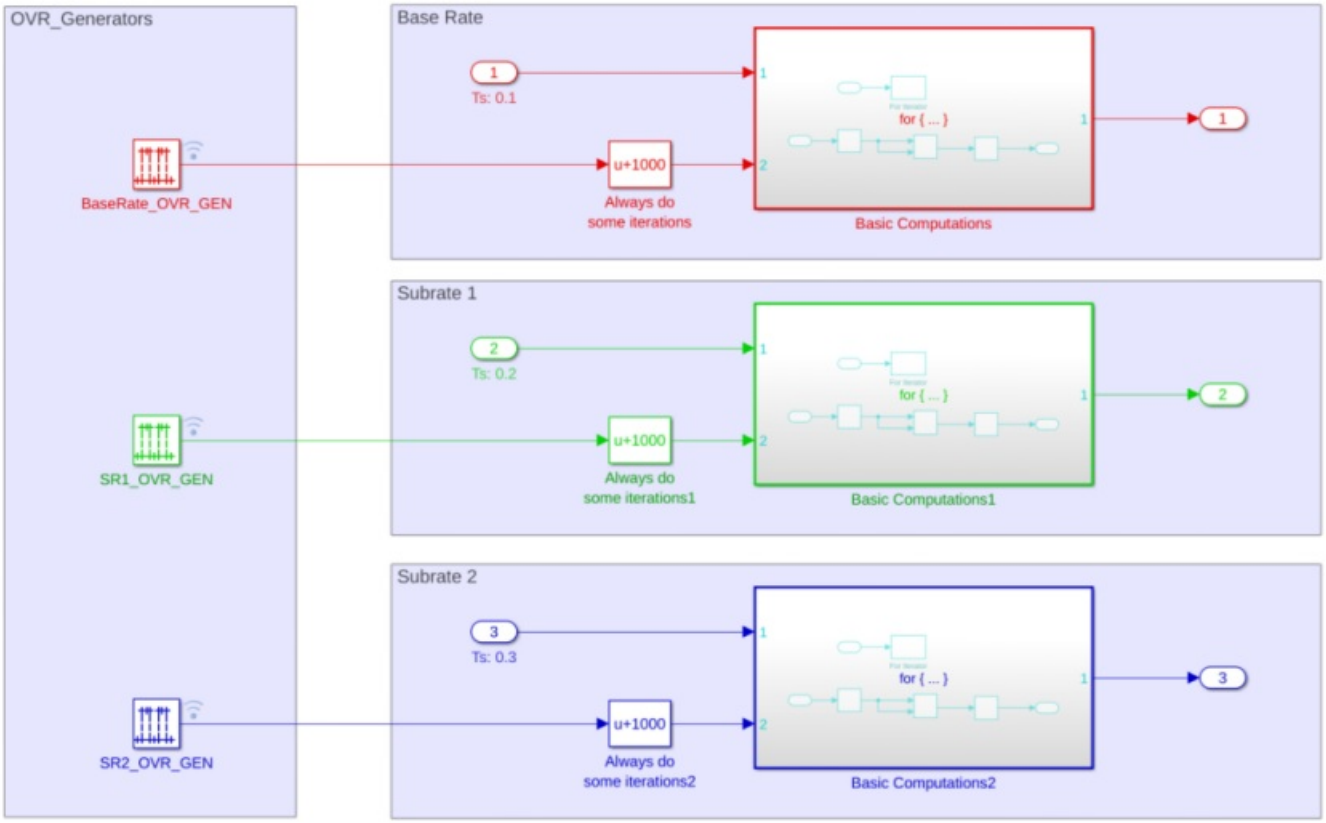
Пример

Демонстрация ritmdemo_detect_overrun предназначена для ознакомления с переполнением шага расчета и режимами обработки данного события.

Принцип работы

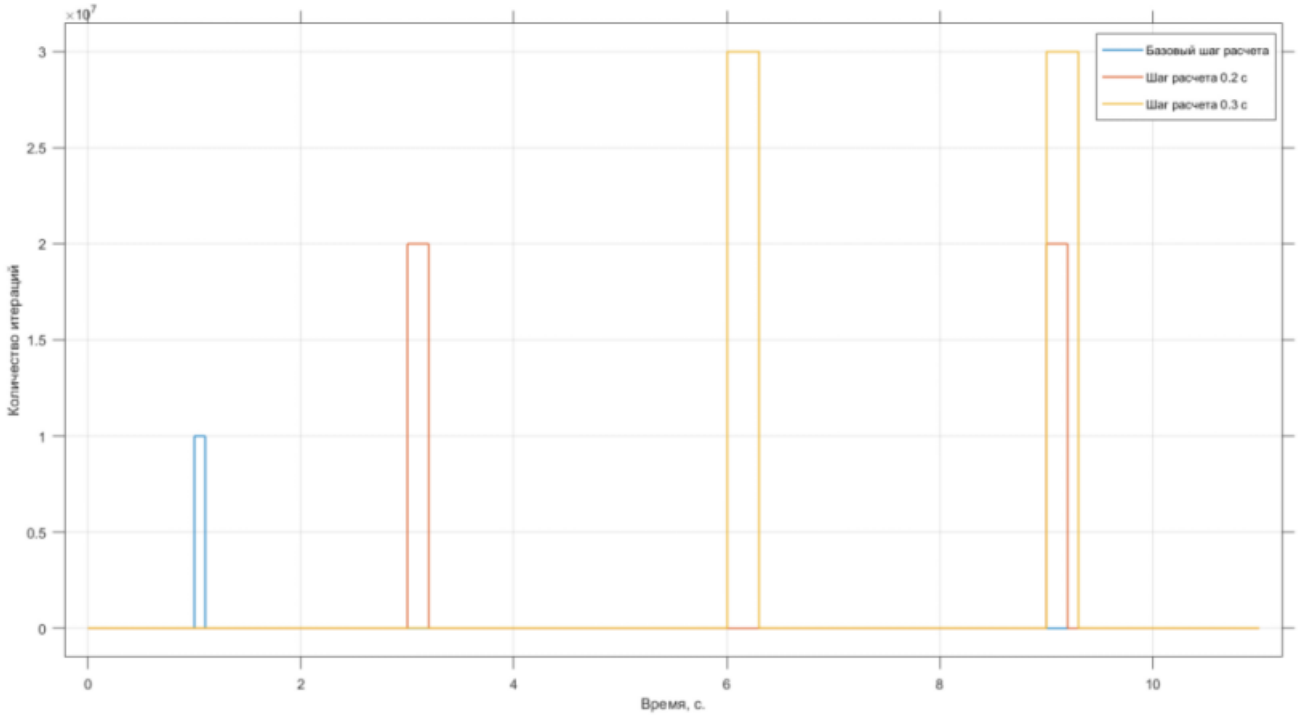
Рассмотрим модель ritmdemo_detect_overrun:

```
open_system('ritmdemo_detect_overrun')
```



В данной модели содержится несколько частот дискретизации (sample times) со следующими шагами расчета: 0.1, 0.2 и 0.3 с. Разные частоты дискретизации в модели отмечены цветами.

Блоки, входящие в группу *OVR_Generators* имитируют вычислительную нагрузку. Их циклограмма следующая:

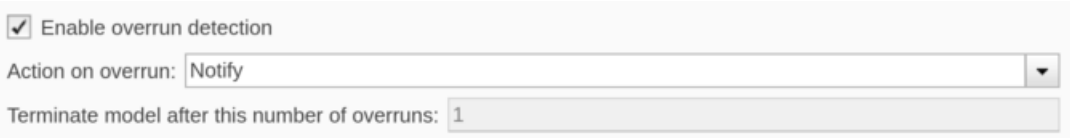


Из циклограммы видно, что шаги расчета будут переполняться по очереди при увеличении вычислительной нагрузки в определенной частоте дискретизации.

Ниже рассмотрено поведение модели при различных режимах обработки переполнения шага расчета.

Режим уведомления пользователя

Для выбора данного режима необходимо отметить настройку модели **Enable overrun detection** и выбрать режим *Notify* в выпадающем меню *Action on overrun*:



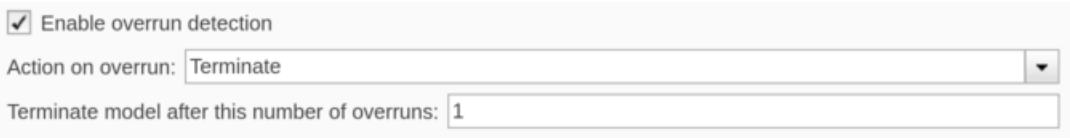
После запуска модели в режиме независимого выполнения, на монитор КПМ “РИТМ” будет выведено следующее:

```
** starting model ritmdemo_detect_overrun
Overrun detected: rate # 0, model time 1.000000!
Overrun detected: rate # 0, model time 1.000000!
Overrun detected: rate # 1, model time 3.300000!
Overrun detected: rate # 2, model time 6.500000!
Overrun detected: rate # 1, model time 9.300000!
Overrun detected: rate # 1, model time 9.500000!
Overrun detected: rate # 2, model time 9.500000!
Overrun detected: rate # 2, model time 9.800000!
** stopping model ritmdemo_detect_overrun
```

Модель не остановила своего выполнения после обнаружения множественных переполнений шага расчета.

Режим останова выполнения модели

Данный режим является режимом по умолчанию:



При выборе *Action on overrun Terminate* пользователь может задать количество переполнений шага расчета, после которых модель будет остановлена с помощью настройки *Terminate model after this number of overruns*. По умолчанию это количество равно 1.

После запуска модели в режиме независимого выполнения, на монитор КПМ “РИТМ” будет выведено следующее:

```
** starting model ritmdemo_detect_overrun
Overrun detected: rate # 0, model time 1.000000!
Maximum number of allowed overruns reached, stopping model execution!
** stopping model ritmdemo_detect_overrun
```

Модель остановила свое выполнение после обнаружения переполнения шага расчета.

Рекомендации производителя

1. Не рекомендуется отключать опцию обнаружения переполнения шага расчета. В противном случае не гарантируется выполнение задачи в реальном времени.
2. Режим уведомления пользователя о переполнении шага расчета служит для целей отладки моделей и не рекомендуется к промышленному применению.

Профилирование выполнения

Содержание

- Общие сведения
- Получение результатов профилирования
- Просмотр и интерпретация результатов
  - Дополнительные возможности
- Пример работы

Общие сведения

Профилирование приложения реального времени служит для точных измерений времени выполнения шага расчета. Профилирование позволяет определить возможное переполнение шага расчета и идентифицировать нарушения режима жесткого реального времени.

Получение результатов профилирования

Для получения результатов профилирования требуется выполнить следующие шаги:

1. Открыть настройки модели
2. На вкладке RITM Settings отметить настройки **Enable Task Execution Time calculation** и **Enable Execution Profiling**
3. Запустить модель
4. Выполнить следующую команду в командном окне MATLAB:

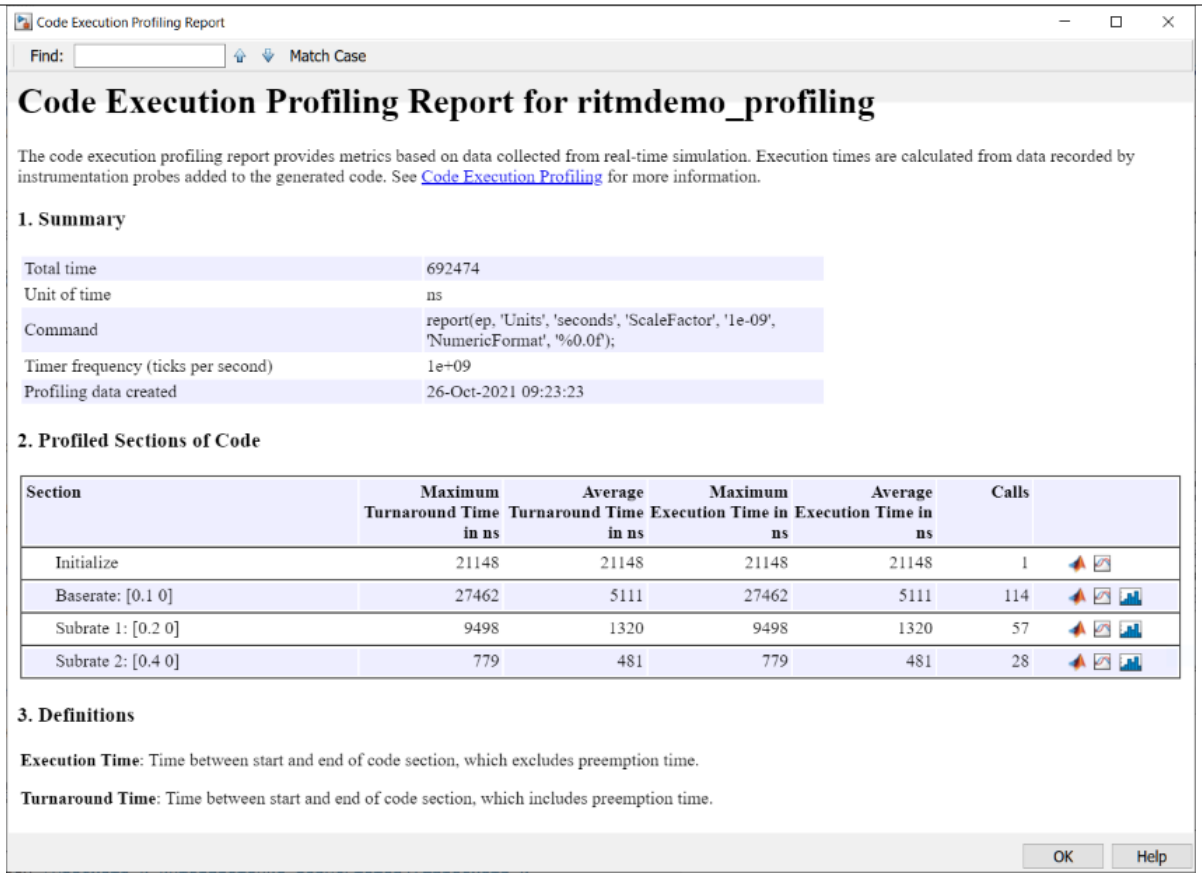
```
ritm.profile.getData(modelName)
```

Параметр modelName - строка, содержащая имя модели. Например:

```
ritm.profile.getData('ritmdemo_profiling')
```

Просмотр и интерпретация результатов

После выполнения команды откроется окно просмотра результатов профилирования:


Code Execution Profiling Report

Find: Match Case

Code Execution Profiling Report for ritmdemo_profiling

The code execution profiling report provides metrics based on data collected from real-time simulation. Execution times are calculated from data recorded by instrumentation probes added to the generated code. See Code Execution Profiling for more information.

1. Summary

Total time	692474
Unit of time	ns
Command	report(ep, 'Units', 'seconds', 'ScaleFactor', '1e-09', 'NumericFormat', '%0.0f');
Timer frequency (ticks per second)	1e+09
Profiling data created	26-Oct-2021 09:23:23

2. Profiled Sections of Code

Section	Maximum Turnaround Time in ns	Average Turnaround Time in ns	Maximum Execution Time in ns	Average Execution Time in ns	Calls
Initialize	21148	21148	21148	21148	1
Baserate: [0.1 0]	27462	5111	27462	5111	114
Subrate 1: [0.2 0]	9498	1320	9498	1320	57
Subrate 2: [0.4 0]	779	481	779	481	28

3. Definitions

Execution Time: Time between start and end of code section, which excludes preemption time.

Turnaround Time: Time between start and end of code section, which includes preemption time.

OK Help

Описание столбцов таблицы с результатами:

- **Section** - Шаг расчета
- **Maximum Turnaround Time in ns** - максимальное время выполнения шага расчета, включая время прерывания, [нс]
- **Average Turnaround Time in ns** - среднее время выполнения шага расчета, включая время прерывания, [нс]
- **Maximum Execution Time in ns** - максимальное время выполнения шага расчета, исключая время прерывания, [нс]
- **Average Execution Time in ns** - среднее время выполнения шага расчета, [нс]
- **Calls** - количество вызовов шага расчета

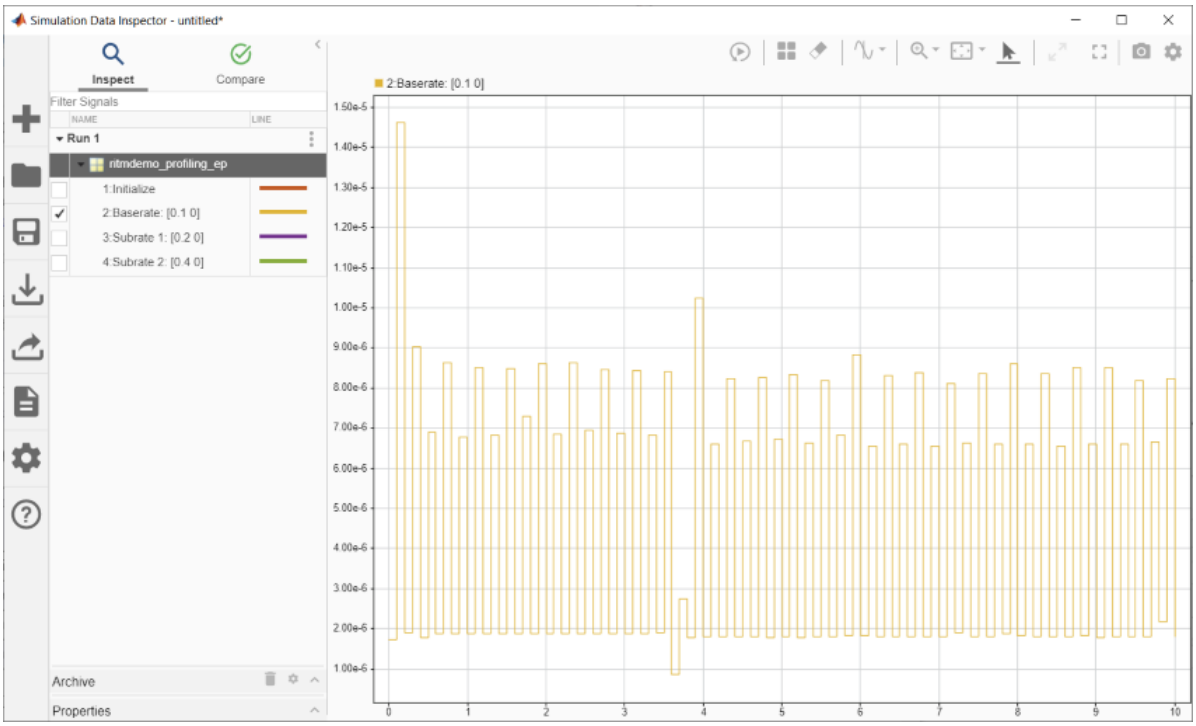
Описание пиктограмм

- Извлечение секции в отдельную переменную *ans* в рабочей области MATLAB:

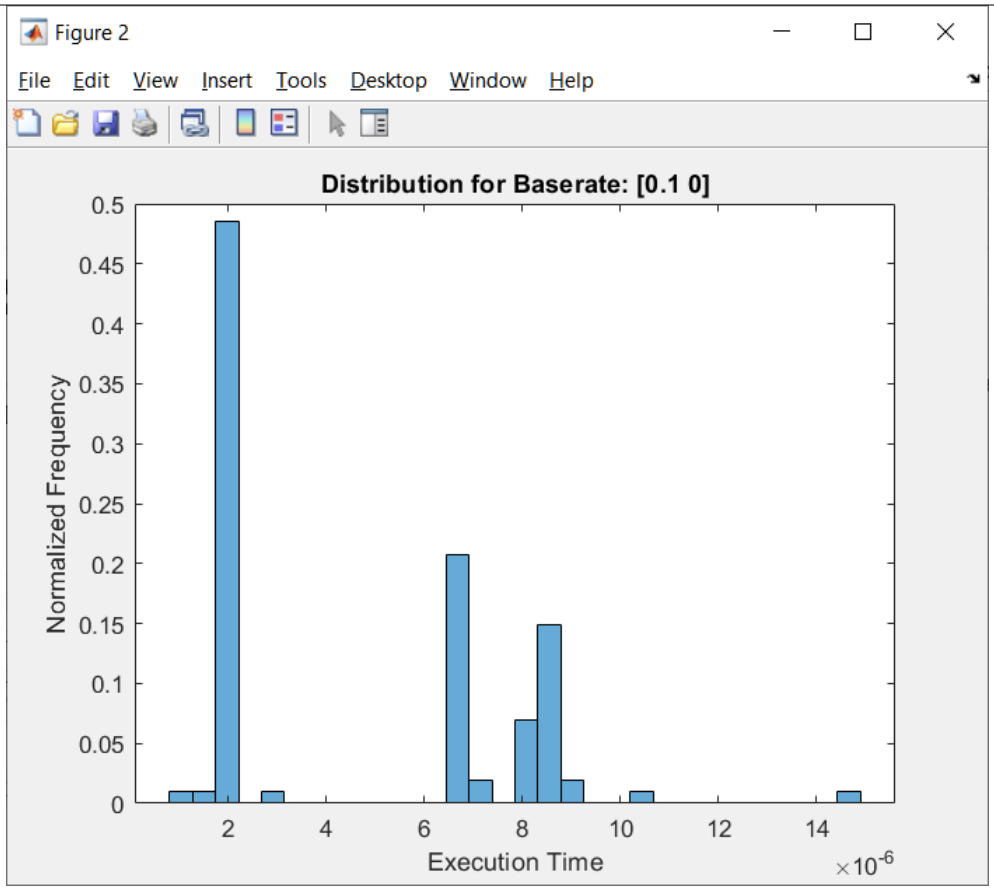
```
>> ritmdemo_profiling_ep.Sections(2)
  ExecutionTimeTaskSection with properties:

                      Name: 'Baserate: [0.1 0]'
                    Number: 2
    ExecutionTimeInTicks: [1722 14623 1890 9036 1779 6891 1864 8625 1867 6767 1865 8511 1862 ... ]
       SelfTimeInTicks: [1722 14623 1890 9036 1779 6891 1864 8625 1867 6767 1865 8511 1862 ... ]
  TurnaroundTimeInTicks: [1722 14623 1890 9036 1779 6891 1864 8625 1867 6767 1865 8511 1862 ... ]
 TotalExecutionTimeInTicks: 477120
      TotalSelfTimeInTicks: 477120
 TotalTurnaroundTimeInTicks: 477120
MaximumExecutionTimeInTicks: 14623
MaximumExecutionTimeCallNum: 2
    MaximumSelfTimeInTicks: 14623
```

MaximumSelfTimeCallNum: 2
MaximumTurnaroundTimeInTicks: 14623
MaximumTurnaroundTimeCallNum: 2
NumCalls: 101
ExecutionTimeInSeconds: [1.7220e-06 1.4623e-05 1.8900e-06 9.0360e-06 1.7790e-06 6.8910e-06 ...]
Time: [101×1 double]

- Построение графика времени исполнения секции в Simulation Data Inspector:



- Построение гистограммы распределения времени исполнения секции:



Дополнительные возможности

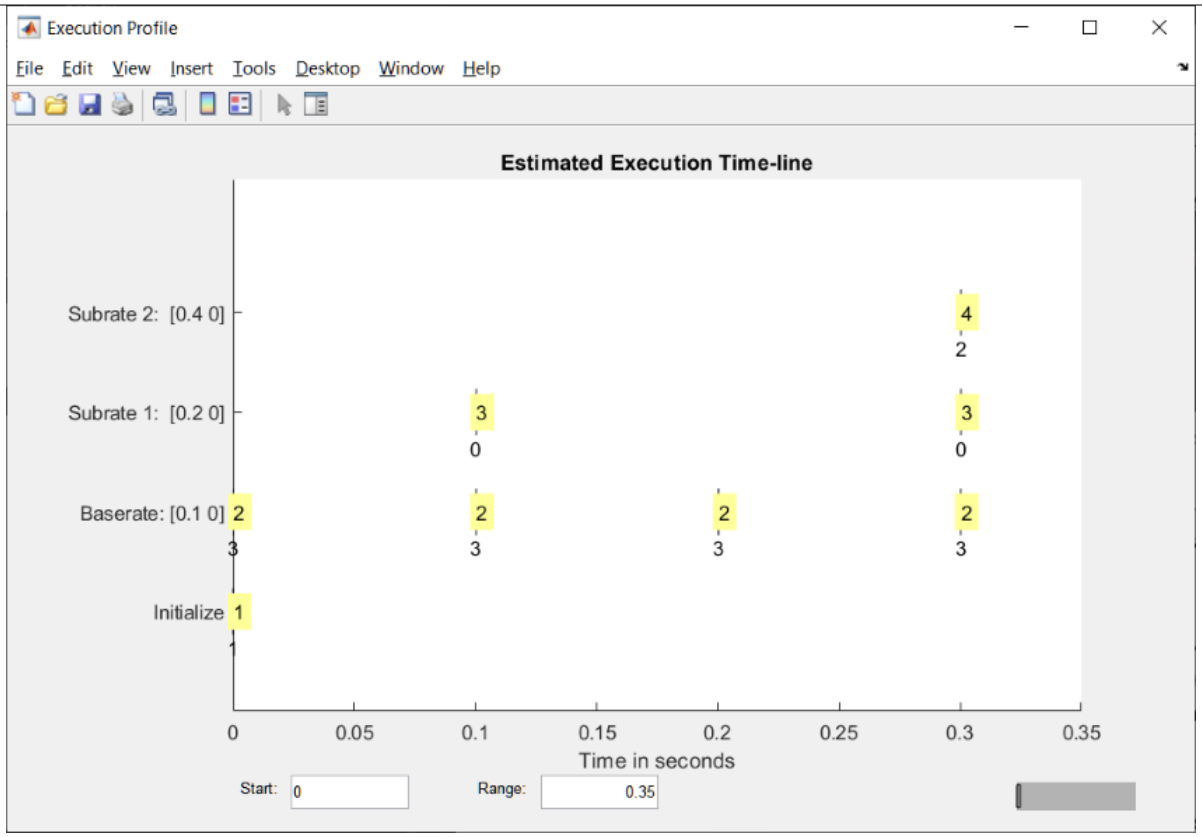
Функция профилирования также позволяет построить циклограмму вызовов шагов расчета. Для этого следует использовать следующую команду MATLAB:

<modelName>_ep.timeline

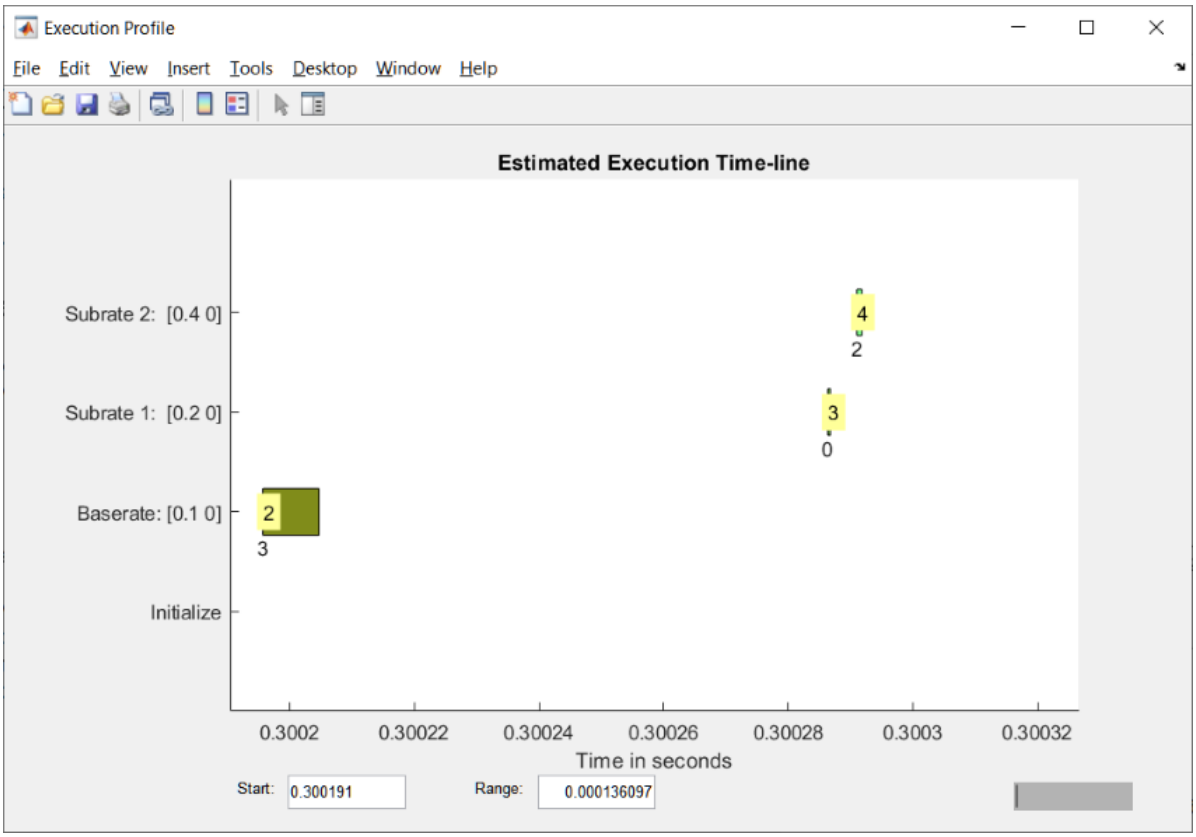
Параметр modelName - строка, содержащая имя модели. Например:

ritmdemo_profiling_ep.timeline

В результате откроется циклограмма:



При увеличении получаем:



На циклограмме можно увидеть вызовы и выполнение каждой секции профилирования на временнОй шкале.
Цифра на желтом фоне означает номер секции.
Цифра под секцией означает ядро процессора, на котором выполнялась секция. Выполнение секции может прерываться более приоритетной секцией - такие секции обозначаются штриховой границей.

Примечание

В режиме профилирования также собирается время, затраченное на инициализацию модели. В отчете инициализации модели соответствует секция **Initialize**.

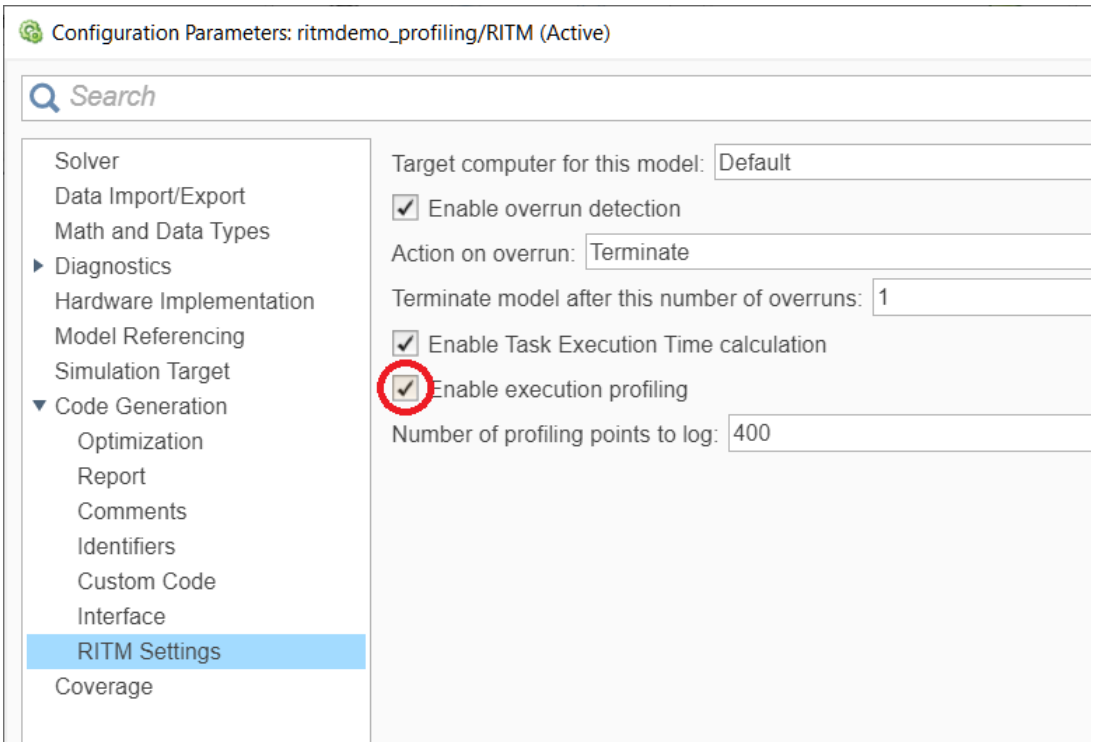
Пример работы

Данный пример демонстрирует настройку модели для профилирования, запуск модели, сбор и анализ результатов.

Откройте модель *ritmdemo_profiling*:

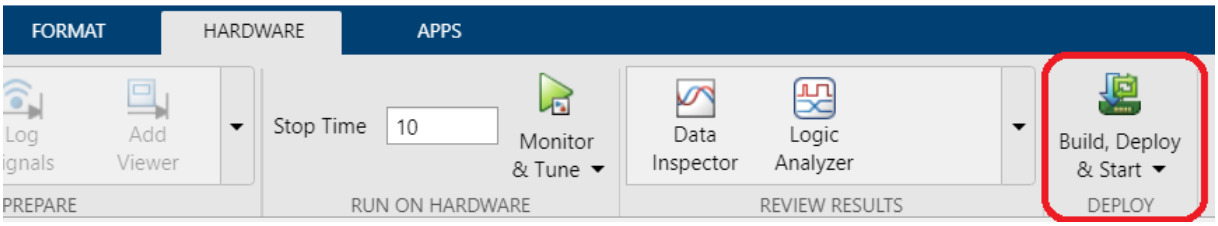
```
open_system('ritmdemo_profiling')
```

Вызовите настройки модели и включите режим профилирования, включив настройку “Enable execution profiling”:



Примените настройки модели и задайте время симуляции 10 секунд.

Нажмите на кнопку **Build, Deploy & Start**:



Подождите 10 секунд и введите в командном окне MATLAB:

```
ritm.profile.getData('ritmdemo_profiling')
```

В результате будет показано окно с результатами профилирования. Для их интерпретации обратитесь к разделу Просмотр и интерпретация результатов

Многозадачное выполнение

Содержание

- Введение
  - Описание технологии
  - Информация в документации Simulink
  - Дополнительная информация
- Пример многозадачного выполнения
- Ограничения

Запуск приложения реального времени с использованием нескольких ядер процессора КПМ “РИТМ”.

Введение

В этом разделе приводится общая информация для начала работы с многозадачным выполнением приложения реального времени.

Описание технологии

Многозадачное выполнение (в документации Simulink называемое Concurrent Execution) позволяет настроить модель Simulink для выполнения на нескольких ядрах многоядерного процессора КПМ “РИТМ”.

Многозадачное выполнение может использоваться для ускорения расчета модели, если выполнение модели не укладывается в шаг расчета на одном ядре.

Механизм Concurrent Execution не является единственным способом многозадачного выполнения моделей Simulink. Два других способа - это использование блока РИТМ Задача ОСРВ и многочастотное (Multirate) выполнение моделей.

В отличие от блока РИТМ Задача ОСРВ, механизм Concurrent Execution не требует разбиения модели на Function-Call Subsystems (что может быть невозможно для некоторых моделей).

В отличие от механизма многочастотного (Multirate) выполнения моделей (когда в модели есть несколько частот дискретизации, которые неявно выполняются на разных ядрах), механизм Concurrent Execution позволяет в явном виде разложить блоки, работающие с одним шагом расчета, на задачи, выполняющиеся на разных ядрах.

В отличие от этих обоих механизмов, Concurrent Execution позволяет пользователю в явном виде указать номер ядра для каждой задачи.

Информация в документации Simulink

Этот раздел документации не является исчерпывающим источником информации о механизме Concurrent Execution. Для более детального понимания рекомендуется изучить раздел документации Multicore Processor Targets со всеми его подразделами.

Дополнительная информация

Также будет полезным посмотреть следующее видео: Многозадачное выполнение

моделей Simulink на многоядерных процессорах

Пример многозадачного выполнения

Обратитесь к разделу документации Примеры и демонстрации многозадачного выполнения, чтобы увидеть пример запуска многозадачного приложения реального времени на нескольких ядрах.

Ограничения

Реализация механизма Concurrent Execution для КПМ “РИТМ” имеет следующие ограничения:

- Не поддерживаются Aperiodic Trigger.
- Поддерживается только один Periodic Trigger. Внутри этого Periodic Trigger может быть любое количество Tasks с привязкой к конкретному ядру.
- Поддерживается только опция Ensure Deterministic Data Transfer.
- Не поддерживается блок РИТМ Время выполнения. Вместо этого блока можно использовать Профилирование выполнения.

Примеры и демонстрации многозадачного выполнения

Содержание

- Предварительные требования
- Разбиение модели на задачи
- Включение Concurrent Execution
- Настройка Concurrent Execution
- Запуск модели

В этом разделе документации приведен пример настройки модели для многозадачного выполнения на нескольких ядрах КПМ “РИТМ”.

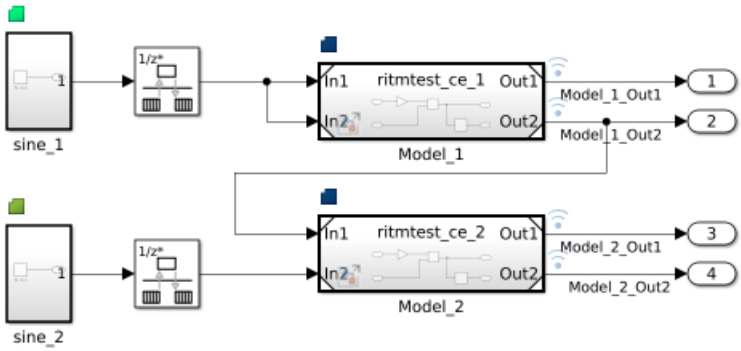
Предварительные требования

Перед началом работы требуется изучить раздел документации Многозадачное выполнение.

Разбиение модели на задачи

Откройте модель.

```
open_system('ritmdemo_ce_top.slx')
```



На верхнем уровне модели находятся две атомарные подсистемы sine_1 и sine_2 и две модели-ссылки ritmdemo_ce_1 и ritmdemo_ce_2. Такая композиция обусловлена требованиями механизма Concurrent Execution, см. Limitations with Multicore Programming in Simulink.

Каждый из этих четырех компонентов в терминологии Concurrent Execution представляет отдельную задачу (Task), которую можно привязать к конкретному ядру КПМ “РИТМ”.

Включение Concurrent Execution

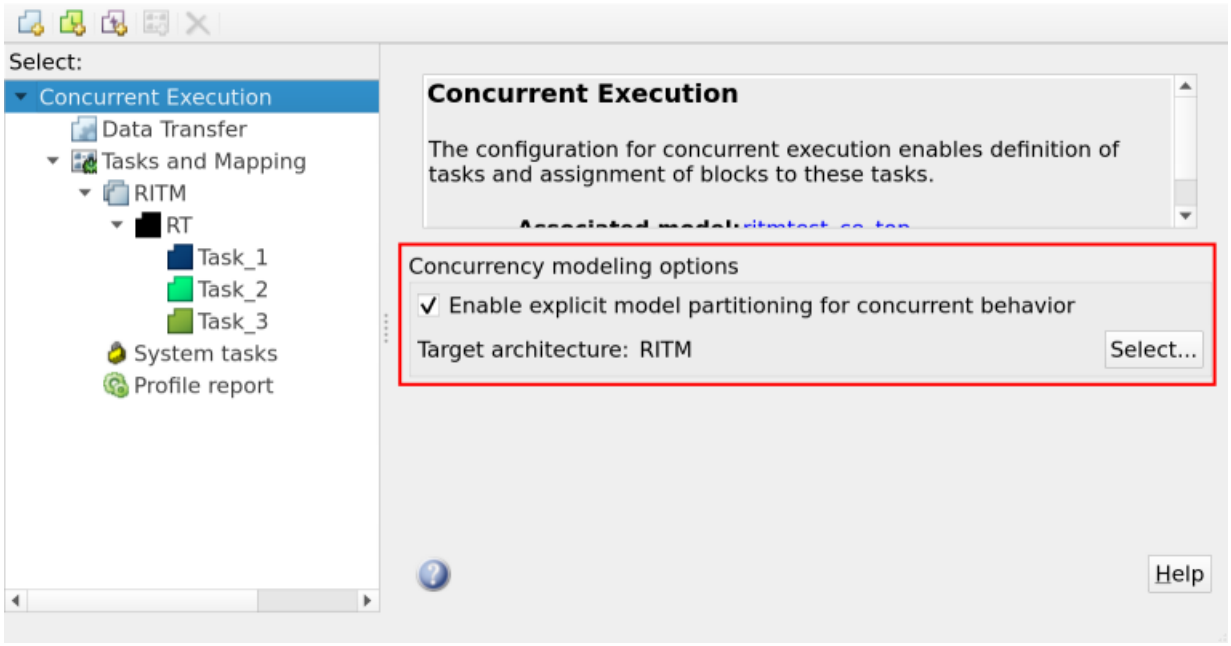
Зайдите в настройки модели и в панели Solver выберите опцию Allow tasks to execute concurrently on target. После этого можно открыть окно настроек Concurrent Execution одним из способов:

- Нажать на кнопку Configure Tasks в панели Solver в настройках модели.
- Нажать на иконку (бейджик) слева вверху любого компонента в модели.

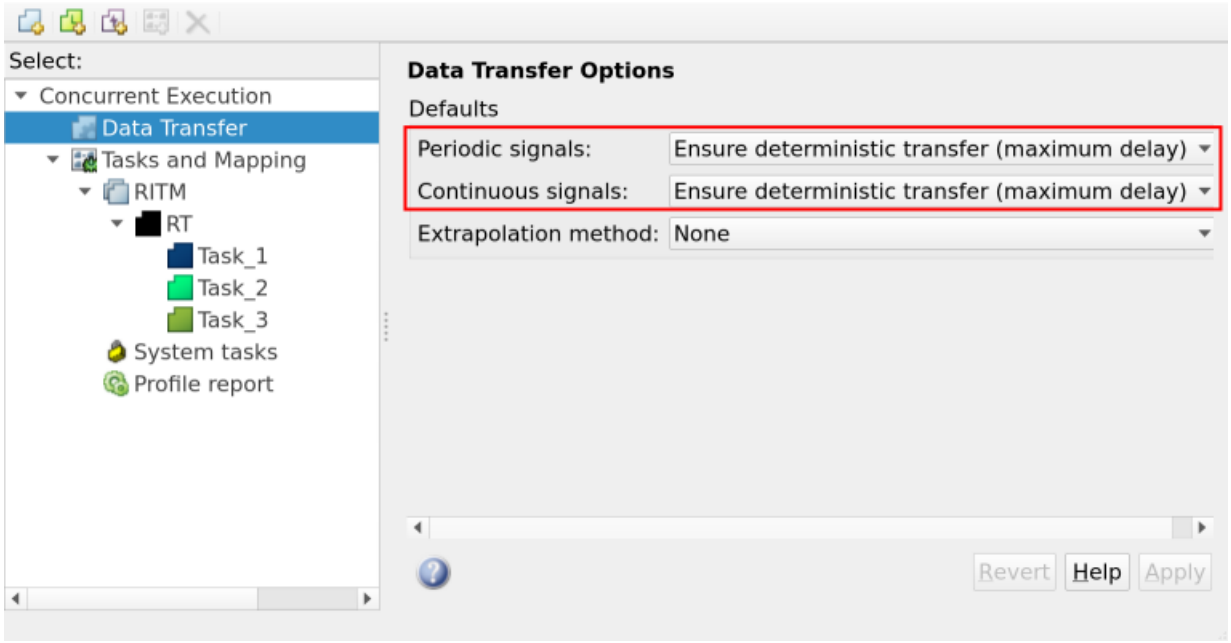
Настройка Concurrent Execution

В открывшемся окне настроек Concurrent Execution настройте следующее.

1. В разделе Concurrent Execution выберите опцию Enable explicit model partitioning и выберите Target Architecture: RITM.

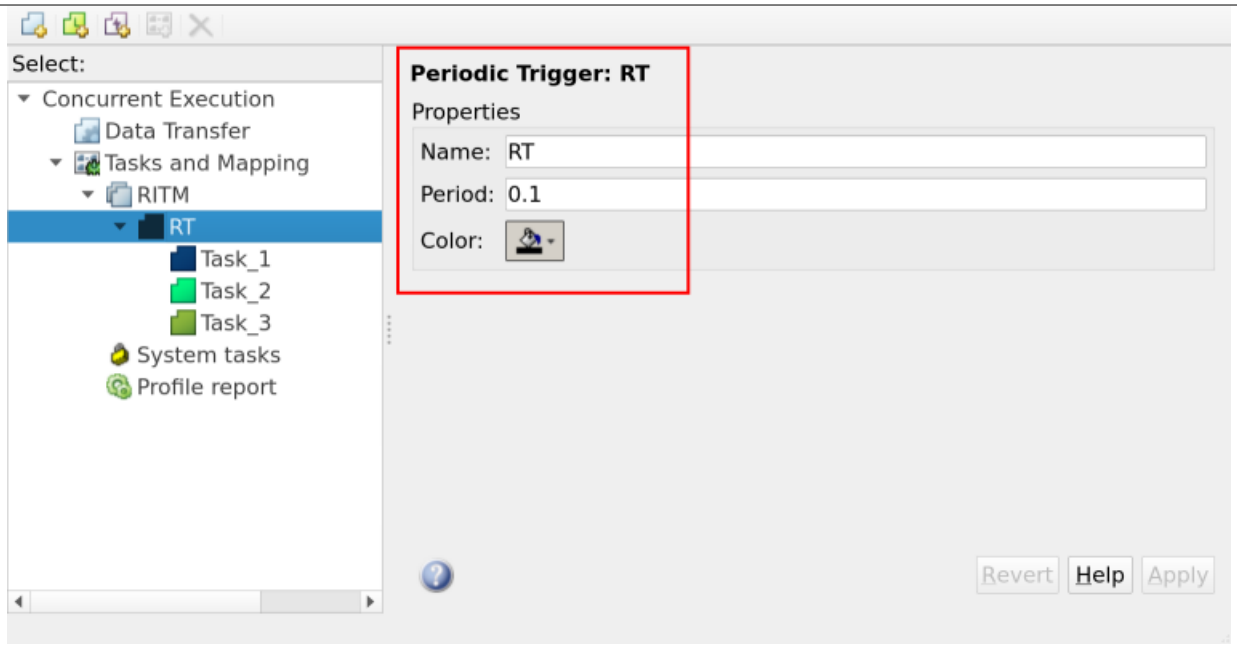


2. В разделе Data Transfer выберите Ensure deterministic transfer.

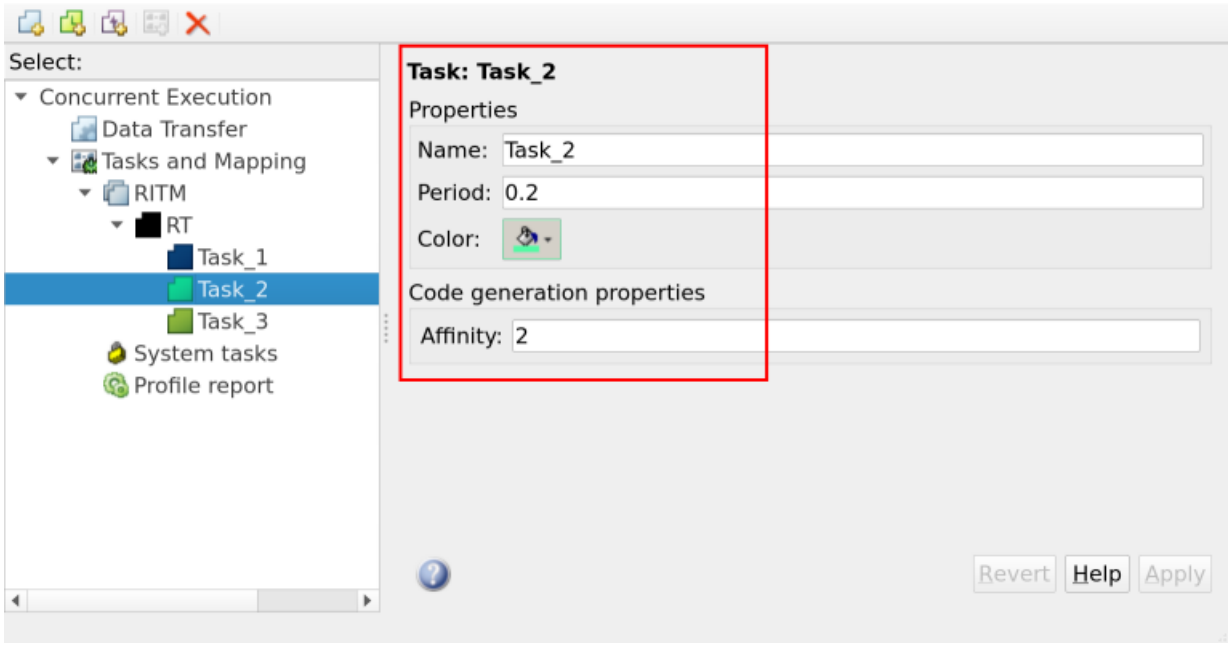


3. В разделе Tasks and Mapping:

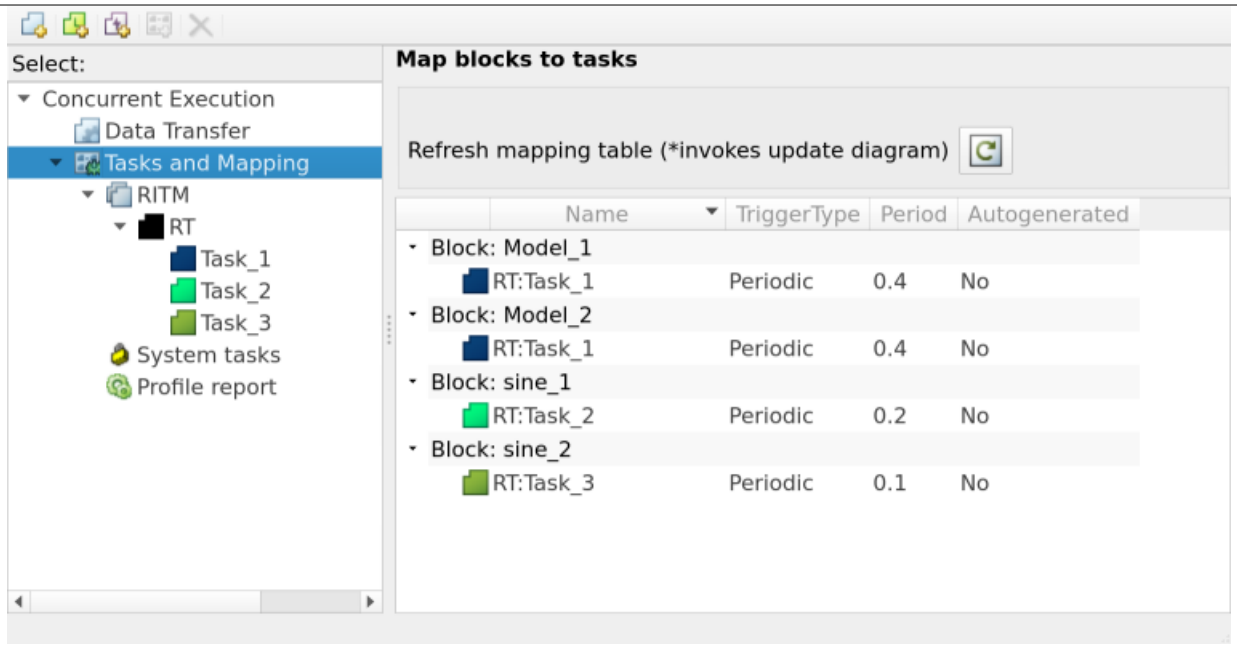
- Создайте ровно один Periodic Trigger и



- Создайте требуемое количество задач (Task) в этом Periodic trigger. Для каждой задачи укажите её имя (Name), период (Period) и привязку к номеру ядра (Affinity). Ядра нумеруются с 0 и параметр Affinity не должен превышать количества ядер в КПМ “РИТМ”.



4. В разделе Tasks and Mapping настройку привязку компонентов модели (например, атомарных подсистем и моделей-ссылок) к созданным задачам.



В этом примере модели-ссылки (Model_1 и Model_2) работают в одной задаче Task_1 с шагом 0.4 с. Атомарная подсистема sine_1 работает в задаче Task_2 с шагом 0.2 с. Атомарная подсистема sine_2 работает в задаче Task_3 с шагом 0.1 с. Задачи Task_1, Task_2 и Task_3 привязаны к ядрам 1, 2 и 3 соответственно.

Запуск модели

После настройки на Concurrent Execution запуск модели ничем не отличается от запуска любой другой модели на КПМ “РИТМ”.

Во время выполнения модели вы можете наблюдать на экране РИТМа, сколько времени (ТЕТ) занимает выполнение каждой задачи.

Для более детального профилирования выполнения, включая визуализацию выполнения на разных ядрах, обратитесь к разделу документации Профилирование выполнения

Интерфейсы пользователя в App Designer

Содержание

- Предварительные требования
- Запуск и остановка модели
- Настройка параметров модели
  - Если параметры заданы в виде чисел в блоках Simulink
  - Если параметры заданы в виде названий переменных
- Отображение сигналов модели
- Пример графического интерфейса пользователя

В этом разделе рассматривается создание графических интерфейсов пользователя в App Designer для управления приложением реального времени, включая запуск и остановку модели, настройку параметров модели и отображение сигналов модели.

Предварительные требования

Предполагается, что пользователь уже знаком с разработкой графических приложений в инструменте App Designer.
Обратитесь к разделу документации *MATLAB -> App Building* для получения дополнительной информации.

В этом разделе документации будут разъяснены только основные команды и API со стороны РИТМа, которые вам потребуются для разработки приложений.
Вопросы интеграции этих API в общее приложение должны быть понятны пользователю, знакомому с App Designer.

Запуск и остановка модели

Для того, чтобы иметь возможность настраивать параметры и визуализировать сигналы модели, используется режим интерактивного выполнения приложения реального времени.

Для **запуска модели** в интерактивном режиме используются следующие команды:

```
open_system(modelName);
set_param(modelName,'SimulationMode','external');
slbuild(modelName);
set_param(modelName, 'SimulationCommand', 'start');
```

Для **проверки состояния** модели используется следующая конструкция:

```
simulationStatus = get_param(modelName, 'SimulationStatus');
if (simulationStatus == "external")
   disp('Model is running')
end
```

Для **остановки модели** используется следующая команда:

```
set_param(modelName, 'SimulationCommand', 'stop');
```

Настройка параметров модели

Для изменения параметров модели в процессе симуляции требуется в настройках модели установить *Code Generation -> Optimization -> Default parameter behaviour* как *Tunable*.

Если параметры заданы в виде чисел в блоках Simulink

Чтобы изменить значение параметра, используется следующая команда:

```
value = 0.5;
set_param([modelName, '/Task01/Sine Wave'], 'Amplitude', num2str(value));
```

Чтобы получить значение параметра, используется следующая команда:

```
value = get_param([modelName, '/Task01/Sine Wave'], 'Amplitude');
```

Если параметры заданы в виде названий переменных

Чтобы изменить значение параметра, используется следующая команда:

```
% Эта команда должна поменять значение переменной в base workspace,
% model workspace или Simulink Data Dictionary.
% См. ниже ссылки на полезные API для этого.
value = 0.5;
% Можно поменять значения сразу нескольких
% переменных выше и "обновить" в приложении реального времени их все сразу.
set_param(modelName, 'SimulationCommand', 'update');
```

Чтобы получить или поменять значение параметра, может использоваться одна из следующих команд.

Для base workspace: см. evalin и Simulink.data.evalinGlobal.
Для model workspace: см. документацию на Simulink.ModelWorkspace.
Для Simulink Data Dictionary: см. документацию на Simulink.data.Dictionary.

Для поиска всех параметров в модели может быть полезна функция Simulink.findVars.

Отображение сигналов модели

Все сигналы, которые требуется визуализировать, должны записываться в модели. Для записи сигнала требуется выделить его в модели и выбрать *Log Selected Signal* во всплывающем меню.
Для получения значений сигналов непосредственно в процессе симуляции используется API Simulation Data Inspector (SDI).

Получить текущий прогон в SDI:

```
sdiRun = Simulink.sdi.getCurrentSimulationRun(modelName);
```

Получить список (имена) всех записываемых сигналов:

```
signals = sdiRun.getAllSignals;
signalNames = {signals.Name}
```

Получить сигнал по его индексу (порядковому номеру в прогоне):

```
sdiSignal = sdiRun.getSignalByIndex(signalIndex);
```

Получить все данные, записанные в сигнале:

```
data = sdiSignal.export;
```

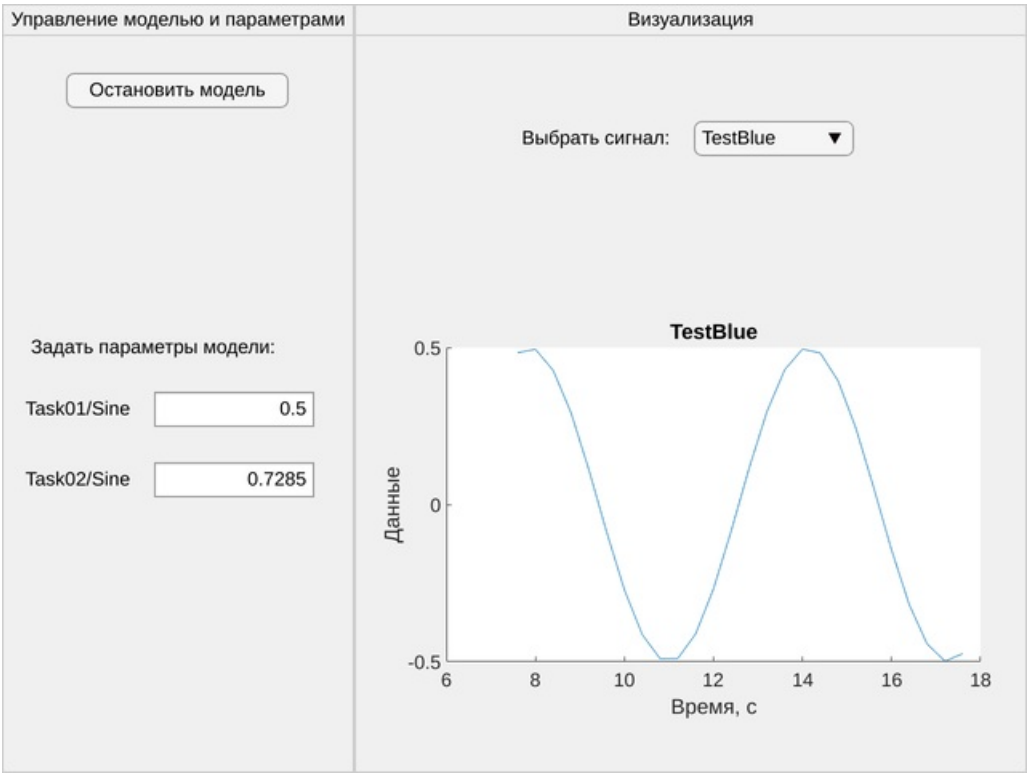
Получить последние duration секунд данных (используется API timeseries):

```
duration = 10;
data = data.getsampleusingtime(data.TimeInfo.End-duration, data.TimeInfo.End);
```

Пример графического интерфейса пользователя

В приложении показан пример простого графического интерфейса пользователя.

```
edit('ritmdemo_appdesigner')
```



Кнопка *Запустить/Остановить модель* управляет моделью и меняет свое назначение в зависимости от статуса модели (запущена или остановлена).

Текстовые поля слева отображают текущие значения некоторых параметров модели и позволяют их менять.

Когда модель запущена, в ниспадающем меню *Выбрать сигнал* появляется список сигналов, записываемых в модели.

После выбора сигнала данные за последние 10 секун начинают отображаться в виде графика.

В приложении запускается таймер, который периодически обновляет значения на графике.

Запуск нейросетей на РИТМе

Содержание

- Предварительные требования
- Общая информация
- Пример генерации независимого C/C++ кода
- Пример генерации C++ кода на oneDNN

В этом разделе рассматривается запуск нейросетей на КПМ РИТМ

Предварительные требования

- Deep Learning Toolbox
- MATLAB Coder Interface for Deep Learning Libraries
- Computer Vision Toolbox Model for YOLO v2 Object Detection (при использовании нейросети YOLO v2)

Общая информация

Для запуска нейросети на РИТМе есть две принципиальные возможности:

1. Генерация C/C++ кода, независимого от сторонних библиотек.
2. Генерация C++ кода, использующего библиотеку oneDNN (ранее называемой MKL-DNN).

Поддерживаемые сети и слои для каждого из вариантов приведены в документации Deep Learning Toolbox

Пример генерации независимого C/C++ кода

Данная демонстрация работает на КПМ РИТМ “как есть”: Generate Generic C/C++ for Sequence-to-Sequence Deep Learning Simulink Models

Пример генерации C++ кода на oneDNN

Данная демонстрация работает на КПМ РИТМ “как есть”: Code Generation for Deep Learning Simulink Model That Performs Lane and Vehicle Detection

Функции и API РИТМ

В данном разделе приводится информация о различных вспомогательных функциях и API РИТМ.

- **ritm.util.getLog**: Получить лог работы приложений на машине.
- **ritm.util.setupEthernet**: Настройка Ethernet интерфейсов на машине.
- **ritm.util.setupTargetGui**: Настройка режима работы графического интерфейса КПМ “РИТМ”.
- **ritm.util.getMatFile**: Получить *.mat* файл с “РИТМ”.
- **ritm.util.setTime**: Установка даты и времени на КПМ “РИТМ”.
- **ritm.util.{buildModel,runModel,stopModel}**: Программное управление моделью
- **ritm.util.isRunning**: Получить статус модели
- **ritm.util.isConnected**: Получить статус подключения к машине

Вспомогательная функция ritm.util.getLog

Функция используется для получения лога работы приложений реального времени с КПМ “РИТМ”. Лог содержит стандартный вывод и вывод ошибок приложений реального времени с момента загрузки машины.

Синтаксис

```
varargout = ritm.util.getLog(profileName)
```

Входные параметры

- 'profileName': имя профиля машины. Если не указать имя профиля, то будет использоваться машина по умолчанию. *Формат - строковый*

Выходные параметры

- 'varargout': путь к тестовому файлу лога на локальном ПК. Если аргумент не указан - имя файла выбирается автоматически и лог открывается в редакторе MATLAB. *Формат - строковый*

Пример использования

```
ritm.util.getLog("RITM 1");
```

Вспомогательная функция ritm.util.setupEthernet

Данная функция используется для настройки доступных Ethernet интерфейсов на КПМ “РИТМ”.

Синтаксис

```
ritm.util.setupEthernet(profileName, interfaceID, interfaceAddress)
```

Входные параметры

- 'profileName': имя профиля машины. *Формат - строковый*
- 'interfaceID': номер Ethernet интерфейса. Номера интерфейсов подписаны на КПМ “РИТМ”. *Формат - строковый*
- 'interfaceAddress': желаемый IP адрес Ethernet интерфейса. Также есть возможность задать маску интерфейса в нотации CIDR. *Формат - строковый*

Пример использования

```
ritm.util.setupEthernet("RITM 1", "1", "127.0.0.1");
```

Вспомогательная функция ritm.util.setupTargetGui

Данная функция используется для настройки режима работы графического интерфейса КПМ “РИТМ”.

Синтаксис

```
ritm.util.setupTargetGui(profileName, option, value)
```

Входные параметры

- 'profileName': имя профиля машины. *Формат - строковый*
- 'option': параметр для настройки. *Формат - строковый*
- 'value': значение параметра. *Формат - строковый*

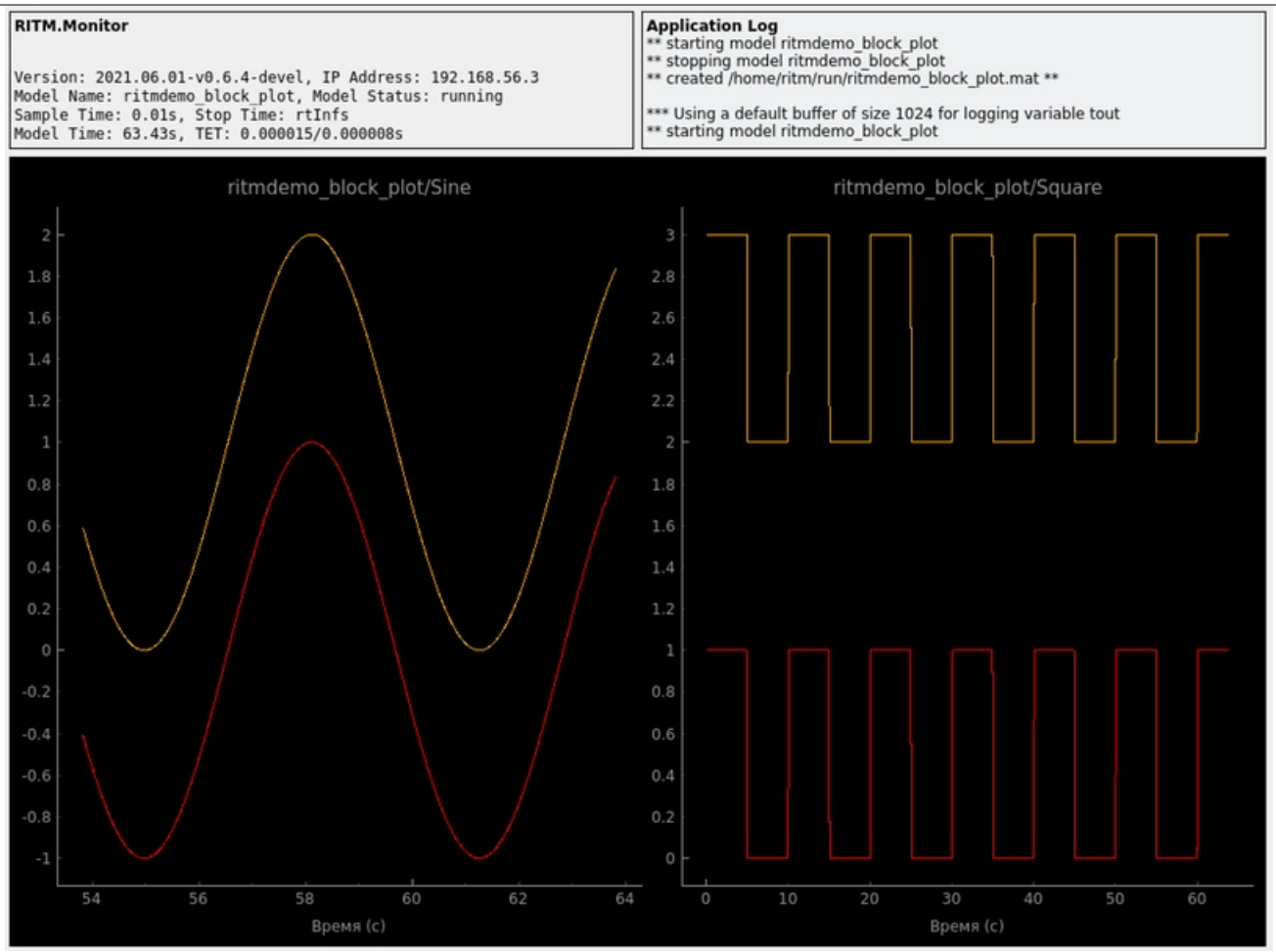
Допустимые параметры и их значения

Параметр	Значение	Описание
gui-mode	graphics-mode	Режим с возможностью отображения графиков
gui-mode	txt-mode	Текстовый режим

Пример использования

Настроить режим на отображение графиков для машины *RITM 1*:

```
ritm.util.setupTargetGui("RITM 1", "gui-mode", "graphics-mode");
```



Настроить режим на отображение только текста для машины *RITM 1*:

```
ritm.util.setupTargetGui("RITM 1", "gui-mode", "txt-mode");
```

RITM.Monitor

```
Version: 2021.06.01-v0.6.4-devel, IP Address: 192.168.56.3
Model Name: ritmdemo_block_plot, Model Status: running
Sample Time: 0.01s, Stop Time: rtInfs
Model Time: 8.69s, TET: 0.000016/0.000006s
```

Application Log

```
** stopping model ritmdemo_block_plot
** created /home/ritm/run/ritmdemo_block_plot.mat **

** starting model ritmdemo_block_plot
** stopping model ritmdemo_block_plot
** created /home/ritm/run/ritmdemo_block_plot.mat **

** starting model ritmdemo_block_plot
** stopping model ritmdemo_block_plot
** created /home/ritm/run/ritmdemo_block_plot.mat **

** starting model ritmdemo_block_plot
** stopping model ritmdemo_block_plot
** created /home/ritm/run/ritmdemo_block_plot.mat **

*** Using a default buffer of size 1024 for logging variable tout
** starting model ritmdemo_block_plot
** stopping model ritmdemo_block_plot
*** Log variable rt_tout has wrapped 8 times
using a circular buffer of size 1024
*** To avoid wrapping, explicitly specify a
buffer size of 9050 in your Simulink model
by adding OPTS="-DDEFAULT_BUFFER_SIZE=9050"
as an argument to the ConfigSet MakeCommand
parameter
** created /home/ritm/run/ritmdemo_block_plot.mat **

*** Using a default buffer of size 1024 for logging variable tout
** starting model ritmdemo_block_plot
** stopping model ritmdemo_block_plot
*** Log variable rt_tout has wrapped 2 times
using a circular buffer of size 1024
*** To avoid wrapping, explicitly specify a
buffer size of 2343 in your Simulink model
by adding OPTS="-DDEFAULT_BUFFER_SIZE=2343"
as an argument to the ConfigSet MakeCommand
parameter
** created /home/ritm/run/ritmdemo_block_plot.mat **

*** Using a default buffer of size 1024 for logging variable tout
** starting model ritmdemo_block_plot
```

Вспомогательная функция ritm.util.getMatFile

Функция используется для выгрузки с КПМ “РИТМ” файлов **.mat**.

Синтаксис

```
ritm.util.getMatFile(profileName, matFileName, targetDirectory)
```

Входные параметры

- 'profileName': имя профиля машины. *Формат - строковый*
- 'matFileName': имя **.mat** файла. *Формат - строковый*
- 'targetDirectory': путь сохранения файла. Значение по умолчанию - текущая директория. *Формат - строковый*

Пример использования

```
ritm.util.getMatFile('RITM 1', 'myFile.mat', 'C:\Users\user\myFolder');
```

Результатом будет выгрузка файла *myFile.mat* в директорию *C:|Users|user|myFolder*.

или

```
ritm.util.getMatFile('RITM 1', 'myFile.mat');
```

Результатом будет выгрузка файла *myFile.mat* в текущую директорию.

Вспомогательная функция ritm.util.setTime

Функция используется для установки даты и времени на КПМ “РИТМ”.

Синтаксис

```
ritm.util.setTime(profileName, time)
```

Входные параметры

- 'profileName': имя профиля машины. *Формат - строковый*
- 'time': время для установки в формате UNIX Time. *Формат - числовой*

Пример использования

```
ritm.util.setTime('RITM 1', posixtime(datetime));
```

В результате дата и время на КПМ “РИТМ” будут установлены в соответствии с датой и временем хост компьютера.

Вспомогательная функция ritm.util.isRunning

Функция проверяет статус выполнения модели Simulink (запущена или остановлена) через командную строку MATLAB.

Синтаксис

```
ritm.util.isRunning(modelName)
```

Входные параметры

- 'modelName': имя модели Simulink, для которой проверяется статус выполнения. *Формат - строковый*

Пример использования

```
ritm.util.isRunning('ritmdemo_block_plot');
```

Вспомогательная функция ritm.util.isConnected

Функция проверяет статус подключения к КПМ “РИТМ”.

Синтаксис

```
ritm.util.isConnected(profileName)
```

Входные параметры

- 'profileName': имя профиля машины. *Формат - строковый*

Пример использования

```
ritm.util.isConnected('RITM 1');
```

Программное управление моделями на РИТМ

В данном разделе приводится информация о функциях для программного управления моделью.

- **ritm.util.buildModel**: Компиляция моделей из командной строки.
- **ritm.util.runModel**: Запуск моделей из командной строки.
- **ritm.util.stopModel**: Остановка моделей из командной строки.

Вспомогательная функция ritm.util.buildModel

Функция используется для компиляции модели в режиме независимого выполнения из командной строки MATLAB.

Синтаксис

```
ritm.util.buildModel(modelName)
```

Модель должна быть открыта в Simulink. Например, командами open_system или load_system.

Входные параметры

- 'modelName': имя компилируемой модели. *Формат - строковый*

Пример использования

```
load_system('ritmdemo_profiling');
ritm.util.buildModel('ritmdemo_profiling');
```

Вспомогательная функция ritm.util.runModel

Функция используется для запуска модели на КПМ “РИТМ” из командной строки MATLAB.

Синтаксис

```
ritm.util.runModel(modelName, mode)
```

Модель должна быть открыта в Simulink. Например, командами open_system или load_system.

Входные параметры

- 'modelName': имя запускаемой модели. *Формат - строковый*
- 'mode': параметр для выбора режима запуска модели. *Формат - строковый*

Параметр	Значение	Описание
mode	standalone	Запуск модели в режиме независимого выполнения
mode	monitor	Запуск модели в режиме **Monitor & Tune** (интерактивное выполнение)

Пример использования

```
load_system('ritmdemo_profiling');
ritm.util.buildModel('ritmdemo_profiling');
ritm.util.runModel('ritmdemo_profiling', 'standalone');
```

или

```
open_system('ritmdemo_profiling');
ritm.util.runModel('ritmdemo_profiling', 'monitor');
```

В случае запуска модели в режиме **Monitor & Tune** сборка происходит автоматически, поэтому команда ritm.util.buildModel не требуется.

При запуске модели автоматически останавливается выполнение модели, работающей в данный момент на РИТМе.

Вспомогательная функция ritm.util.stopModel

Функция используется для остановки модели на КПМ “РИТМ” из командной строки MATLAB.

Синтаксис

```
ritm.util.stopModel(modelName)
```

Входные параметры

- 'modelName': имя останавливаемой модели. *Формат - строковый*

Пример использования

```
ritm.util.stopModel('ritmdemo_profiling');
```

Блоки для работы с графическим интерфейсом

Содержание

- Библиотека блоков РИТМ График

Библиотека блоков РИТМ График

Библиотека блоков *РИТМ График* для отображения графиков на мониторе КПМ РИТМ.

Библиотека блоков РИТМ График

Содержание

- Описание
- Библиотека блоков
- Блок РИТМ График
  - Маска блока
  - Входы блока
  - Параметры блока
  - Примечания об использовании

Описание

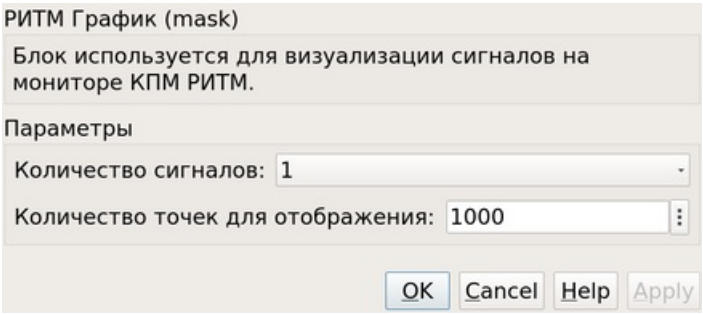
Библиотека блоков для отображения графиков на мониторе КПМ РИТМ.
Информация о настройке графического интерфейса.

Библиотека блоков

Блок РИТМ График

Блок используется для визуализации сигналов на мониторе КПМ РИТМ. Каждому блоку соответствует свой график на мониторе КПМ РИТМ, который может содержать до 8 сигналов.

Маска блока



Входы блока

Блок имеет переменное число входов в зависимости от значения параметра **Количество сигналов**.

- **Данные**: Значение типа *double* для отображения.

Параметры блока

- **Количество сигналов**: Задаёт количество сигналов для отображения. Каждый вход соответствует своему сигналу на графике.

- **Количество точек для отображения**: Задаёт количество точек на графике для отображения.

Примечания об использовании

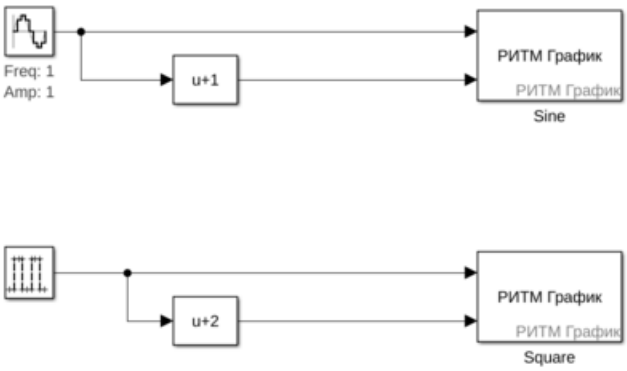
Дискретизация сигнала на графике соответствует шагу расчёта блока. При этом параметр **Количество точек для отображения** задает количество последних точек, которые надо отображать на графике.
По мере поступления новых точек на график, старые точки будут отбрасываться с графика. Подпись на окне графика соответствует названию (подписи) блока *РИТМ График*.

Для эффективного управления данными на графике перед блоком *РИТМ График* имеет смысл использовать стандартные блоки *Rate Transition* и *Data Type Conversion*.

Примеры и демонстрации РИТМ График

Пример использования блока РИТМ График.

```
open_system('ritmdemo_block_plot.slx')
```



Блоки для работы с ОСРВ “РИТМ”

Содержание

- Библиотека блоков РИТМ Время выполнения
- Библиотека блоков РИТМ Unix Время
- Библиотека блоков РИТМ Задержка
- Библиотека блоков РИТМ Задача ОСРВ
- Библиотека блоков сериализации шин Simulink

Библиотека блоков РИТМ Время выполнения

Библиотека блоков для получения времени выполнения шага расчёта на КПМ РИТМ.

Библиотека блоков РИТМ Unix Время

Библиотека блоков для получения времени, прошедшего по часам КПМ РИТМ, с начала эпохи Unix.

Библиотека блоков РИТМ Задержка

Библиотека блоков для перевода потока в спящий режим на заданное число микросекунд.

Библиотека блоков РИТМ Задача ОСРВ

Библиотека блоков для для запуска частей модели в разных потоках с пользовательским приоритетом исполнения.

Библиотека блоков сериализации шин Simulink

Библиотека блоков для сериализации и десериализации шин Simulink для передачи их по цифровым интерфейсам.

Библиотека блоков РИТМ Время выполнения

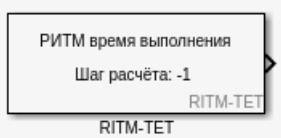
Содержание

- Описание
- Библиотека блоков
- Блок РИТМ Время выполнения
  - Маска блока
  - Выходы блока
  - Параметры блока
  - См. также

Описание

Библиотека блоков для получения времени выполнения шага расчёта на КПМ РИТМ.

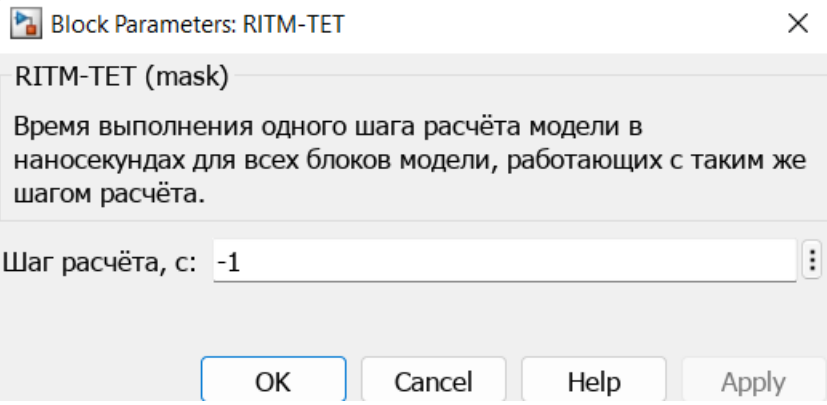
Библиотека блоков



Блок РИТМ Время выполнения

Время выполнения одного шага расчёта модели в наносекундах для всех блоков модели, работающих с таким же шагом расчёта.

Маска блока



Выходы блока

- **Время выполнения**: Суммарное время в наносекундах, которое заняло выполнение всех операций, тип данных *uint32*.

Параметры блока

- **Шаг расчёта**: Шаг расчёта, суммарное время выполнения операций которого будет измеряться.

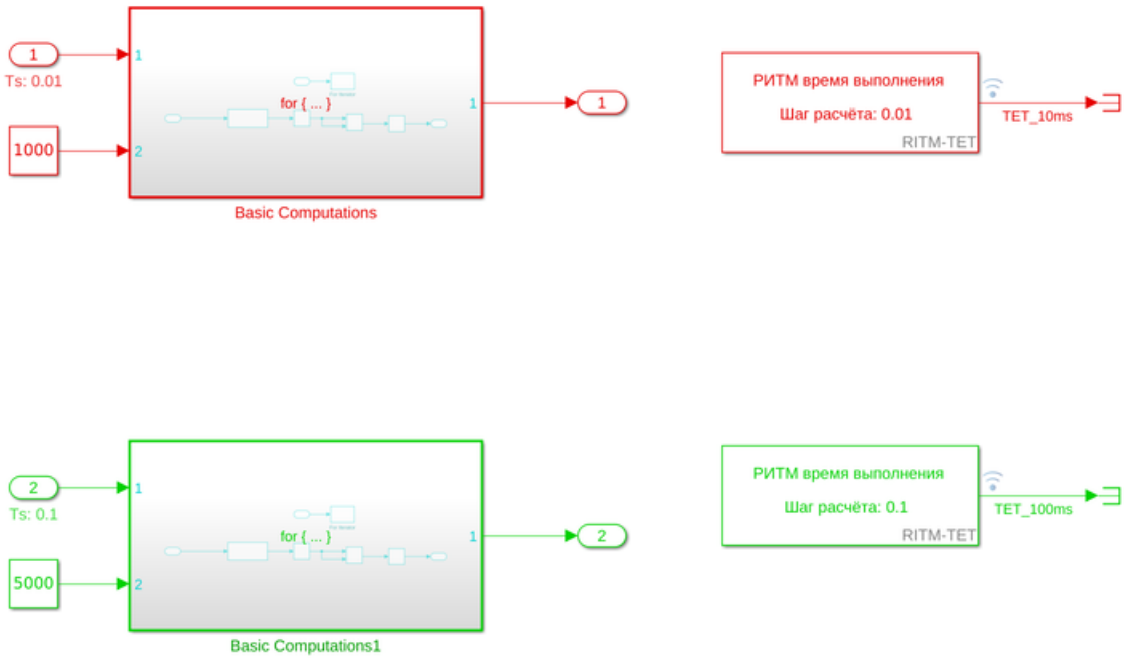
См. также

Профилирование выполнения.

Примеры и демонстрации РИТМ Время выполнения

Пример использования блока РИТМ Время выполнения.

```
open_system('ritmdemo_block_tet.slx')
```



Библиотека блоков РИТМ Unix Время

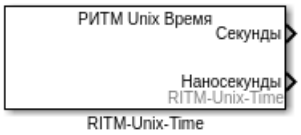
Содержание

- Описание
- Библиотека блоков
- Блок РИТМ Unix Время
  - Маска блока
  - Выходы блока
  - Параметры блока

Описание

Библиотека блоков для получения времени, прошедшего по часам КПМ РИТМ, с начала эпохи Unix.

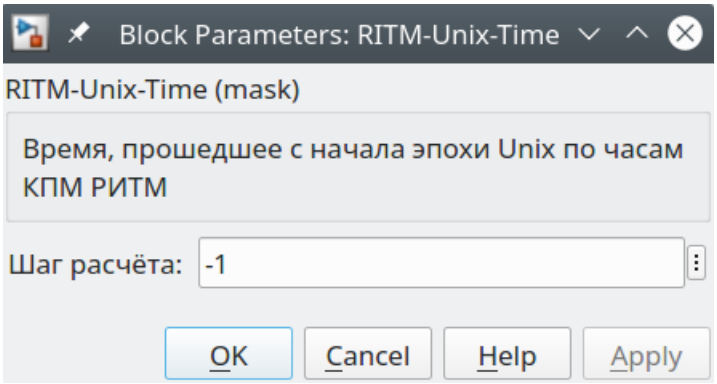
Библиотека блоков



Блок РИТМ Unix Время

Блок используется для получения времени, прошедшего по часам КПМ РИТМ, с начала эпохи Unix.

Маска блока



Выходы блока

- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*.

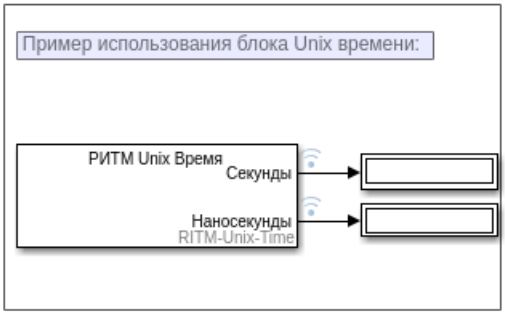
Параметры блока

- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах, с которым будет получаться значение текущего времени.

Примеры и демонстрации РИТМ Unix Время

Пример использования блока РИТМ Unix Время.

```
open_system('ritmdemo_block_unix_time.slx')
```

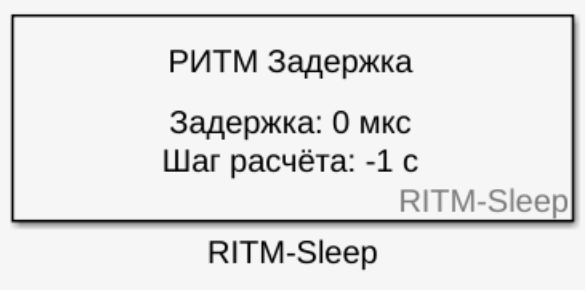


Библиотека блоков РИТМ Задержка

Содержание

- Библиотека блоков
- Блок РИТМ Задержка
  - Маска блока
  - Параметры блока

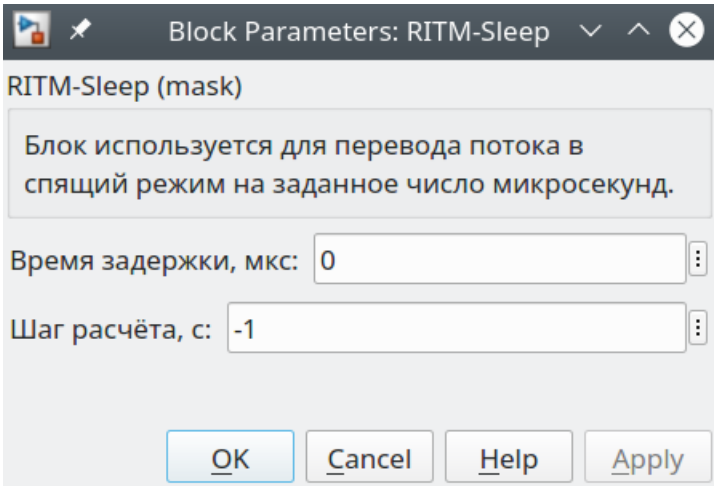
Библиотека блоков



Блок РИТМ Задержка

Блок используется для перевода потока в спящий режим на заданное число микросекунд.

Маска блока



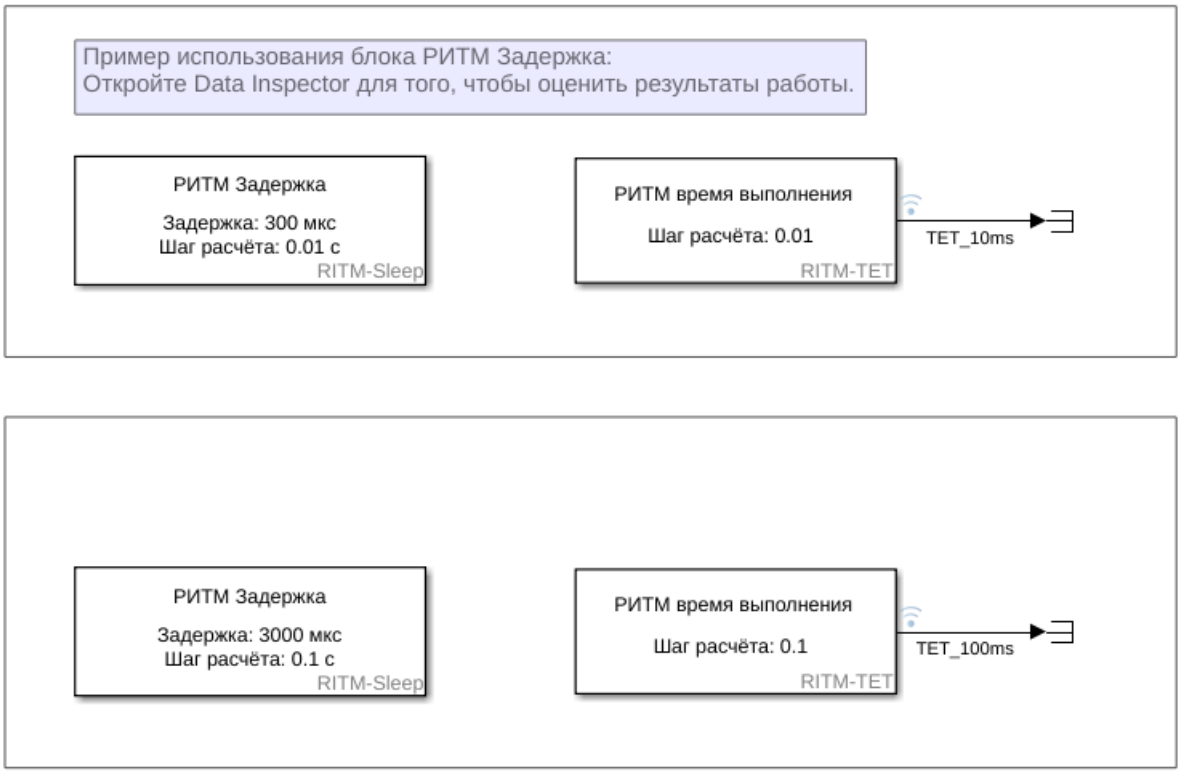
Параметры блока

- **Время задержки, мкс:** Число микросекунд, на которое будет приостановлен поток.
- **Шаг расчёта, с**: Позволяет задать шаг расчёта в секундах, с которым будет вызываться функция задержки.

Примеры и демонстрации РИТМ Задержка

Пример использования блока РИТМ Задержка.

```
open_system('ritmdemo_block_sleep.slx')
```



Библиотека блоков РИТМ Задача ОСРВ

Содержание

- Описание
  - Область применения
- Библиотека блоков
- Блок РИТМ Задача ОСРВ
- Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Указания по моделированию

Описание

Библиотека блоков для работы с задачами (потоками) ОСРВ “РИТМ”.

Область применения

Блок РИТМ Задача ОСРВ рекомендуется применять для:

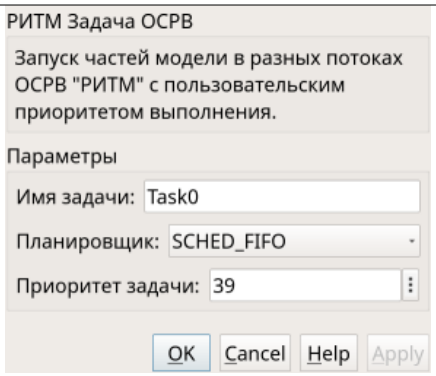
- Ускорения расчета модели на нескольких ядрах
- Организации приема-передачи данных в отдельных потоках
- Распараллеливания вычислений

Библиотека блоков

Блок РИТМ Задача ОСРВ

Блок РИТМ Задача ОСРВ предназначен для запуска частей модели в разных потоках ОСРВ “РИТМ” с пользовательским приоритетом выполнения.

Маска блока



Входы блока

- **Шаг расчёта**: Для задания шага расчёта создаваемого потока используется шаг расчёта (Sample Time) сигнала, приходящего на этот входной порт. Любые другие характеристики (значение самого сигнала, тип данных, размерность и т.п.) не влияют на симуляцию или генерацию кода.

Выходы блока

- **Вызов**: Выход типа “Function-Call” для подключения блока “Function-Call Subsystem”. Подключенный блок содержит в себе алгоритм, выполняющийся в отдельной задаче.

Параметры блока

- **Имя задачи**: Имя создаваемого потока. Должно быть валидным именем функции в Си. Должно быть уникальным для каждого блока РИТМ Задача ОСРВ в модели.

- **Планировщик**: Тип планировщика.

Доступны 2 планировщика: SCHED_FIFO и SCHED_OTHER:
- SCHED_FIFO: Планировщик Linux для задач в реальном времени, высокоприоритетные процессы реального времени могут вытеснять низкоприоритетные процессы реального времени
- SCHED_OTHER: стандартный планировщик Linux с разделением времени для процессов, работающих не в реальном времени

Рекомендуется применять планировщик SCHED_FIFO, так как данный планировщик позволяет задавать приоритет создаваемого потока.

- **Приоритет задачи**: Приоритет создаваемого потока. Более высокое значение означает более высокий приоритет. Любое значение от 1 до 39. Опция доступна только для планировщика SCHED_FIFO. Не рекомендуется создавать два потока с одинаковым приоритетом.

Указания по моделированию

Блок РИТМ Задача ОСРВ является *асинхронным* блоком в терминах Simulink. В связи с этим при использовании этого блока надо учитывать аспекты работы с асинхронными блоками. Дополнительная информация доступна в документации Simulink Coder:

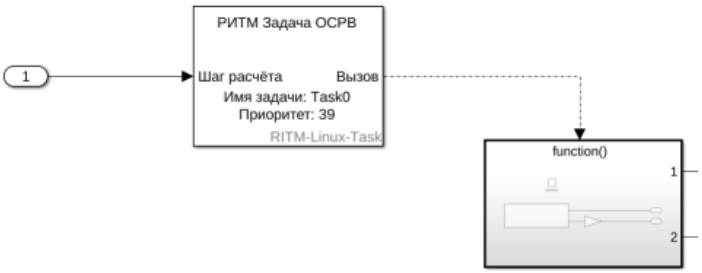
- Asynchronous Events
- Rate Transitions and Asynchronous Blocks

Наиболее важные аспекты работы с блоком следующие:

- Шаг расчёта для асинхронного блока может быть только наследованным. Таким образом, блок РИТМ Задача ОСРВ наследует шаг расчёта от шага расчета сигнала на своём входном порту “Шаг расчёта”. В дальнейшем этот шаг расчёта используется для задачи в

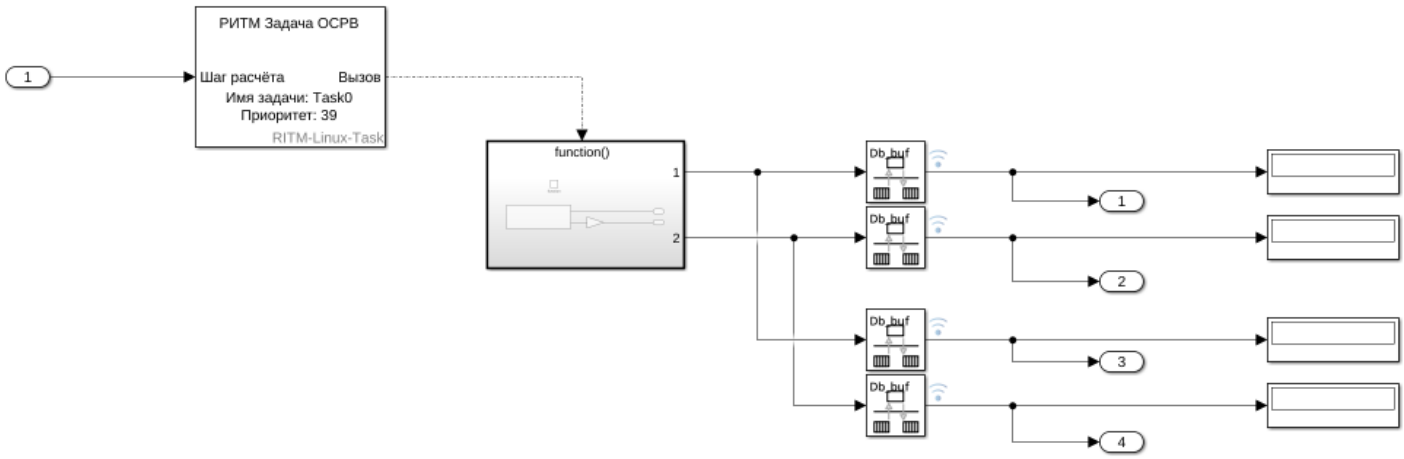
создаваемом потоке.

- Блок РИТМ Задача ОСРВ генерирует на выходе сигнал типа Function-Call. Его поведение схоже со стандартным блоком Function-Call Generator. Для запуска части модели в отдельном потоке требуется применять следующую конструкцию:



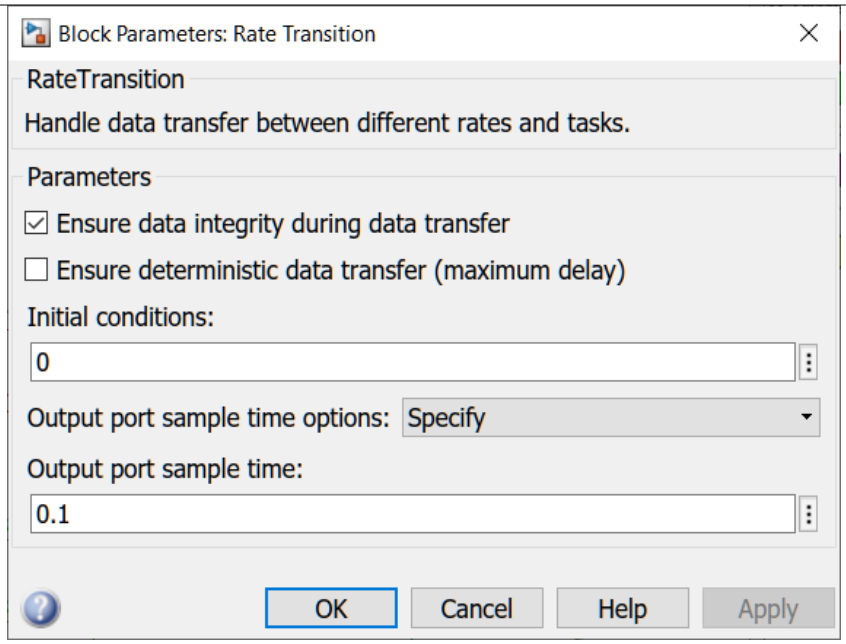
Необходимая часть модели для работы в отдельном потоке помещена в Function-Call Subsystem.

Для работы с выходными сигналами подключенной Function-Call Subsystem **необходимо** использовать блоки Rate Transition:



Применение блоков Rate Transition позволяет работать с выходами блока Function-Call Subsystem с шагом расчёта, необходимым пользователю (переводя сигналы из асинхронных в периодические в модели). Также требуется применять Rate Transition и для сигналов, поступающих на входы Function-Call Subsystem, за исключением констант.

Рекомендуемые настройки блока Rate Transition:



Для изменения выходного шага расчета требуется задать параметр Output port sample time.

В том случае, если требуется создать несколько вызовов подсистем внутри одного потока, требуется применить блок Function-Call Split.

Дополнительная информация и примеры использования блока приведены на странице Примеры и демонстрации РИТМ Задача ОСРВ

Примеры и демонстрации РИТМ Задача ОСРВ

Содержание

- Прием и передача данных по протоколу UDP
- Конвейеризация расчетов
  - Замечания по работе

Прием и передача данных по протоколу UDP

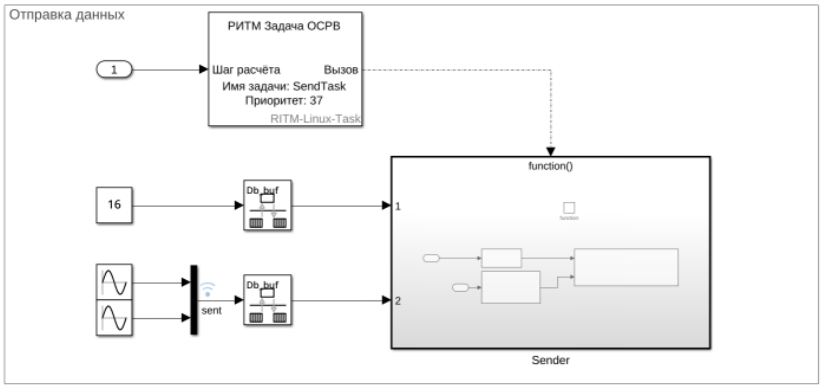
В демонстрации используется библиотека блоков RITM UDP.

Откройте модель.

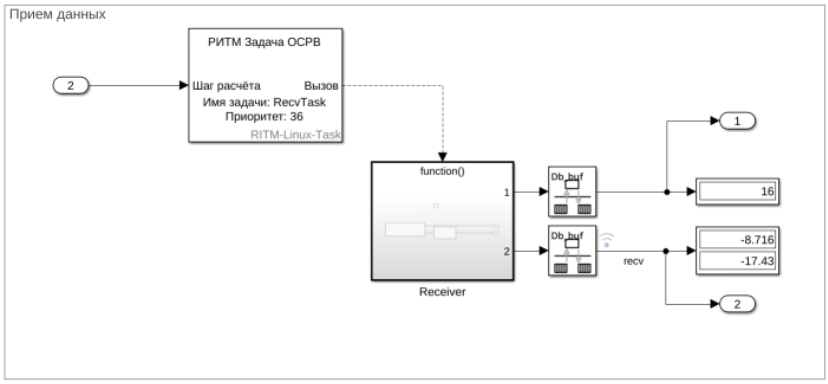
```
open_system('ritmdemo_block_linuxtask_udp')
```

Данная модель содержит два региона:

- Передатчик:



- Приемник:



Для передачи данных по UDP создается поток с именем SendTask, а для приема - поток с именем RecvTask.

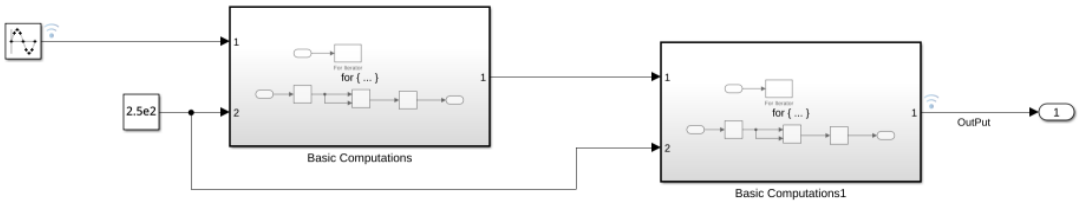
Таким образом, пользователь может вынести обмен данными в отдельный поток и назначить ему пониженный приоритет.

Конвейеризация расчетов

Цель демонстрации - ознакомление с приемом разбиения вычислений по потокам.

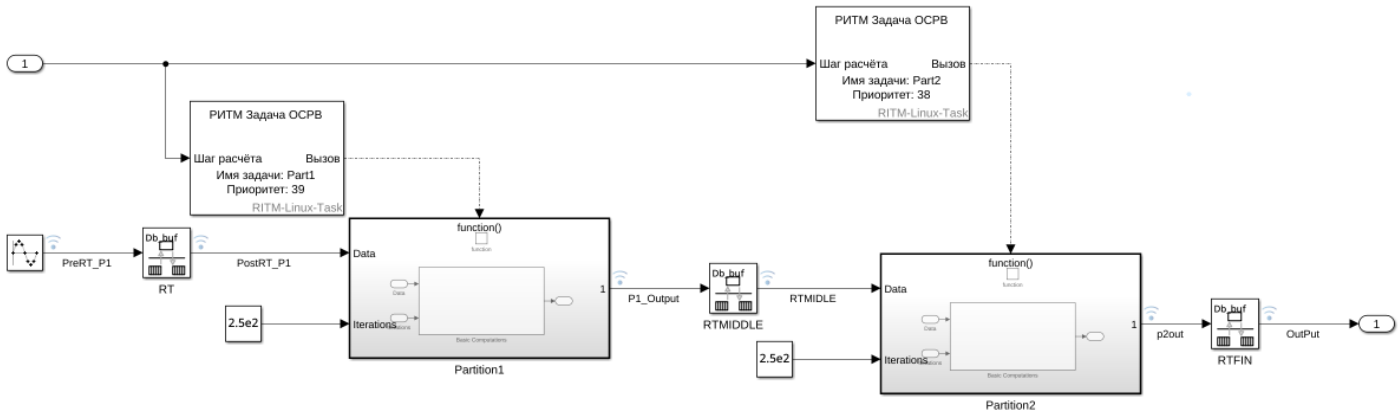
В качестве примера рассмотрим модель.

```
open_system('ritmdemo_block_linuxtask_partition_start')
```



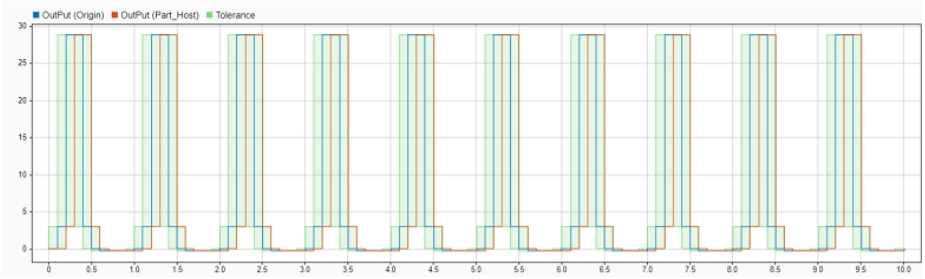
Вычисления осуществляются в одном потоке и на одном ядре. Таким образом, возникает вероятность переполнения шага расчета. Применяя блок РИТМ Задача ОСРВ, можно разбить вычисления на несколько потоков и запускать эти потоки на различных ядрах, как показано в модели:

```
open_system('ritmdemo_block_linuxtask_partition_ref')
```



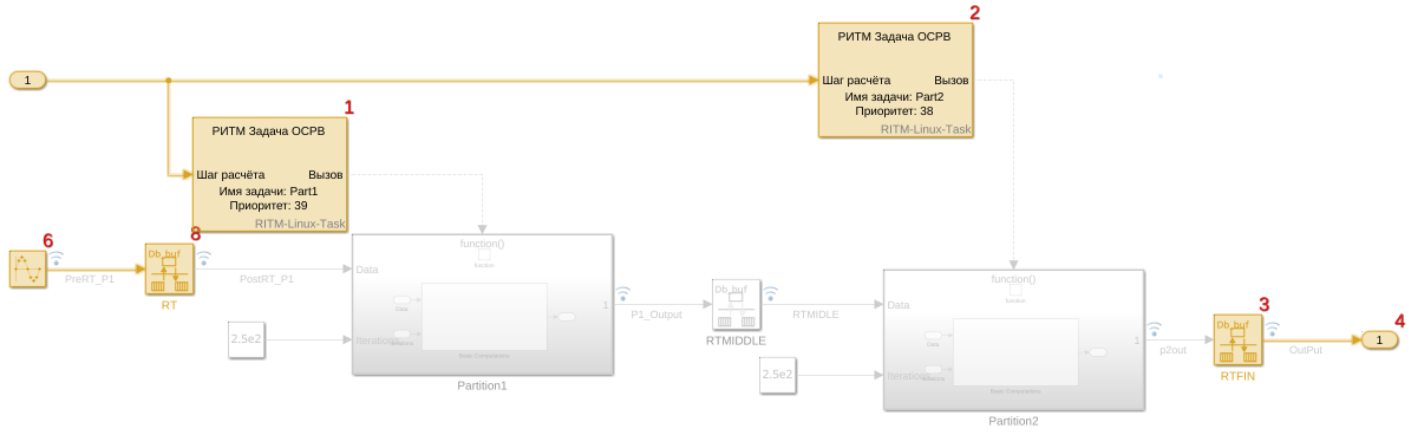
Замечания по работе

Применение данной конструкции приводит к появлению задержки в модели. Рассмотрим результаты симуляции моделей ritmdemo_block_linuxtask_partition_start (оригинальная модель) и ritmdemo_block_linuxtask_partition_ref (конвейеризированные вычисления):

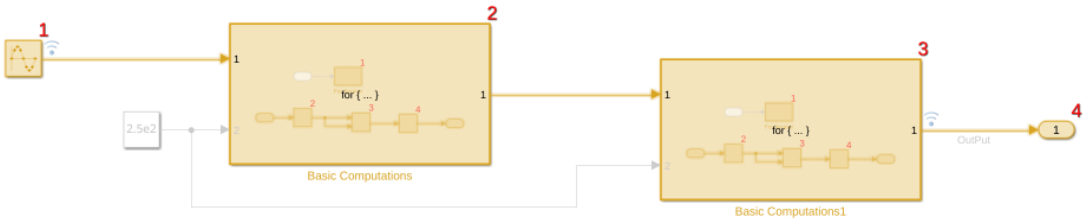


Конвейризация приводит к возникновению дополнительных транспортных задержек - это известный компромисс между пропускной способностью (throughput) и задержкой (latency) системы.

Видно, что конвейеризированные вычисления задержаны относительно оригинальных на 1 такт. Данное поведение вызвано применением блоков РИТМ Задача ОСРВ и измененным порядком выполнения блоков в модели. Рассмотрим порядок выполнения блоков в модели:

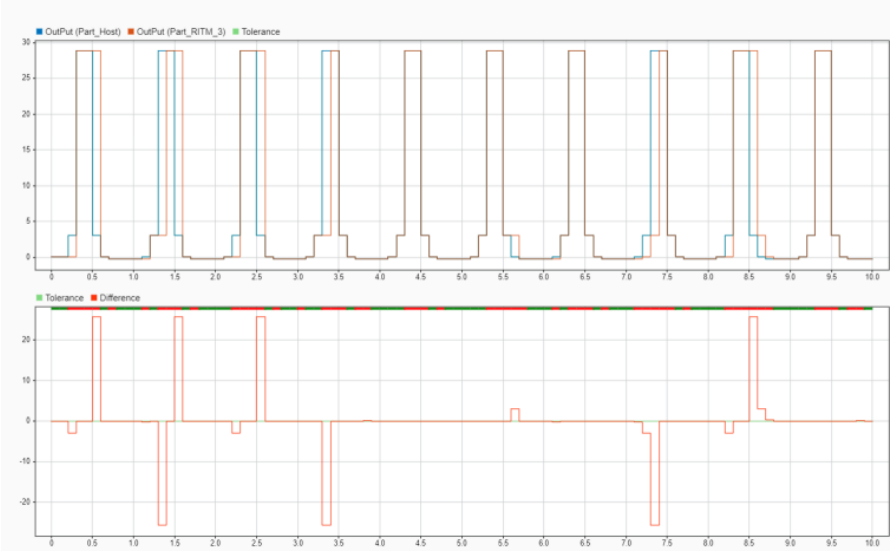


Видно, что расчет блока Sine Wave происходит **после** расчета выхода модели, что и приводит к задержке. Для сравнения рассмотрим порядок расчета блоков для исходной модели:



Блок RTMIDDLE обеспечивает обмен данными между потоками Partition1 и Partition2 и является двойным буфером. Двойной буфер представляет собой массив из двух ячеек и предназначен для обеспечения целостности данных расчетов. Особенно это актуально при передаче массивов между двумя потоками. Принцип работы двойного буфера для модели ritmdemo_block_linuxtask_partition_ref следующий:

Запустим модель в режиме *Monitor And Tune* и сравним результаты симуляции и выполнения модели на РИТМе:



Видно, что результаты различаются. Это объясняется тем, что при многопоточном выполнении модели порядок выполнения частей модели не гарантируется, таким образом **детерминизм передачи** данных между потоками не соблюдается, в то время как **целостность** передачи данных обеспечивается за счет блоков Rate Transition.

Библиотека блоков сериализации шин Simulink

Содержание

- Описание
  - Область применения
- Библиотека блоков
- Блок Сериализация шины
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок Десериализация шины
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока

Описание

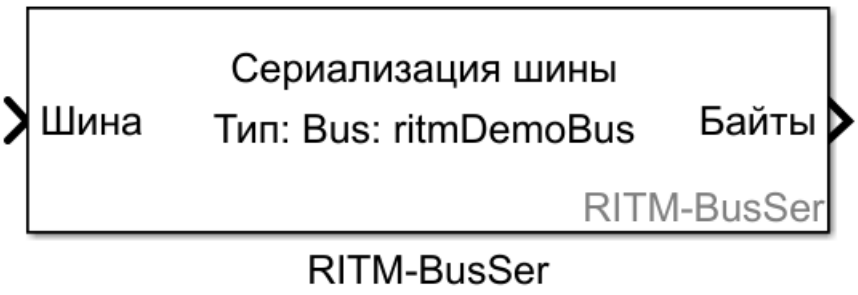
Библиотека блоков для сериализации и десериализации шин Simulink для передачи их по цифровым интерфейсам.

Область применения

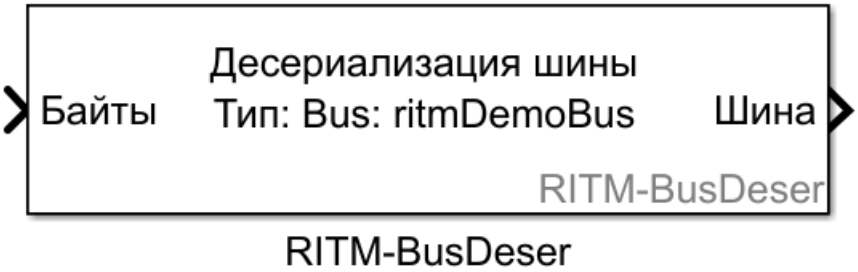
Блоки рекомендуется применять для:

- Для обмена с хост компьютером по UDP
- Передачи данных между несколькими машинами РИТМ

Библиотека блоков



RITM-BusSer

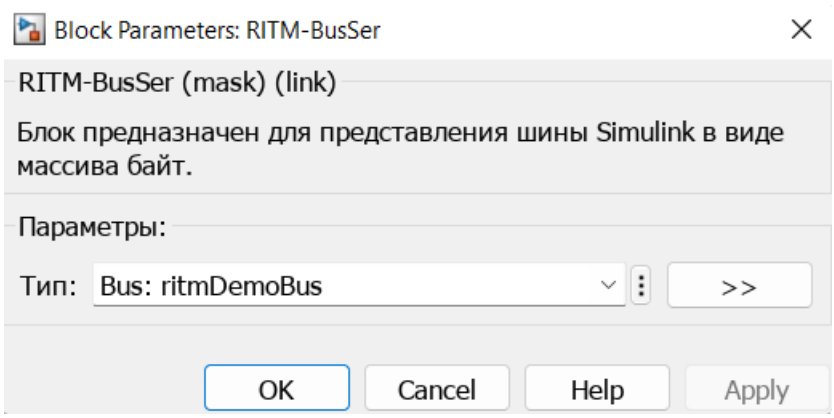


RITM-BusDeser

Блок Сериализация шины

Блок предназначен для представления шины Simulink в виде массива байт.

Маска блока



Входы блока

- **Шина**: Объект Simulink Bus, который будет преобразован в массив байт.

Выходы блока

- **Байты**: Сериализованное представление Simulink Bus в виде массива байт uint8.

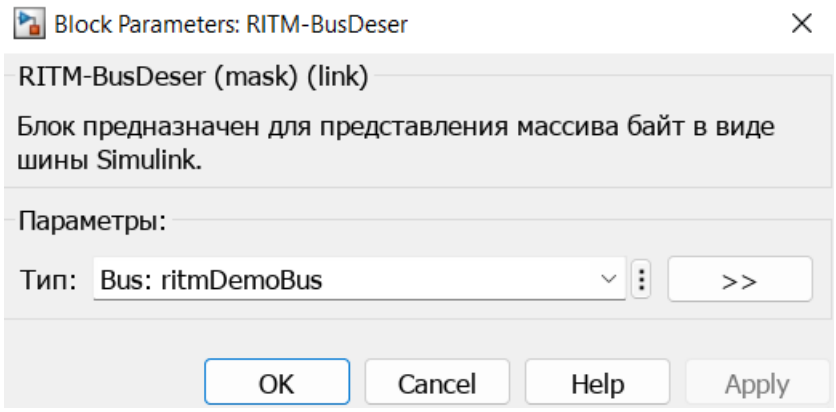
Параметры блока

- **Тип**: Имя Simulink Bus, которое будет использоваться для получения информации о шине для её преобразования.

Блок Десериализация шины

Блок предназначен для представления массива байт в виде шины Simulink.

Маска блока



Входы блока

- **Байты**: Сериализованное представление Simulink Bus, полученное от блока Сериализация шины.

Выходы блока

- **Шина**: Шина Simulink, полученная из массива байт.

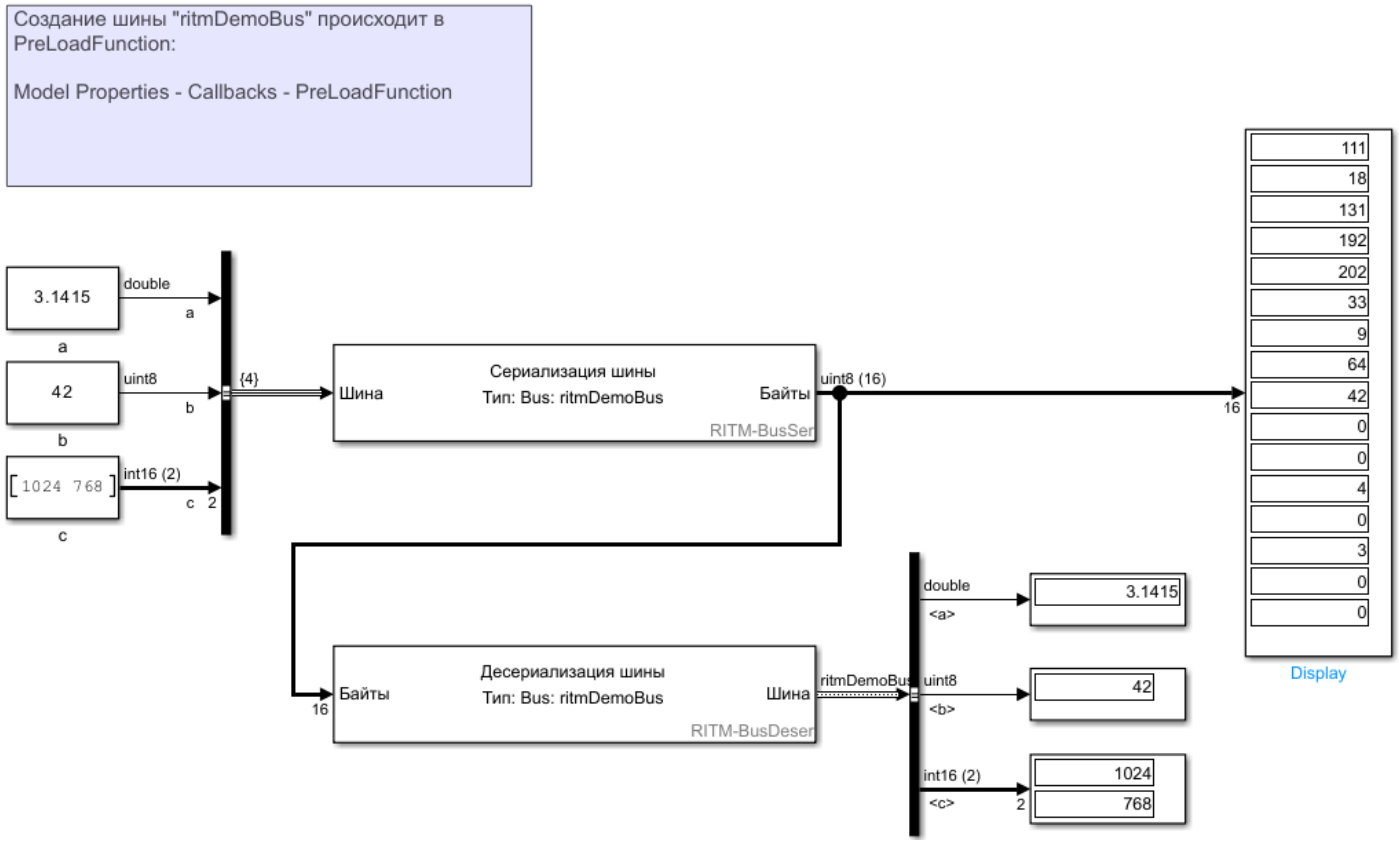
Параметры блока

- **Тип**: Имя Simulink Bus, которое будет использоваться для получения информации о шине её корректной десериализации.

Пример использования блоков сериализации шин Simulink

Пример использования блоков сериализации/десериализации шины Simulink.

```
open_system('ritmdemo_block_busserdes.slx')
```

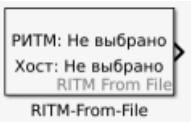


Библиотека блоков RITM From File

Содержание

- Библиотека блоков
- Блок RITM From File
  - Маска блока
  - Выходы блока
  - Параметры блока

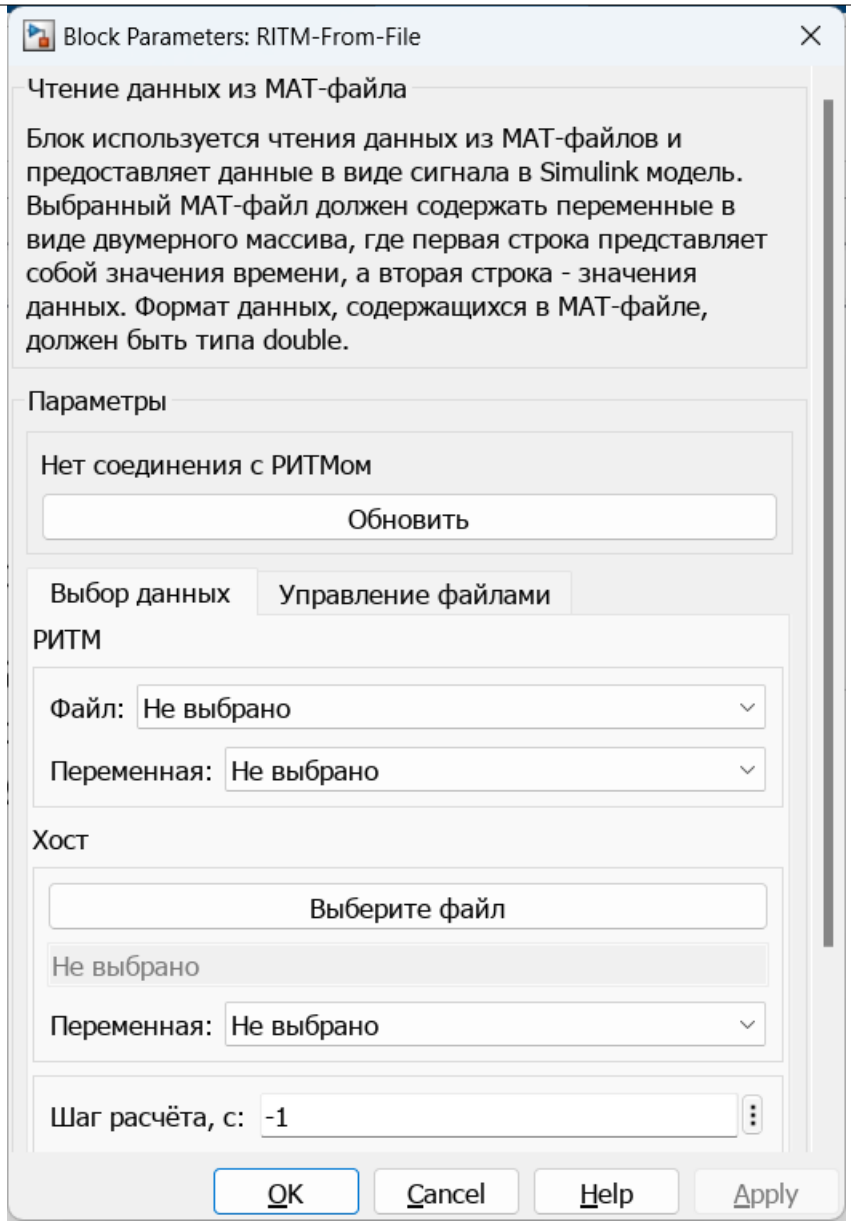
Библиотека блоков

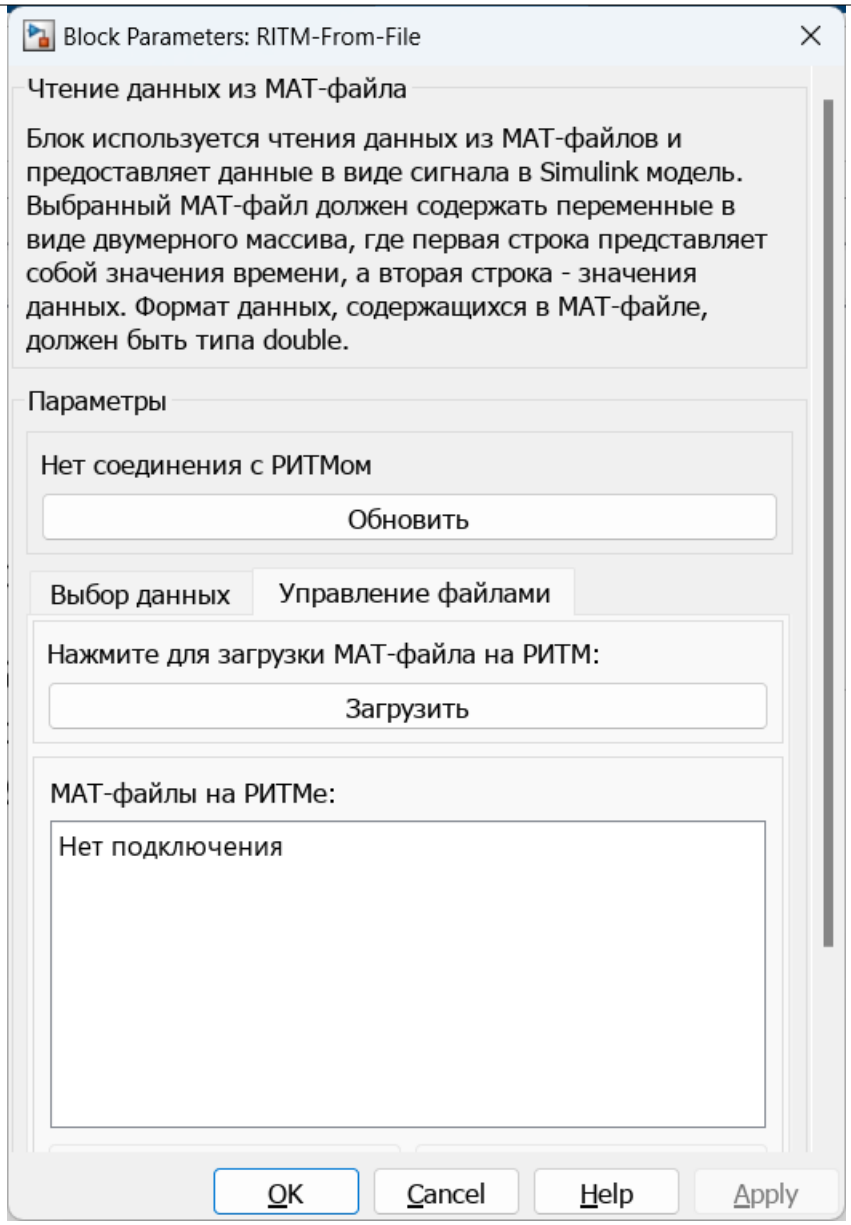


Блок RITM From File

Блок используется чтения данных из MAT-файлов и предоставляет данные в виде сигнала в Simulink модель. Выбранный MAT-файл должен содержать переменные в виде двумерного массива, где первая строка представляет собой значения времени, а вторая строка - значения данных. Формат данных, содержащихся в MAT-файле, должен иметь тип *double*.

Маска блока


Block Parameters: RITM-From-File
Чтение данных из MAT-файла
Блок используется чтения данных из MAT-файлов и предоставляет данные в виде сигнала в Simulink модель. Выбранный MAT-файл должен содержать переменные в виде двумерного массива, где первая строка представляет собой значения времени, а вторая строка - значения данных. Формат данных, содержащихся в MAT-файле, должен быть типа double.
Параметры
Нет соединения с РИТМом
Обновить
Выбор данных
Управление файлами
РИТМ
Файл: Не выбрано
Переменная: Не выбрано
Хост
Выберите файл
Не выбрано
Переменная: Не выбрано
Шаг расчёта, с: -1
OK
Cancel
Help
Apply



Выходы блока

На выходе блока выдаются значения выбранной переменной из MAT-файла в типе *double*.

Параметры блока

- **Соединение с РИТМом**: отображает текущий статус подключения к РИТМу и позволяет обновить состояние подключения.

Маска блока содержит 2 вкладки: ***Выбор данных*** и ***Управление файлами***.

Вкладка ***Выбор данных*** содержит три панели:

- **РИТМ**: Позволяет выбрать MAT-файл (предварительно загруженный на РИТМ через вкладку ***Управление файлами***) и переменную в MAT-файле.
- **Хост**: Позволяет выбрать MAT-файл на компьютере разработчика (хост-компьютере) для работы в обычном режиме симуляции модели Simulink и переменную в MAT-файле.
- **Шаг расчёта, с**: Шаг расчёта в секундах, в соответствии с которым переменная будет считываться из MAT-файла и выдаваться на выход блока.

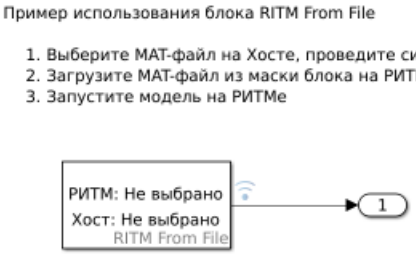
Вкладка ***Управление файлами*** содержит две панели:

- **Загрузить**: Позволяет выбрать MAT-файл и загрузить его на РИТМ.
- **MAT-файлы на РИТМе**: Отображает списов MAT-файлов на РИТМе и позволяет их удалить и выгрузить обратно на хост-компьютер.

Примеры и демонстрации RITM From File

Пример использования блока RITM From File.

```
open_system('ritmdemo_block_fromfile.slx')
```



Модули общего назначения

Содержание

- Модуль RITMeX GP-LC-45
- Модуль RITMeX GP-ID-4X
- Модуль RITMeX GP-AD-24
- Модуль RITMeX GP-ID-24
- Модуль RITMeX GP-ID-34
- Модуль RITMeX GP-EL-64

Модуль RITMeX GP-LC-45

Аналогово-цифровой модуль ввода-вывода RITMeX GP-LC-45, обеспечивающий поддержку:

- Аналоговых входов (32 канала)
- Аналоговых выходов (2 канала)
- Цифровых входов (16 каналов)
- Цифровых выходов (16 каналов)

Модуль RITMeX GP-ID-4X

Аналогово-цифровой модуль ввода-вывода RITMeX GP-ID-4X, обеспечивающий поддержку:

- Аналоговых выходов ток (16 каналов)
- Аналоговых выходов напряжение (16 каналов)
- Цифровых входов (16 каналов)
- Цифровых выходов (16 каналов)

Модуль RITMeX GP-AD-24

Аналоговый модуль ввода-вывода RITMeX GP-AD-24, обеспечивающий поддержку:

- Аналоговых выходов напряжение (32 канала)

Модуль RITMeX GP-ID-24

Цифровой модуль ввода-вывода RITMeX GP-ID-24, обеспечивающий поддержку:

- Цифровых входов/выходов (24 канала)

Модуль RITMeX GP-ID-34

Цифровой модуль ввода-вывода RITMeX GP-ID-34, обеспечивающий поддержку:

- Цифровых входов (32 канала, до 24 вольт)
- Цифровых выходов (32 канала, до 30 вольт)

Модуль RITMeX GP-EL-64

Цифровой модуль ввода-вывода RITMeX GP-EL-64, обеспечивающий поддержку:

- Цифровых входов (16 каналов TTL, 16 каналов с открытым коллектором)
- Цифровых выходов (16 каналов TTL, 16 каналов с открытым коллектором)

Модуль RITMeX GP-LC-45

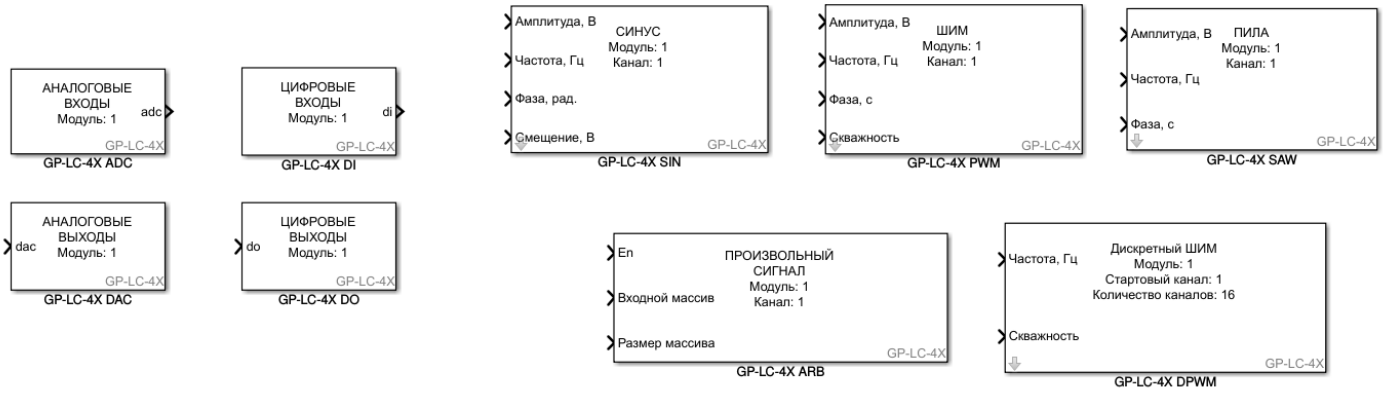
Содержание

- Описание
- Библиотека блоков
- Аналоговые входы
  - Характеристики АЦП
  - Блок АЦП в Simulink
- Аналоговые выходы
  - Характеристики ЦАП
  - Блок ЦАП в Simulink
  - Входы блока
- Цифровые входы
  - Характеристики цифровых входов
  - Блок цифровых входов в Simulink
  - Выходы блока
- Цифровые выходы
  - Характеристики цифровых выходов
  - Блок цифровых выходов в Simulink
  - Входы блока
- Генерация высокочастотных периодических сигналов
  - Блок генерации синусоидального сигнала в Simulink
  - Входы блока
  - Блок генерации ШИМ в Simulink
  - Входы блока
  - Блок генерации пилообразного сигнала в Simulink
  - Входы блока
  - Блок генерации произвольного периодического сигнала в Simulink
  - Входы блока
  - Блок генерации ШИМ на дискретных выходах в Simulink
  - Входы блока

Описание

Аналогово-цифровой модуль ввода-вывода **RITMeX GP-LC-45**, обеспечивающий поддержку аналоговых входов, аналоговых выходов, цифровых входов, цифровых выходов в КПМ “РИТМ”.

Библиотека блоков



Аналоговые входы

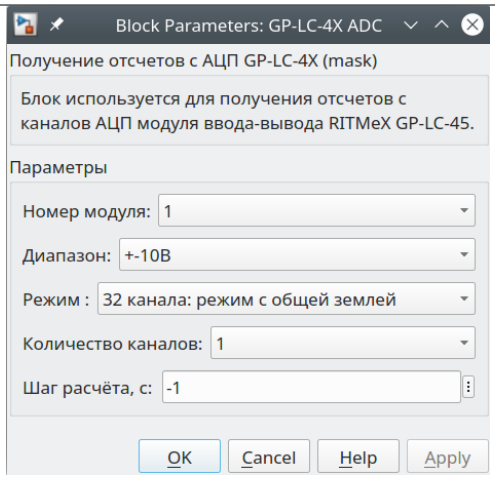
Характеристики АЦП

Количество каналов: 32 в режиме с общей землей и 16 в дифференциальном режиме.
Разрядность: 16 бит.
Диапазоны измерения: ±10 В, ±5 В, ±2 В, ±1 В, ±0,5 В, ±0,2 В.
Частота преобразования: 2 МГц.

Блок АЦП в Simulink

Блок используется для получения отсчетов с каналов АЦП модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Диапазон*** Позволяет выбрать диапазон измерений АЦП. Доступные диапазоны: ±10 В, ±5 В, ±2 В, ±1 В, ±0,5 В, ±0,2 В.
- ***Режим*** Позволяет выбрать режим АЦП (с общей землей или дифференциальный). В режиме с общей землей доступны 32 канала. В дифференциальном режиме доступны 16 каналов.
- ***Количество каналов*** Позволяет выбрать количество используемых каналов АЦП.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для получения отсчетов с АЦП в приложение реального времени.

Выходы блока

На выходе блока формируется массив отсчетов с каналов АЦП типа double. Размер массива соответствует параметру “Количество каналов” в блоке.

Аналоговые выходы

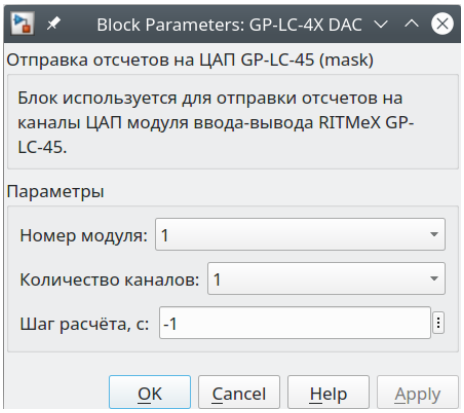
Характеристики ЦАП

Количество каналов: 2.
Выходной диапазон: ±5 В.
Частота вывода: 1 МГц.

Блок ЦАП в Simulink

Блок используется для отправки отсчетов на каналы ЦАП модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество каналов*** Позволяет выбрать количество используемых каналов ЦАП.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на ЦАП из приложения реального времени.

Входы блока

На вход блока поступает массив отсчетов типа double для передачи в каналы ЦАП. Размер массива соответствует параметру “Количество каналов” в блоке.

Цифровые входы

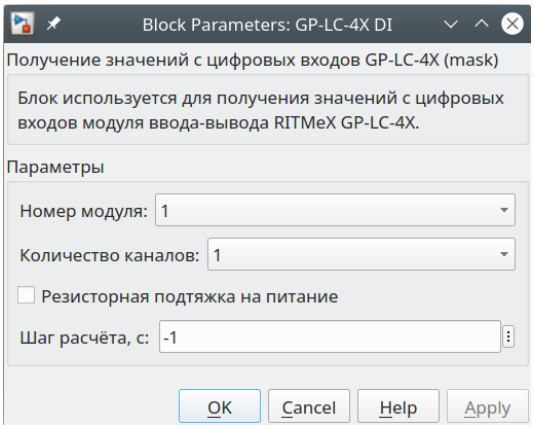
Характеристики цифровых входов

Количество каналов: 16.
Диапазон напряжений: -0,2…+0,6 В (“ноль”), +2,4…+5,0 В (“единица”).
Частота ввода: 2 МГц.

Блок цифровых входов в Simulink

Блок используется для получения значений с цифровых входов модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество каналов*** Позволяет выбрать количество используемых цифровых каналов входа.
- ***Резисторная подтяжка на питание*** Позволяет включить резисторную подтяжку на питание для всех цифровых входов.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Выходы блока

На выходе блока формируется массив отсчетов типа double, принятых с цифровых входов. Размер массива соответствует параметру “Количество каналов” в блоке.

Цифровые выходы

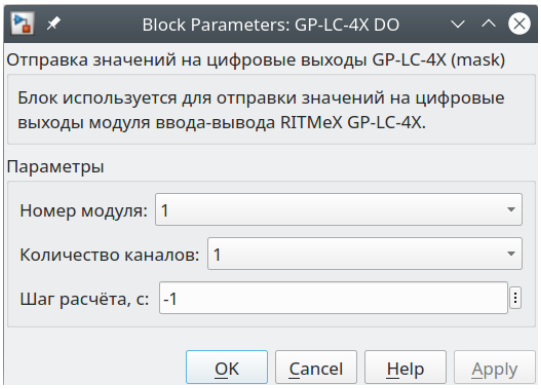
Характеристики цифровых выходов

Количество каналов: 16.
Диапазон напряжений: 0…+0,4 В (“ноль”), +2,4…+3,3 В (“единица”).
Частота вывода: 1 МГц.

Блок цифровых выходов в Simulink

Блок используется для отправки значений на цифровые выходы модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество каналов*** Позволяет выбрать количество используемых цифровых каналов выхода.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.

Входы блока

На вход блока поступает массив отсчетов типа double для передачи на цифровые выходы. Размер массива соответствует параметру “Количество каналов” в блоке.

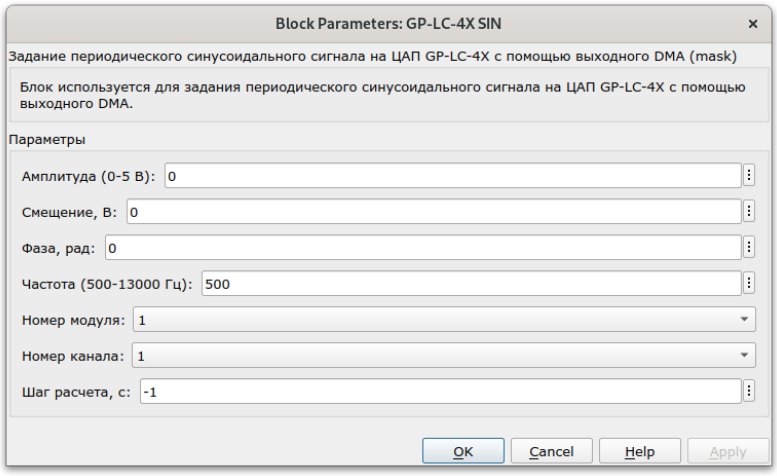
Генерация высокочастотных периодических сигналов

Блок генерации синусоидального сигнала в Simulink

Блок используется для генерации высокочастотного синусоидального сигнала на аналоговых выходах модуля ввода-вывода RITMeX GP-LC-

45.

Маска блока



Параметры блока

- ***Амплитуда*** Используется для задания начальной амплитуды в вольтах при инициализации сигнала. Данный параметр имеет тип double и диапазон значений 0-5В.
- ***Смещение*** Используется для задания начального смещения синусоидального сигнала по оси значений. Сумма амплитуды и смещения не должна превышать 5В или быть меньше -5В. Данный параметр имеет тип double.
- ***Фаза*** Используется для задания начальной фазы сигнала в радианах. Данный параметр имеет тип double.
- ***Частота*** Используется для задания начальной частоты сигнала в Гц. Данный параметр имеет тип double и диапазон значений 500-13000Гц.
- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала*** Используется для задания номера канала аналогового вывода, на котором будет сгенерирован данный сигнал.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.
- ***Важно*** Все вышеперечисленные параметры кроме последних трех задают форму сигнала только при инициализации. Если на первом расчетном шаге соответствующие параметры на входах блока будут отличаться, то форма сигнала будет изменена.

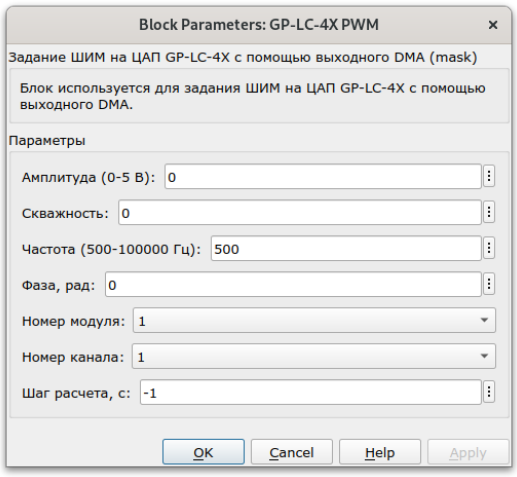
Входы блока

- ***Амплитуда*** Используется для задания амплитуды сигнала в вольтах. Данный вход имеет тип double и диапазон значений 0-5В.
- ***Частота*** Используется для задания частоты сигнала в Гц. Данный вход имеет тип double и диапазон значений 500-13000Гц.
- ***Фаза*** Используется для задания начальной фазы сигнала в радианах. Данный параметр имеет тип double.
- ***Смещение*** Используется для задания смещения синусоидального сигнала по оси значений. Сумма амплитуды и смещения не должна превышать 5В или быть меньше -5В. Данный вход имеет тип double.

Блок генерации ШИМ в Simulink

Блок используется для генерации высокочастотного ШИМ на аналоговых выходах модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Амплитуда*** Используется для задания начальной амплитуды в вольтах при инициализации сигнала. Данный параметр имеет тип double и диапазон значений 0-5В.
- ***Скважность*** Используется для задания начальной скважности ШИМ. Данный параметр имеет тип double. Под скважностью понимается отношение периода сигнала к длительности импульса.

- ***Частота*** Используется для задания начальной частоты сигнала в Гц. Данный параметр имеет тип double и диапазон значений 500-100000Гц.
- ***Фаза*** Используется для задания начальной фазы сигнала в секундах. Данный параметр имеет тип double.
- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала*** Используется для задания номера канала аналогового вывода, на котором будет сгенерирован данный сигнал.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.
- ***Важно*** Все вышеперечисленные параметры, кроме последних трех, задают форму сигнала только при инициализации. Если на первом расчетном шаге соответствующие параметры на входах блока будут отличаться, то форма сигнала будет изменена.

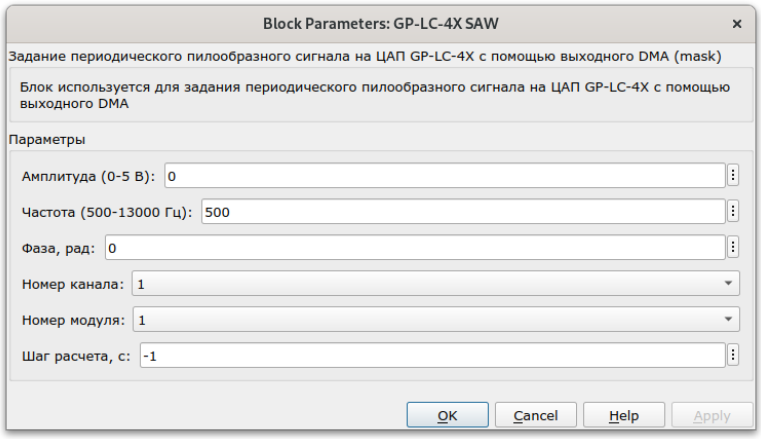
Входы блока

- ***Амплитуда*** Используется для задания амплитуды сигнала в вольтах. Данный вход имеет тип double и диапазон значений 0-5В.
- ***Частота*** Используется для задания частоты сигнала в Гц. Данный вход имеет тип double и диапазон значений 500-100000Гц.
- ***Фаза*** Используется для задания фазы сигнала в секугдах. Данный вход имеет тип double.
- ***Скважность*** Используется для задания скважности ШИМ. Данный вход имеет тип double. Под скважностью понимается отношение периода сигнала к длительности импульса.

Блок генерации пилообразного сигнала в Simulink

Блок используется для генерации высокочастотного пилообразного сигнала на аналоговых выходах модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Амплитуда*** Используется для задания начальной амплитуды в вольтах при инициализации сигнала. Данный параметр имеет тип double и диапазон значений 0-5В.а
- ***Частота*** Используется для задания начальной частоты сигнала в Гц. Данный параметр имеет тип double и диапазон значений 500-13000Гц.
- ***Фаза*** Используется для задания начальной фазы сигнала в секундах. Данный параметр имеет тип double.
- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала*** Используется для задания номера канала аналогового вывода, на котором будет сгенерирован данный сигнал.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.
- ***Важно*** Все вышеперечисленные параметры, кроме последних трех, задают форму сигнала только при инициализации. Если на первом расчетном шаге соответствующие параметры на входах блока будут отличаться, то форма сигнала будет изменена.

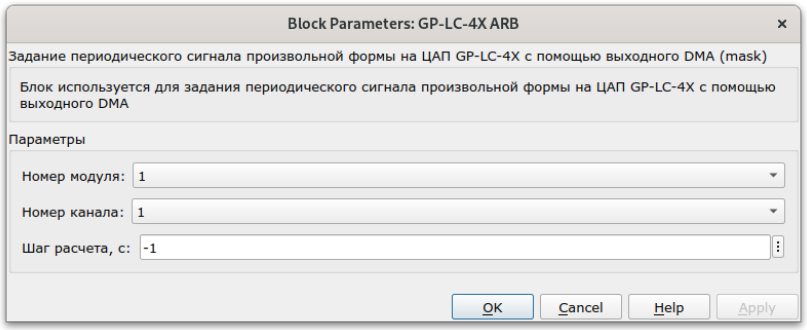
Входы блока

- ***Амплитуда*** Используется для задания амплитуды сигнала в вольтах. Данный вход имеет тип double и диапазон значений 0-5В.
- ***Частота*** Используется для задания частоты сигнала в Гц. Данный вход имеет тип double и диапазон значений 500-13000Гц.
- ***Фаза*** Используется для задания фазы сигнала в секугдах. Данный вход имеет тип double.

Блок генерации произвольного периодического сигнала в Simulink

Блок используется для генерации произвольного периодического сигнала на аналоговых выходах модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала*** Используется для задания номера канала аналогового вывода, на котором будет сгенерирован данный сигнал.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.
- ***Важно*** Все вышеперечисленные параметры, кроме последних трех, задают форму сигнала только при инициализации. Если на первом расчетном шаге соответствующие параметры на входах блока будут отличаться, то форма сигнала будет изменена.

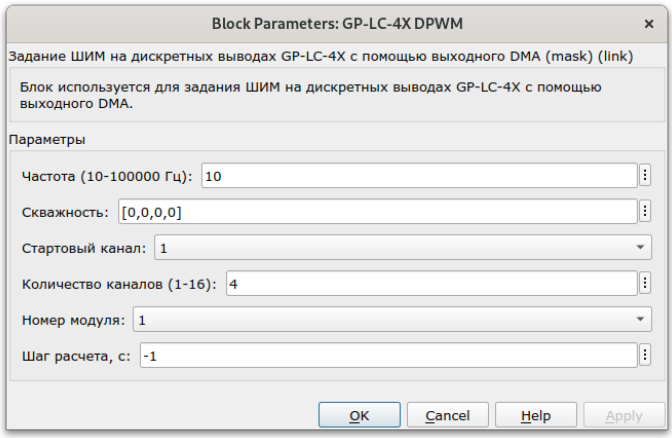
Входы блока

- ***En*** Флаг разрешения перезаписи массива с формой сигнала. Переформирование сигнала будет происходить только тогда, когда на данном входе установлена единица. Данный вход имеет тип uint8.
- ***Входной массив*** Массив элементов типа double, в котором содержится форма одного периода сигнала, необходимого для воспроизведения. Частота сэмплирования ЦАП 1МГц, соответственно для воспроизведения сигнала частотой 10кГц необходимо подготовить массив из 100 элементов, сигнала частотой 1кГц необходимо подготовить массив из 1000 элементов и т.д. Максимальный размер данного массива один миллион элементов, что соответствует сигналу частотой 1Гц.
- ***Размер массива*** Размер вышеуказанного массива в элементах.

Блок генерации ШИМ на дискретных выходах в Simulink

Блок используется для генерации ШИМ на дискретных выходах модуля ввода-вывода RITMeX GP-LC-45.

Маска блока



Параметры блока

- ***Частота*** Используется для задания начальной частоты сигнала в Гц для всех задействованных каналов дискретного вывода. Данный параметр имеет тип double и диапазон значений 10-100000Гц.
- ***Скважность*** Используется для задания начальной скважности ШИМ. Данный параметр является массивом типа double, который должен иметь размер равный количеству задействованных каналов. Под скважностью понимается отношение периода сигнала к длительности импульса.
- ***Стартовый канал*** Используется для задания первого используемого канала дискретного вывода.
- ***Количество каналов*** Используется для задания количества задействованных каналов дискретного вывода.
- ***Номер модуля*** Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Шаг расчёта*** Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени. ***Важно!*** Все вышеперечисленные параметры кроме последних четырех задают форму сигнала только при инициализации. Если на первом расчетном шаге соответствующие параметры на входах блока будут отличаться, то форма сигнала будет изменена.

Входы блока

- ***Частота*** Используется для задания частоты сигнала в Гц для всех задействованных каналов дискретного вывода. Данный вход имеет тип double и диапазон значений 10-100000Гц.
- ***Скважность*** Используется для задания скважности ШИМ. Данный вход принимает массив типа double, который должен иметь размер равный количеству задействованных каналов. Под скважностью понимается отношение периода сигнала к длительности импульса.

Важно! При использовании данного блока теряется возможность использовать блок GP-LC-4X DO на данном модуле ввода-вывода даже на незадействованных каналах.

Предупреждение! При использовании блоков генерации высокочастотных периодических сигналов на конкретном канале аналогового вывода модуля ввода-вывода RITMeX GP-LC-45 уже не будет возможности использовать другие блоки генерации высокочастотных периодических сигналов на другом канале данного модуля.

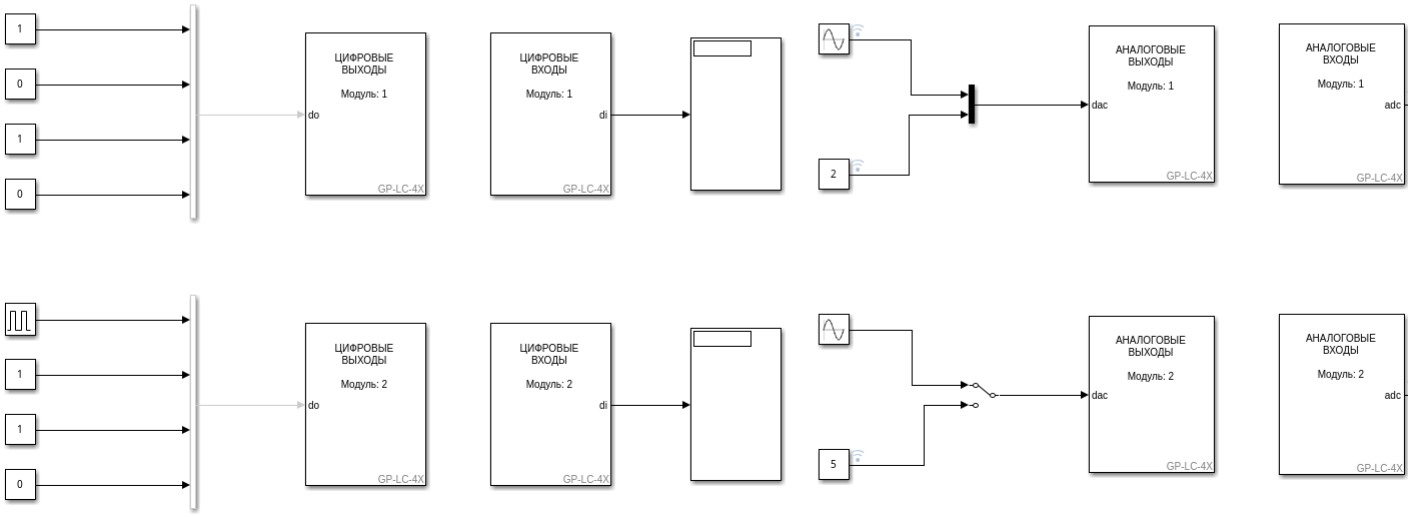
Предупреждение! При изменении параметров сигнала на низких частотах (до 1кГц) изменение формы сигнала вступает в силу через 1-2 периода данного сигнала. При изменении параметров сигнала на высоких частотах изменения вступают в силу не менее чем через 1 мс и не более чем через 2 мс. Процесс изменения формы сигнала не сказывается на ТЕТ блока.

Примеры и демонстрации GP-LC-45

Пример использования входов и выходов модуля GP-LC-45.

```
open_system('ritmdemo_block_gplc4x.slx')
```

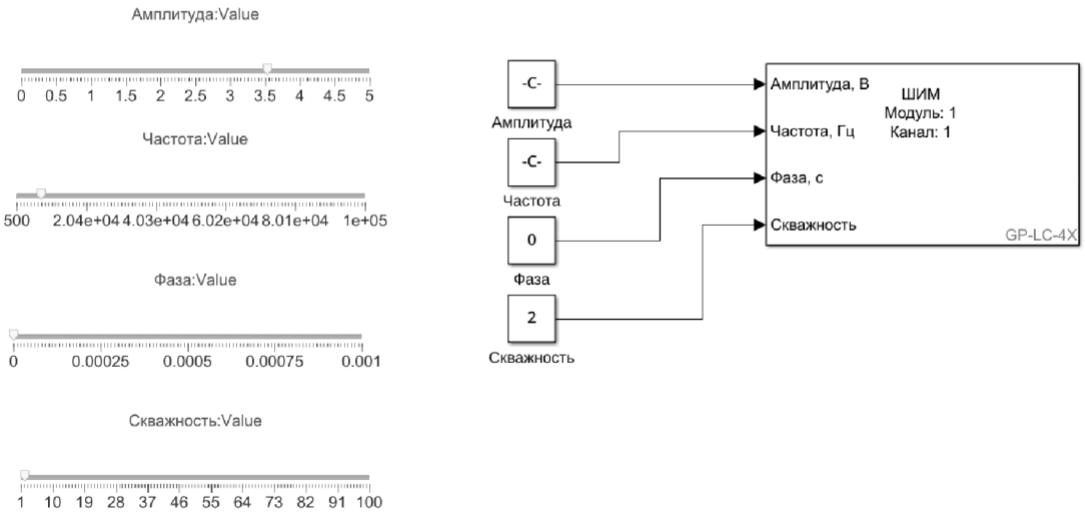
```
Для воспроизведения работы модели, необходимо:
Цифровые входы/выходы:
1. Аппаратно подключить цифровые входы с выходами:
    DI1(18) - DO1(10)
    DI2(37) - DO2(29)
    DI3(17) - DO3(9)
    DI4(36) - DO4(28)
2. Установить нужные значения, подаваемые на выход
3. Запустить симуляцию
4. Посмотреть значения цифровых входов в блоке Display или в Data Inspector

Аналоговые входы/выходы:
1. Аппаратно подключить аналоговые входы с выходами:
    X1(16) - DAC1(18)
    X2(15) - DAC2(19)
    X3(14) - DAC1(18)
    X4(13) - DAC2(19)
    AGND(17) - GND32(36)
2. Подключить к аналоговым выходам нужные значения
3. Выбрать количество каналов АЦП и ЦАП
4. Запустить симуляцию
5. Посмотреть значения аналоговых входов в Data Inspector
```



Пример использования блока генерации ШИМ GP-LC-45.

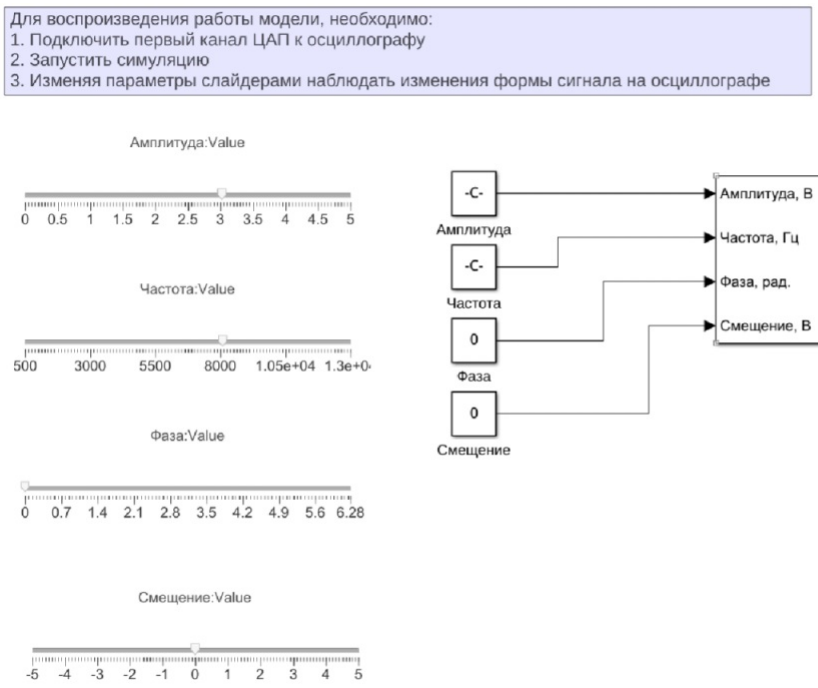
```
open_system('ritmdemo_block_gplc4x_pwm.slx')
```

```
Для воспроизведения работы модели, необходимо:
1. Подключить первый канал ЦАП к осциллографу
2. Запустить симуляцию
3. Изменяя параметры слайдерами наблюдать изменения формы сигнала на осциллографе
```



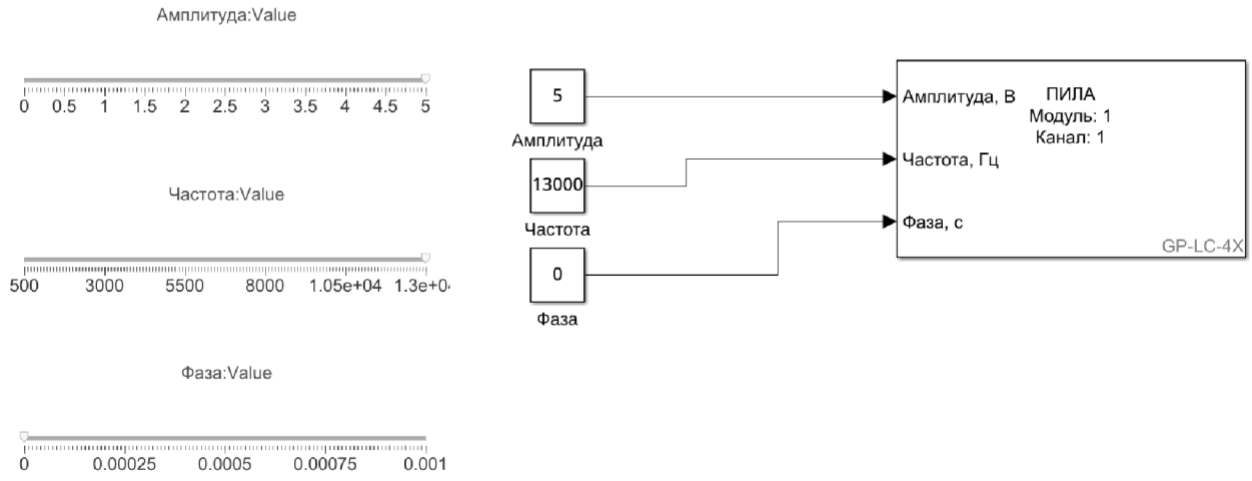
Пример использования блока генерации синусоидального сигнала GP-LC-45.

```
open_system('ritmdemo_block_gplc4x_sin.slx')
```



Пример использования блока генерации пилообразного сигнала GP-LC-45.

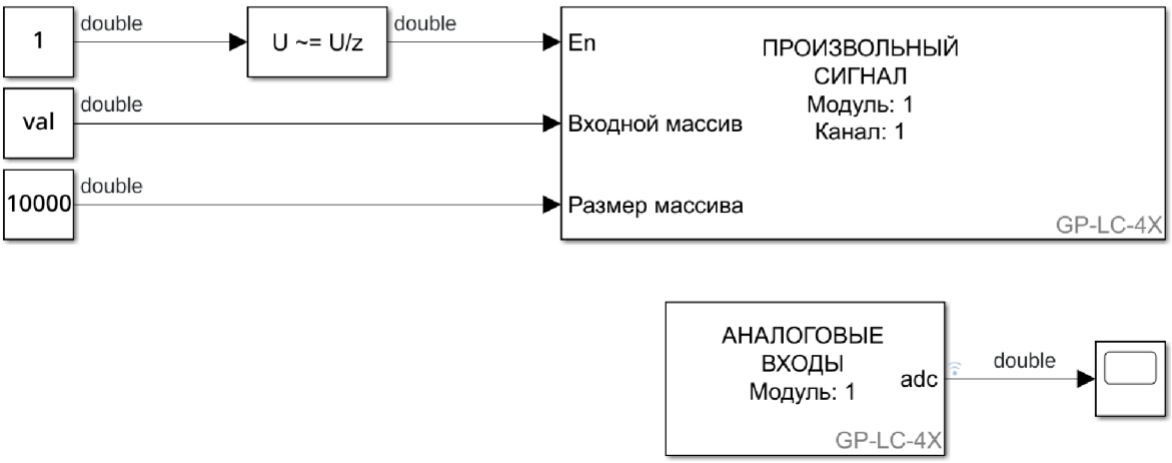
```
open_system('ritmdemo_block_gplc4x_saw.slx')
```



Пример использования блока генерации периодического сигнала произвольной формы GP-LC-45.

```
open_system('ritmdemo_block_gplc4x_arb.slx')
```

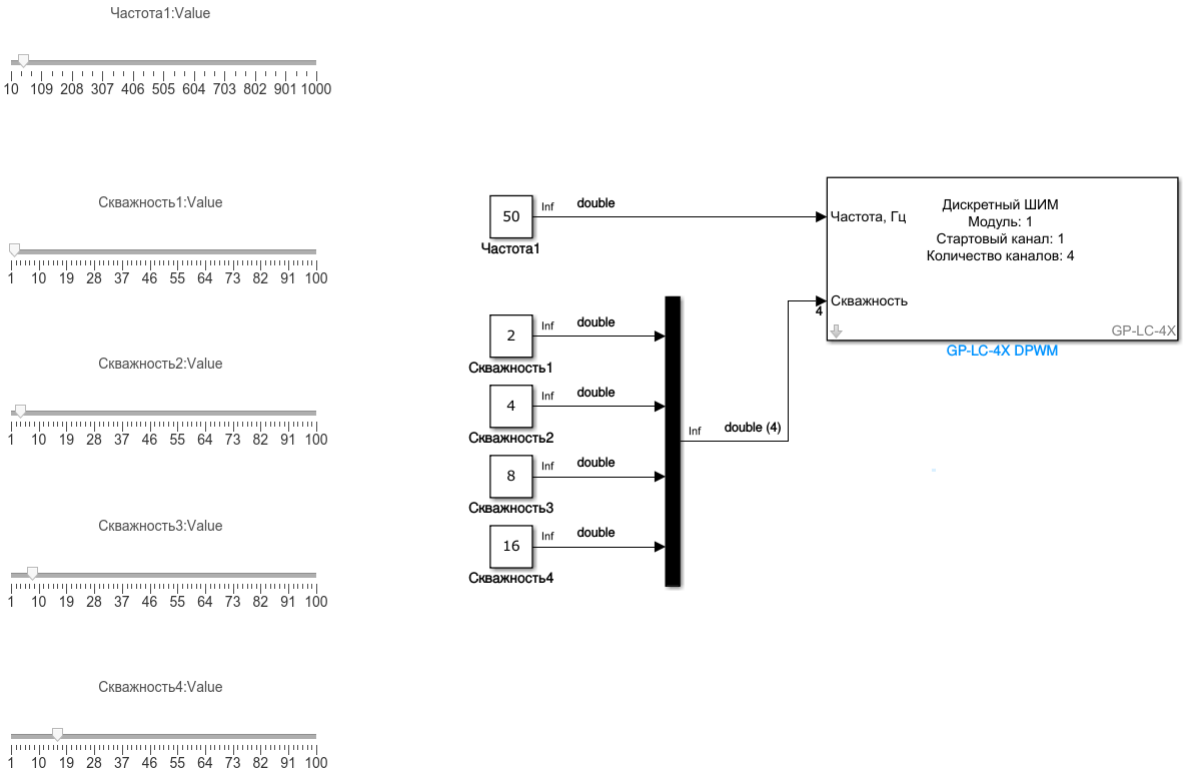
Для воспроизведения работы модели, необходимо:
1. Подключить первый канал ЦАП к 1 каналу АЦП.
2. Запустить симуляцию.
3. Открыть Scope.



Пример использования блока генерации ШИМ на дискретных выходах GP-LC-45.

```
open_system('ritmdemo_block_gplc4x_dpwm.slx')
```

Для воспроизведения работы модели, необходимо:
1. Подключить DO1-DO4 к осциллографу.
2. Запустить симуляцию.
3. Изменять параметры слайдерами и смотреть на изменение формы сигнала на осциллографе.



Назначение пинов GP-LC-45

Содержание

- Аналоговые входы и выходы
- Цифровые входы и выходы

Аналоговые входы и выходы

Назначение контактов на поставляемом клеммном модуле:

Контакт	Сигнал	Контакт	Сигнал
1	X16	20	Y16
2	X15	21	Y15
3	X14	22	Y14
4	X13	23	Y13
5	X12	24	Y12
6	X11	25	Y11
7	X10	26	Y10
8	X9	27	Y9
9	X8	28	Y8
10	X7	29	Y7
11	X6	30	Y6
12	X5	31	Y5
13	X4	32	Y4
14	X3	33	Y3
15	X2	34	Y2
16	X1	35	Y1
17	AGND	36	GND32
18	DAC1	37	NC
19	DAC2		

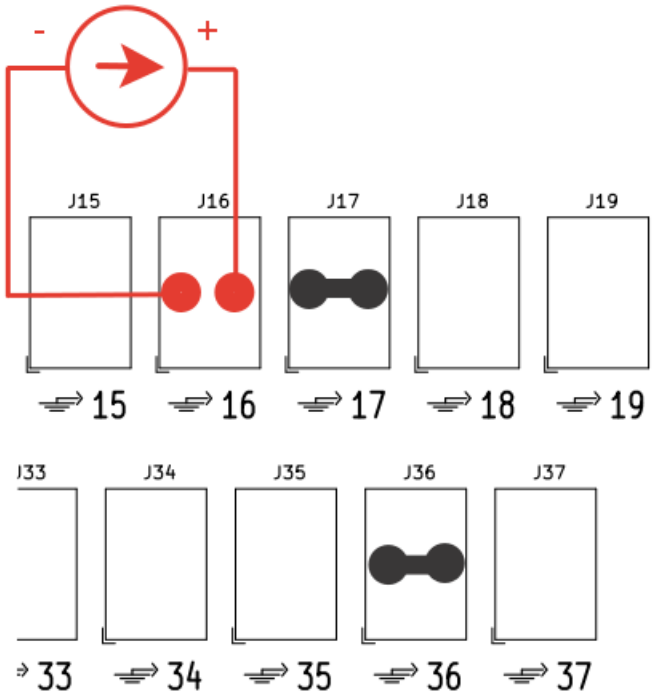
Обозначения:
X1..X16: Неинвертирующий вход напряжения каналов 1...16 для дифференциального режима. Каналы 1..16 для режима “с общей землей”.
Y1..Y16: Инвертирующий вход напряжения каналов 1..16 для дифференциального режима. Каналы 17..32 для режима “с общей землей”.
DAC1, DAC2: Выходы 1 и 2 канала ЦАП.
AGND: Аналоговая земля.
GND32: В режиме “с общей землей”: общий инвертирующий вход каналов 1..32. Для

всех режимов должен быть подключен к AGND (в дифференциальном режиме – для увеличения помехозащищенности).
NC: Не подключено.

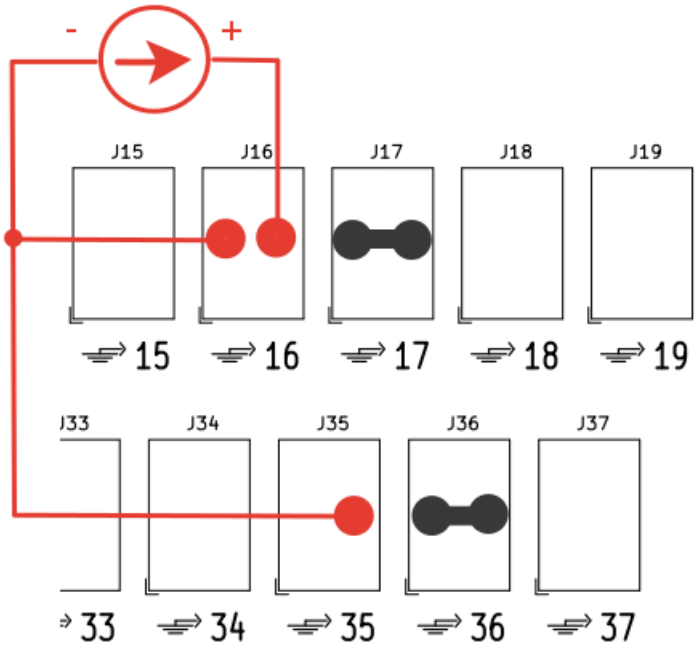
В случае комплектации клеммным модулем **GP-RT-Terminal-37** для корректной работы АЦП необходимо установить два провода-перемычки между контактами **17**, **36** и **землей** клеммного модуля. Пример соединения приведен на рисунке.

Ниже приведены основные варианты подключения внешних источников сигналов на примере канала 1.

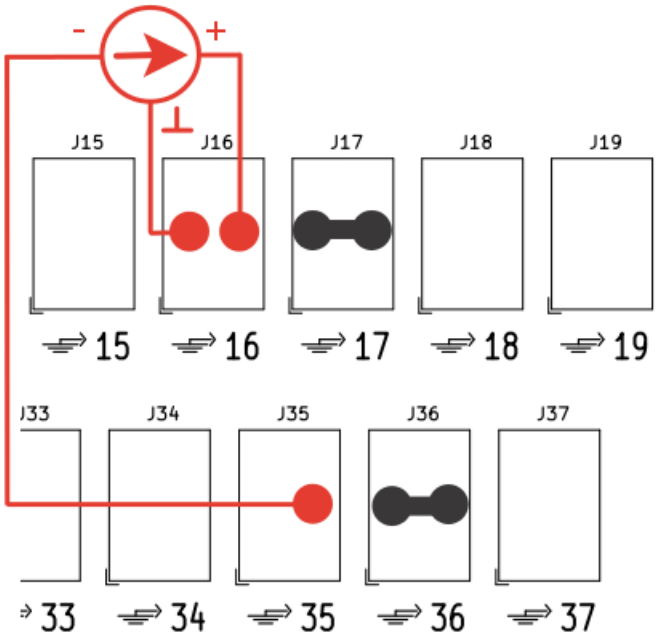
Подключение однофазного источника в режиме “с общей землей”



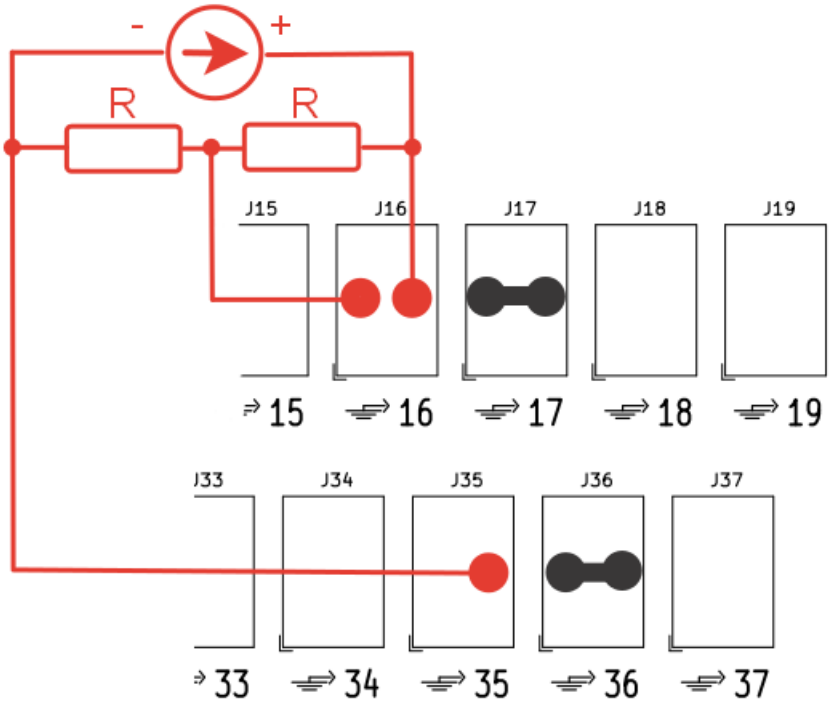
Подключение однофазного источника в дифференциальном режиме



Подключение дифференциального источника при наличии вывода земли



Подключение дифференциального источника при отсутствии вывода земли



Цифровые входы и выходы

Назначение контактов на поставляемом клеммном модуле:

Контакт	Сигнал	Контакт	Сигнал
1	DGND	20	DGND
2	+3.3V	21	+3.3V
3	DO15	22	DO16
4	DO13	23	DO14
5	DO11	24	DO12
6	DO9	25	DO10
7	DO7	26	DO8
8	DO5	27	DO6
9	DO3	28	DO4
10	DO1	29	DO2
11	DI15	30	DI16
12	DI13	31	DI14
13	DI11	32	DI12
14	DI9	33	DI10
15	DI7	34	DI8
16	DI5	35	DI6

Контакт	Сигнал	Контакт	Сигнал
17	DI3	36	DI4
18	DI1	37	DI2
19	NC		

Обозначения:
DI1..DI16: Цифровые входы с 1 до 16.
DO1..DO16: Цифровые выходы с 1 до 16.
DGND: Цепь “общего провода” для цифровых входов и выходов.
NC: Не используется.

Модули RITMeX GP-ID-45/GP-ID-46

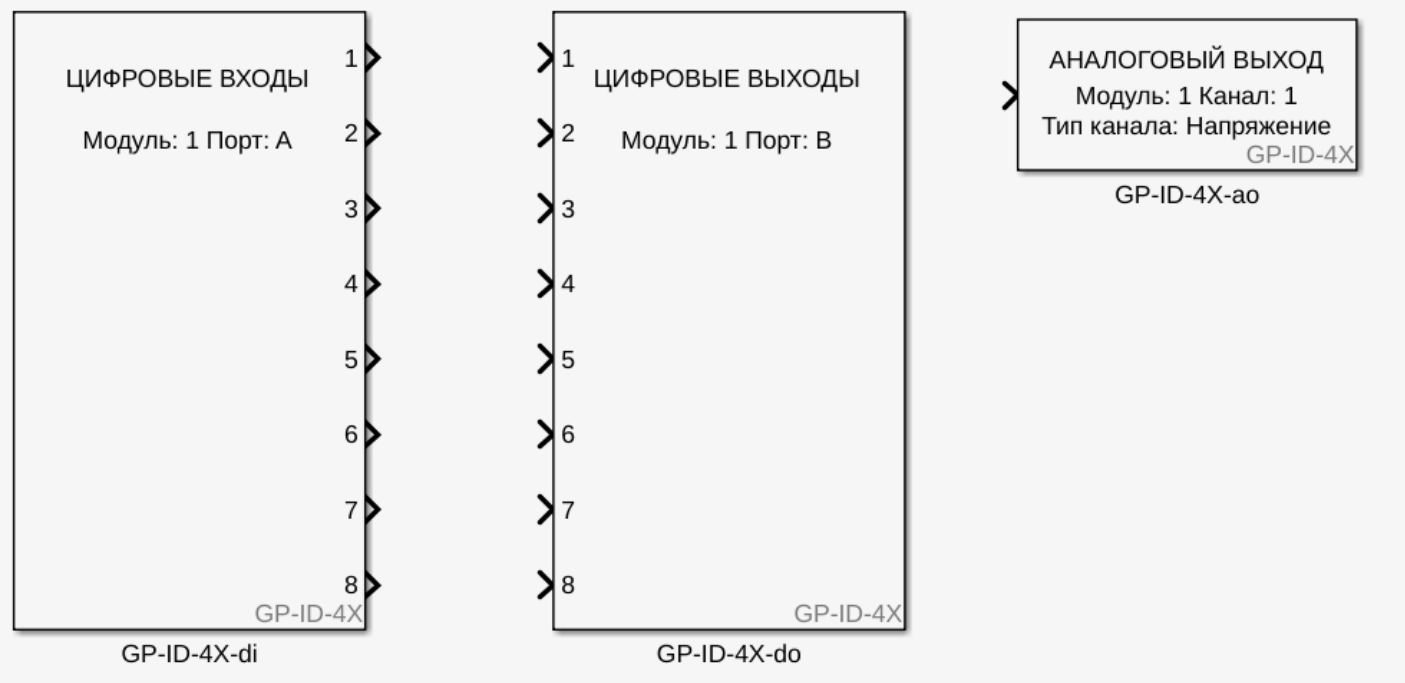
Содержание

- Описание
- Библиотека блоков
- Аналоговые выходы
  - Характеристики ЦАП
  - Блок “Аналоговый выход” в Simulink
- Цифровые входы
  - Характеристики цифровых входов
  - Блок “Цифровые входы” в Simulink
- Цифровые выходы
  - Характеристики цифровых выходов
  - Блок “Цифровые выходы” в Simulink

Описание

Аналогово-цифровые модули ввода-вывода RITMeX GP-ID-45 и RITMeX GP-ID-46, обеспечивающие поддержку аналоговых выходов (напряжение или ток), цифровых входов, цифровых выходов в КПМ “РИТМ”.

Библиотека блоков



Аналоговые выходы

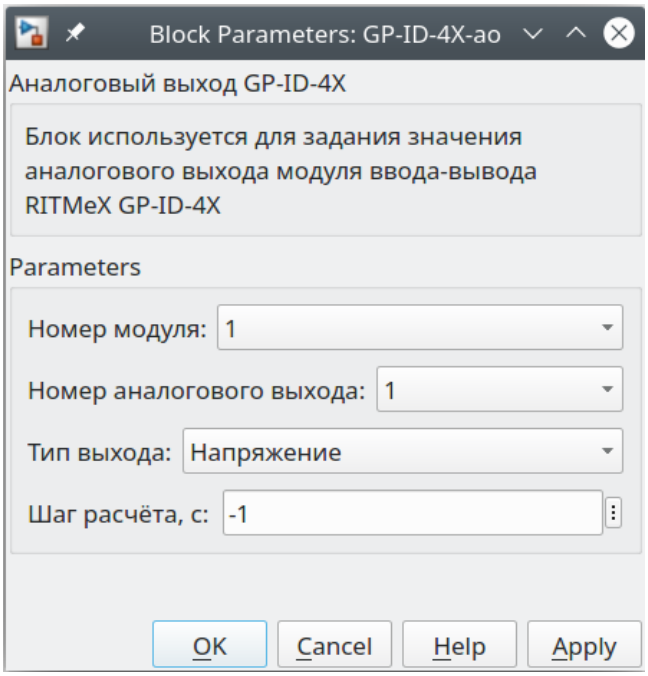
Характеристики ЦАП

Количество каналов: 16.
Разрядность: 14 бит.
Выходной диапазон: ±10 В или 0-20 мА.
Скорость нарастания: 0.71 В/мкс.

Блок “Аналоговый выход” в Simulink

Блок используется для задания значения аналогового выхода модуля ввода-вывода RITMeX GP-ID-45 и RITMeX GP-ID-46.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Номер аналогового выхода***: Номер канала аналогового выхода.
- ***Тип выхода***: *Ток* или *Напряжение*.
  Важно: Изменение режима *Ток/Напряжение* требует физического переключения контактов в соответствии со схемой в разделе Назначение пинов GP-ID-4X.
- ***Шаг расчёта***: Позволяет задать шаг расчета в секундах для отправки отсчетов на ЦАП из приложения реального времени.

Вход блока

На вход блока подается значение типа double, которое, в зависимости от параметра ***Тип выхода***, интерпретируются как значение тока в миллиамперах или значение напряжение в вольтах.

Цифровые входы

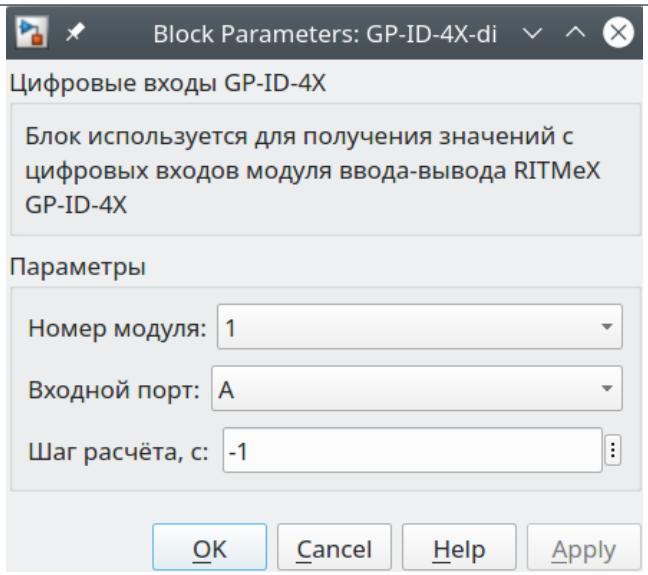
Характеристики цифровых входов

Количество каналов: 16 каналов сгруппированных в 2 порта: **A** и **B**.
Диапазон напряжений: 0…+0,8 В (“ноль”), +2,0…+5,0 В (“единица”).
Частота ввода: 1 МГц (GP-ID-45), 200кГц (GP-ID-46).

Блок “Цифровые входы” в Simulink

Блок используется для получения значений с цифровых входов модуля ввода-вывода RITMeX GP-ID-45 и RITMeX GP-ID-46.

Маска блока



Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Входной порт**: Обозначение порта, со входов которого блок будет получать данные.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Цифровые выходы

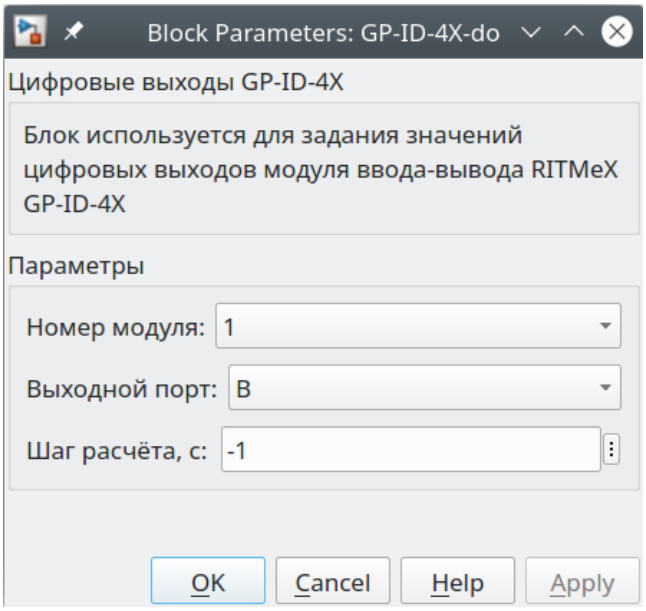
Характеристики цифровых выходов

Количество каналов: 16 каналов, сгруппированных в 2 порта: **C** и **D**.
Диапазон напряжений: 0...+0,8 В (“ноль”), +2,0...+5,0 В (“единица”).
Частота ввода: 1 МГц (GP-ID-45), 200кГц (GP-ID-46).

Блок “Цифровые выходы” в Simulink

Блок используется для задания значений цифровых выходов модуля ввода-вывода RITMeX GP-ID-45 и RITMeX GP-ID-46.

Маска блока



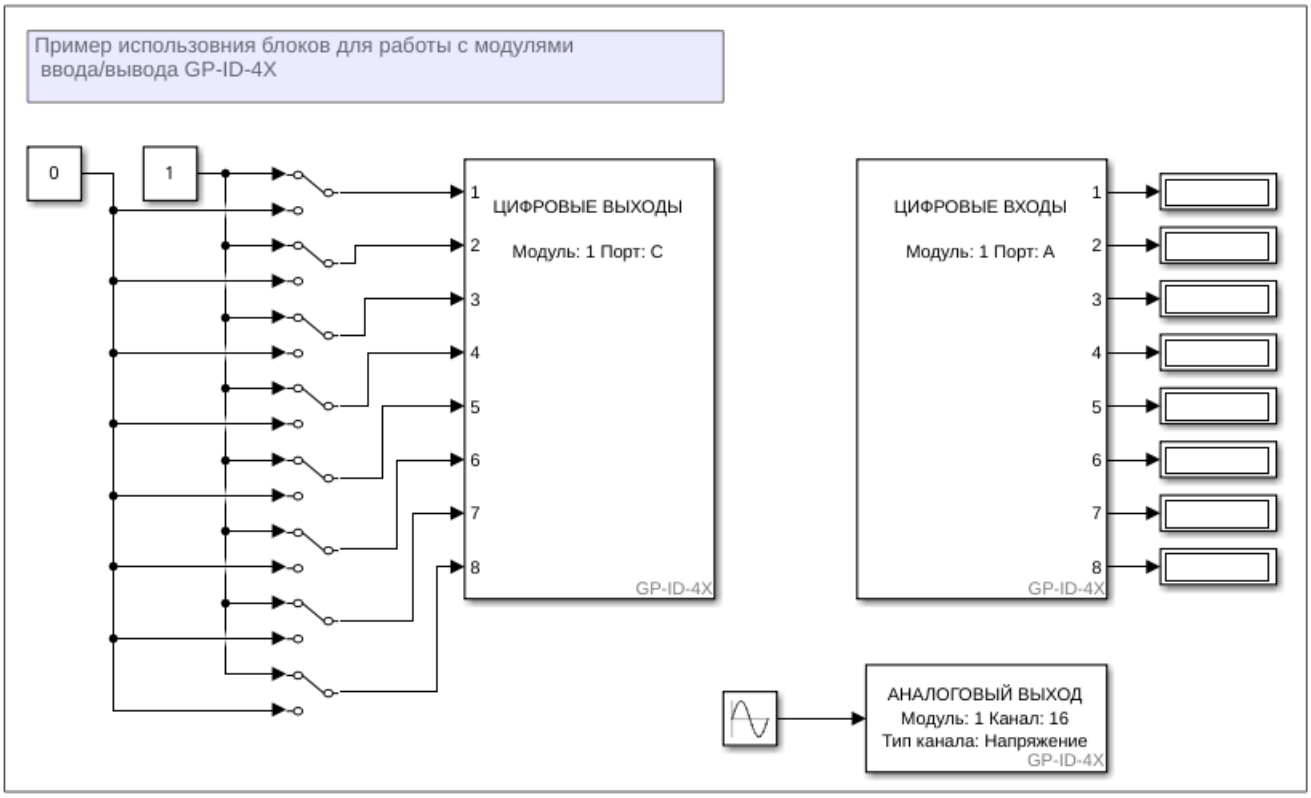
Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Выходной порт**: Обозначение порта, на выходы которого блок будет подавать значения.
- **Шаг расчета**: Позволяет задать шаг расчета в секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.

Примеры и демонстрации GP-ID-4X

Пример использования блоков GP-ID-4X.

```
open_system('ritmdemo_block_gpid4x.slx')
```



Назначение пинов GP-ID-4X

Содержание

- Аналоговые выходы
  - Схема подключения аналогового выхода как источника тока
  - Схема подключения аналогового выхода как источника напряжения
- Цифровые выходы
- Цифровые входы

Аналоговые выходы

Назначение контактов на поставляемом клеммном модуле:

Контакт	Сигнал	Контакт	Сигнал
1	VO1	20	IO1
2	VO2	21	IO2
3	VO3	22	IO3
4	VO4	23	IO4
5	AGND	24	N/A
6	VO5	25	IO5
7	VO6	26	IO6
8	VO7	27	IO7
9	VO8	28	IO8
10	AGND	29	N/A
11	VO9	30	IO9
12	VO10	31	IO10
13	VO11	32	IO11
14	VO12	33	IO12
15	AGND	34	IO13
16	VO13	35	IO14
17	VO14	36	IO15
18	VO15	37	IO16
19	VO16		

Обозначения:
VO1..VO16: Аналоговые выводы заданного напряжения.
IO1..IO16: Аналоговые выводы заданного тока.
AGND: Общая земля для аналоговых выходов.
N/A: Неподключенные контакты.

Схема подключения аналогового выхода как источника тока

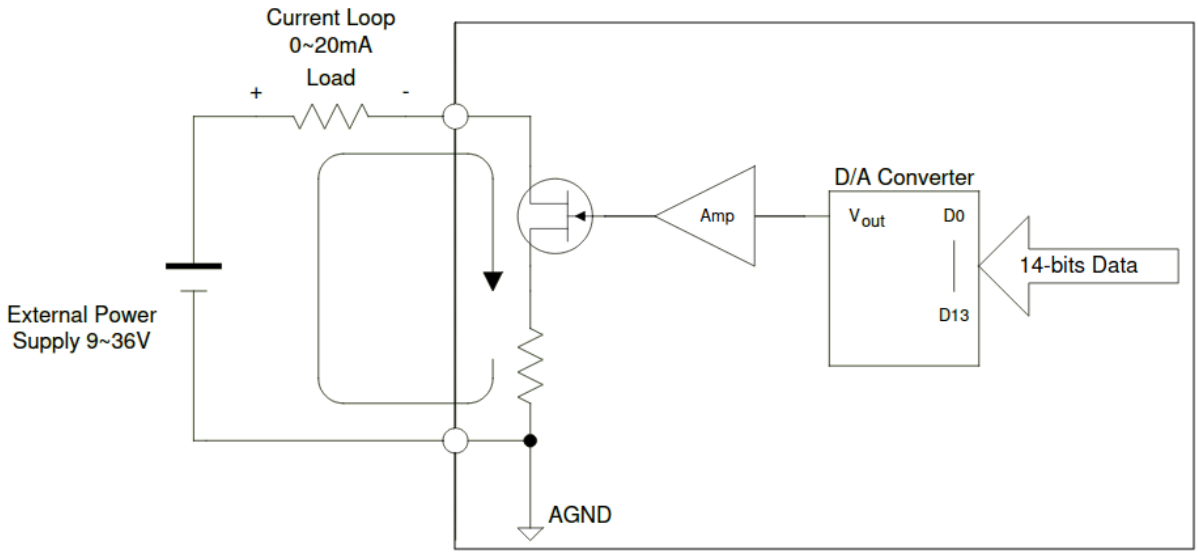
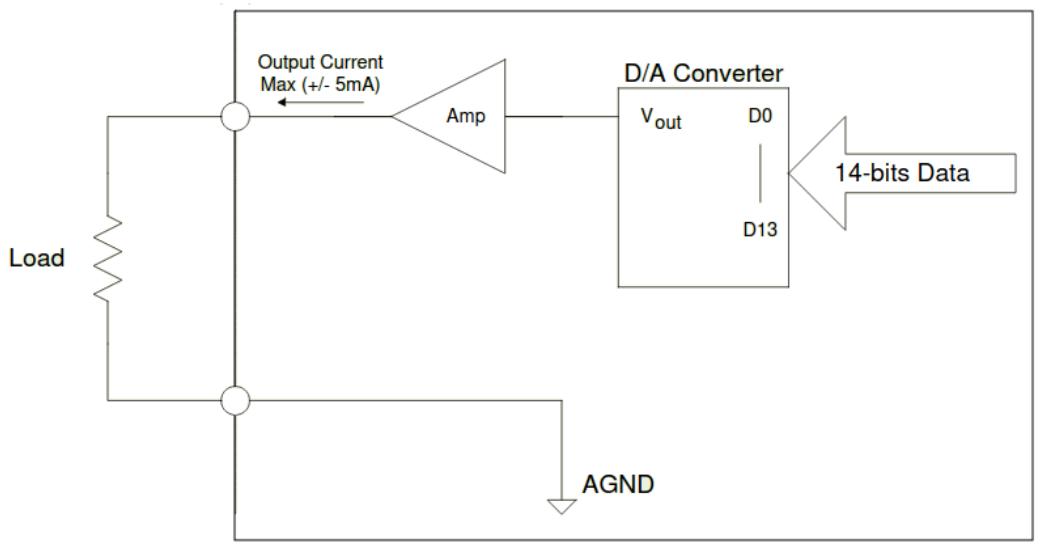


Схема подключения аналогового выхода как источника напряжения



Цифровые выходы

Назначение контактов внешнего сигнального разъема CON2 (снизу на передней планке):

Контакт	Порт	Сигнал
1	C	DO1
2	C	DO2
3	C	DO3
4	C	DO4
5	C	DO5
6	C	DO6

Контакт	Порт	Сигнал
7	C	DO7
8	C	DO8
9	D	DO1
10	D	DO2
11	D	DO3
12	D	DO4
13	D	DO5
14	D	DO6
15	D	DO7
16	D	DO8
17	-	GND
18	-	GND
19	-	+5V
20	-	+12V

Обозначения:
GND: Цепь “общего провода” для цифровых входов и выходов.
+5 В: Выход +5 В питания внешних цифровых узлов.
+12 В: Выход +12 В питания внешних цифровых узлов.
Порт: Обозначение порта цифрового выхода.
DO1..DO8: Номер цифрового выхода на цифровом порте.

Цифровые входы

Назначение контактов внешнего сигнального разъема CON3 (сверху на передней планке):

Контакт	Порт	Сигнал
1	A	DI1
2	A	DI2
3	A	DI3
4	A	DI4
5	A	DI5
6	A	DI6
7	A	DI7
8	A	DI8
9	B	DI1
10	B	DI2
11	B	DI3
12	B	DI4
13	B	DI5
14	B	DI6
15	B	DI7
16	B	DI8

Контакт	Порт	Сигнал
17	-	GND
18	-	GND
19	-	+5
20	-	+12

Обозначения:
GND: Цепь “общего провода” для цифровых входов и выходов.
+5 В: Выход +5 В питания внешних цифровых узлов.
+12 В: Выход +12 В питания внешних цифровых узлов.
Порт: Обозначение порта цифрового входа.
DI1..DI8: Номер цифрового входа на цифровом порте.

Модуль RITMeX GP-AD-24

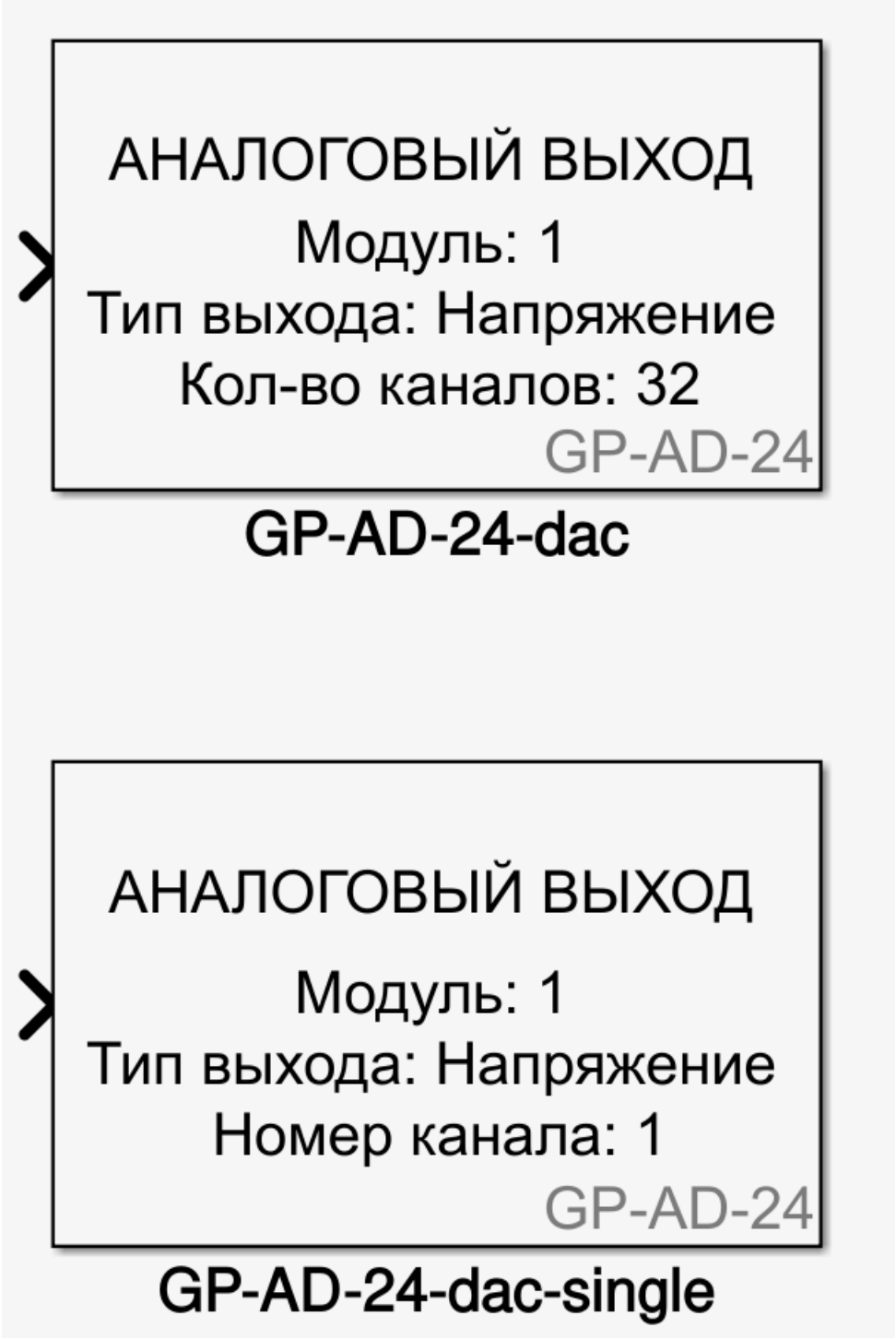
Содержание

- Описание
- Библиотека блоков
- Аналоговые выходы
  - Характеристики ЦАП
  - Блок “Аналоговый выход” многоканальный в Simulink
  - Блок “Аналоговый выход” одноканальный в Simulink

Описание

Аналоговый модуль вывода RITMeX GP-AD-24, обеспечивающий поддержку аналоговых выходов (напряжение), оснащенный 16-битными ЦАП, в КПМ “РИТМ”.

Библиотека блоков



Аналоговые выходы

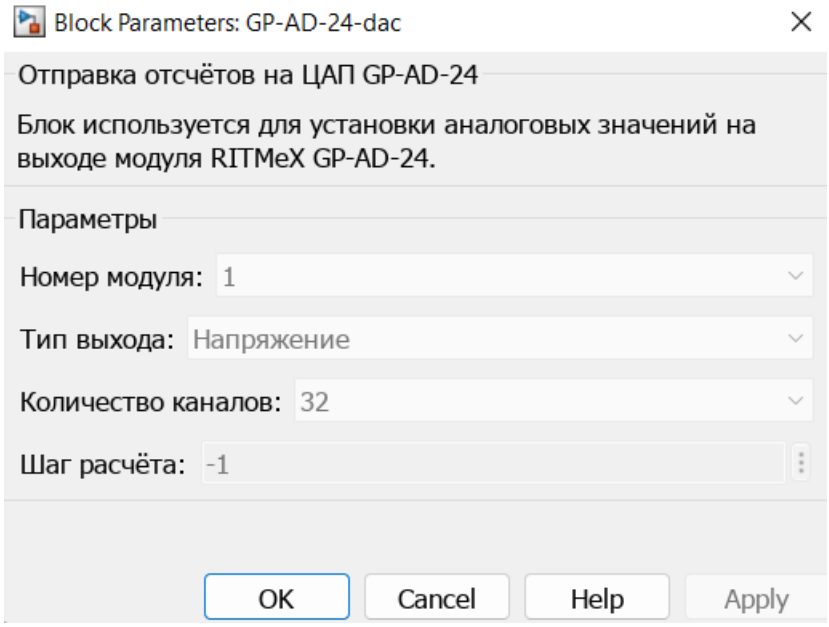
Характеристики ЦАП

- Количество выходных каналов: 32;
- Разрядность ЦАП: 16 бит;
- Выходной диапазон: ±10 В, 0-20 мА или 4-20 мА.
- Скорость нарастания: 0.7 В/мкс
- Сопротивление нагрузки: > 1 кОм

Блок “Аналоговый выход” многоканальный в Simulink

Блок используется для задания значения аналоговых выходов модуля ввода-вывода RITMeX GP-AD-24.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Тип выхода***: Устанавливает тип аналоговых выходов модуля, напряжение, ток 0-20 мА или ток 4-20 мА.
- ***Кол-во каналов***: Устанавливает значение задействованных аналоговых выходов модуля.
  Важно: Подключение контактов осуществляется в соответствии со схемой в разделе Назначение пинов GP-AD-24.
- ***Шаг расчёта***: Позволяет задать шаг расчета в секундах для отправки отсчетов на ЦАП из приложения реального времени.

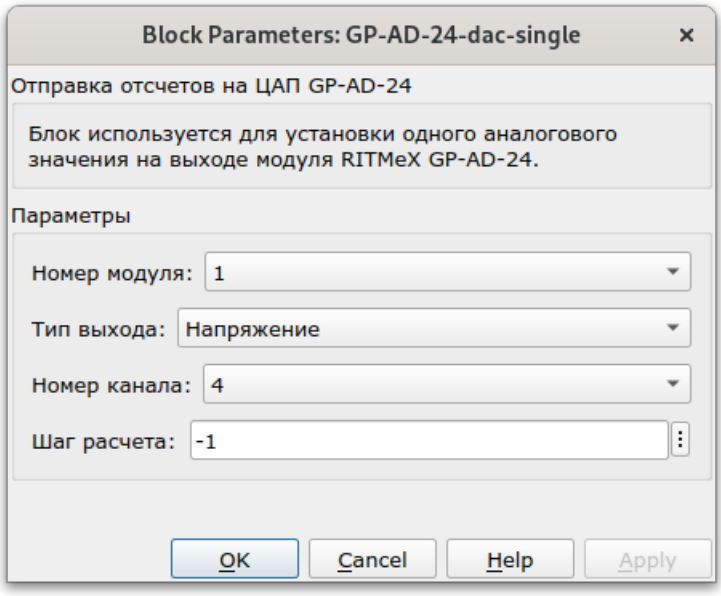
Вход блока

На вход блока подается массив значений типа double, которые интерпретируются как значения напряжения в вольтах или тока в миллиамперах. Каждому элементу массива соответствует выходной канал с порядковым номером: первый элемент массива соответствует первому выходному каналу, второй канал соответствует второму выходному каналу и т.д.

Блок “Аналоговый выход” одноканальный в Simulink

Блок используется для задания значения одного из аналоговых выходов модуля ввода-вывода RITMeX GP-AD-24.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Тип выхода***: Устанавливает тип аналоговых выходов модуля, напряжение, ток 0-20 мА или ток 4-20 мА.
- ***Номер канала***: Устанавливает номер задействованного аналогового выхода модуля.
  Важно: Подключение контактов осуществляется в соответствии со схемой в разделе Назначение пинов GP-AD-24.
- ***Шаг расчёта***: Позволяет задать шаг расчета в секундах для отправки отсчетов на ЦАП из приложения реального времени.

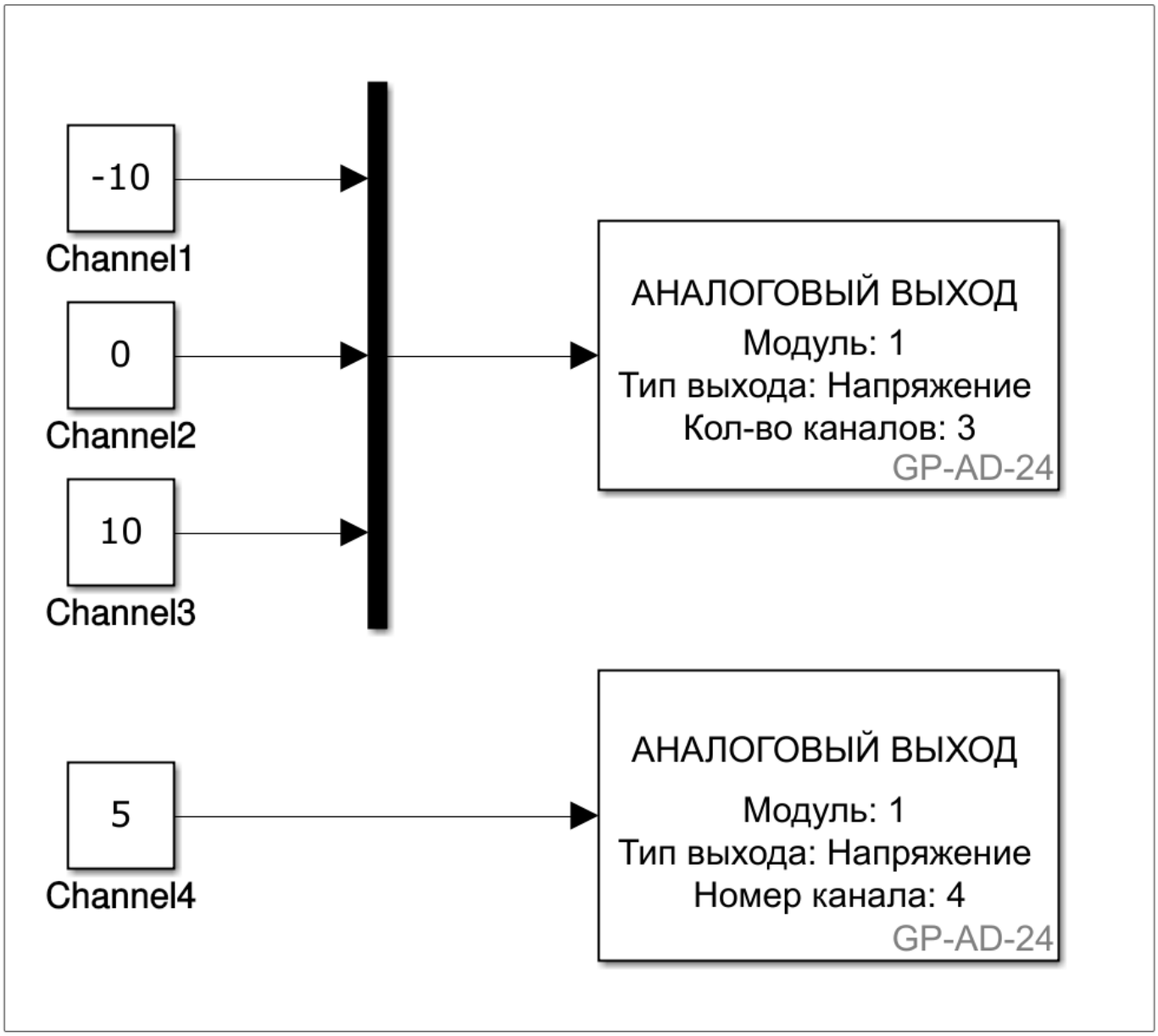
Вход блока

На вход блока подается значение типа double, которое интерпретируется как значение напряжения в вольтах или тока в миллиамперах.

Примеры и демонстрации GP-AD-24

Пример использования блоков GP-AD-24.

```
open_system('ritmdemo_block_gpad24.slx')
```

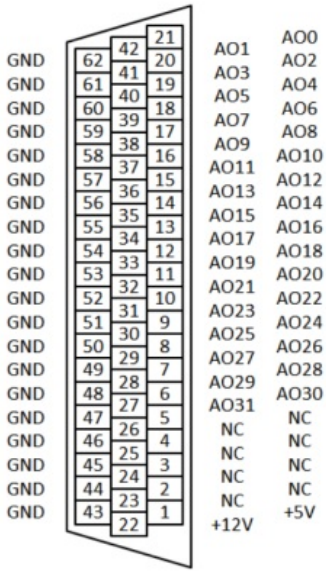


Назначение пинов GP-AD-24

Содержание

- Аналоговые выходы
  - Схема подключения аналогового выхода в режиме источника напряжения
  - Схема подключения аналогового выхода в режиме источника тока

Аналоговые выходы



Назначение контактов на поставляемом клеммном модуле:

Сигнал	Контакт	Описание
AO0	21	Выходной аналоговый канал 0
AO1	42	Выходной аналоговый канал 1
AO2	20	Выходной аналоговый канал 2
AO3	41	Выходной аналоговый канал 3
AO4	19	Выходной аналоговый канал 4
AO5	40	Выходной аналоговый канал 5
AO6	18	Выходной аналоговый канал 6
AO7	39	Выходной аналоговый канал 7
AO8	17	Выходной аналоговый канал 8
AO9	38	Выходной аналоговый канал 9
AO10	16	Выходной аналоговый канал 10
AO11	37	Выходной аналоговый канал 11
AO12	15	Выходной аналоговый канал 12
AO13	36	Выходной аналоговый канал 13
AO14	14	Выходной аналоговый канал 14
AO15	35	Выходной аналоговый канал 15

Сигнал	Контакт	Описание
AO16	13	Выходной аналоговый канал 16
AO17	34	Выходной аналоговый канал 17
AO18	12	Выходной аналоговый канал 18
AO19	33	Выходной аналоговый канал 19
AO20	11	Выходной аналоговый канал 20
AO21	32	Выходной аналоговый канал 21
AO22	10	Выходной аналоговый канал22
AO23	31	Выходной аналоговый канал 23
AO24	9	Выходной аналоговый канал 24
AO25	30	Выходной аналоговый канал 25
AO26	8	Выходной аналоговый канал 26
AO27	29	Выходной аналоговый канал 27
AO28	7	Выходной аналоговый канал 28
AO29	28	Выходной аналоговый канал 29
AO30	6	Выходной аналоговый канал 30
AO31	27	Выходной аналоговый канал 31
+12V	22	+12 В для внешнего питания
+5V	1	+5 V для внешнего питания
GND	43 ~ 63	Земля
NC	2 ~ 5, 23 ~ 26	Не используются

Для выхода ***+12V*** ограничение по току - не более 100 мА.
Для выхода ***+5V*** ограничение по току - не более 200 мА.

Схема подключения аналогового выхода в режиме источника напряжения

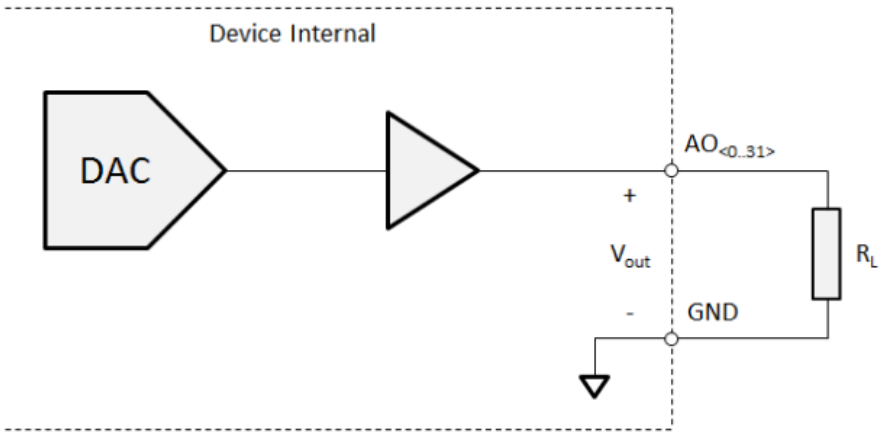
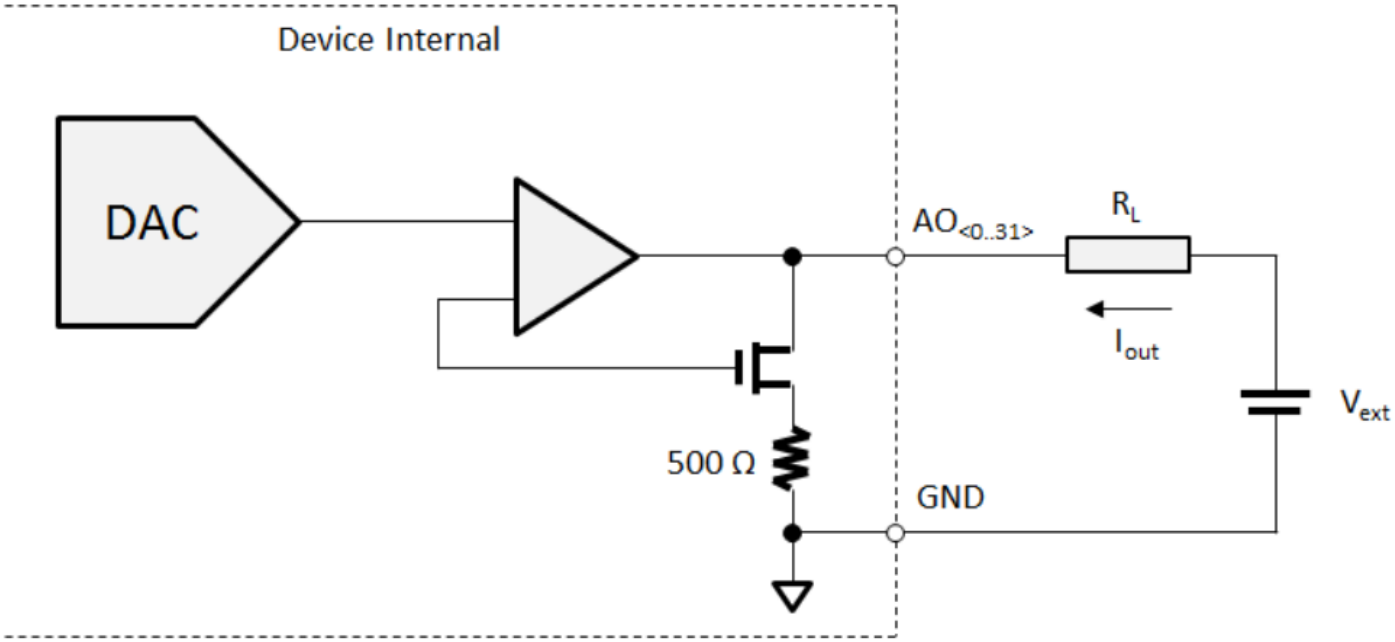


Схема подключения аналогового выхода в режиме источника тока


Device Internal
DAC
AO<0..31>
R_L
I_out
V_ext
500 Ω
GND

Модуль RITMeX GP-ID-24

Содержание

- Описание
- Характеристики модуля
- Библиотека блоков
- Блок “Цифровые входы” в Simulink
  - Маска блока
  - Параметры блока
- Блок “Цифровые выходы” в Simulink
  - Маска блока
  - Параметры блока

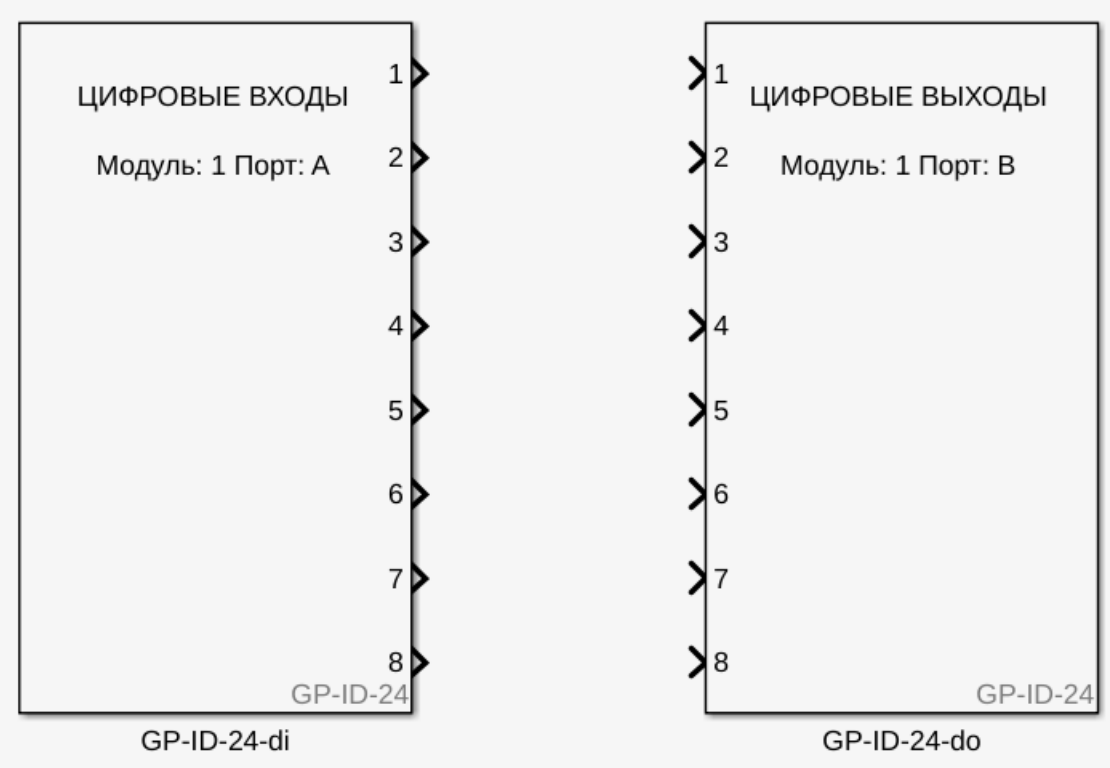
Описание

Цифровой модуль ввода-вывода RITMeX GP-ID-24, обеспечивающий поддержку цифровых входов и цифровых выходов в КПМ “РИТМ”.

Характеристики модуля

Количество каналов: 24 цифровых канала, сгруппированных в 3 порта: **A**, **B** и **C**. Каждый из каналов может быть настроен на работу как на вход, так и на выход.
Диапазон напряжений: 0...+0,8 В (“ноль”), +2,0...+5,0 В (“единица”).

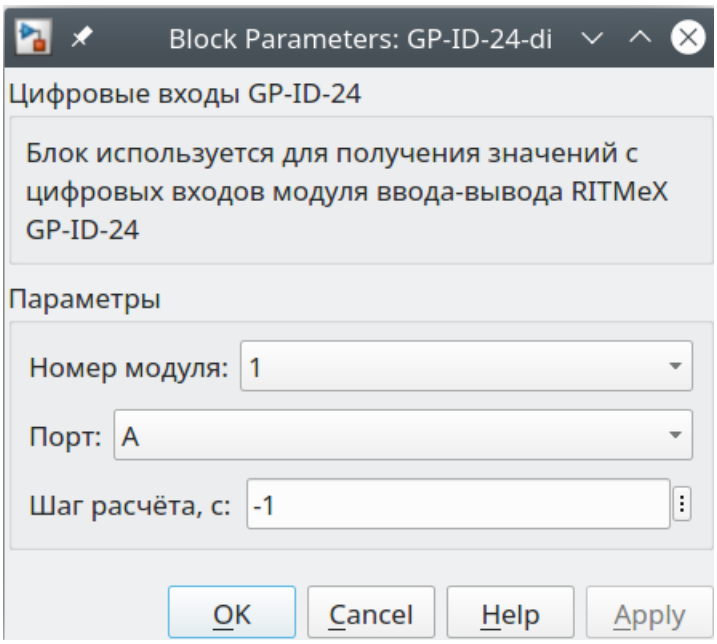
Библиотека блоков



Блок “Цифровые входы” в Simulink

Блок используется для получения значений с цифровых входов модуля ввода-вывода RITMeX GP-ID-24.

Маска блока



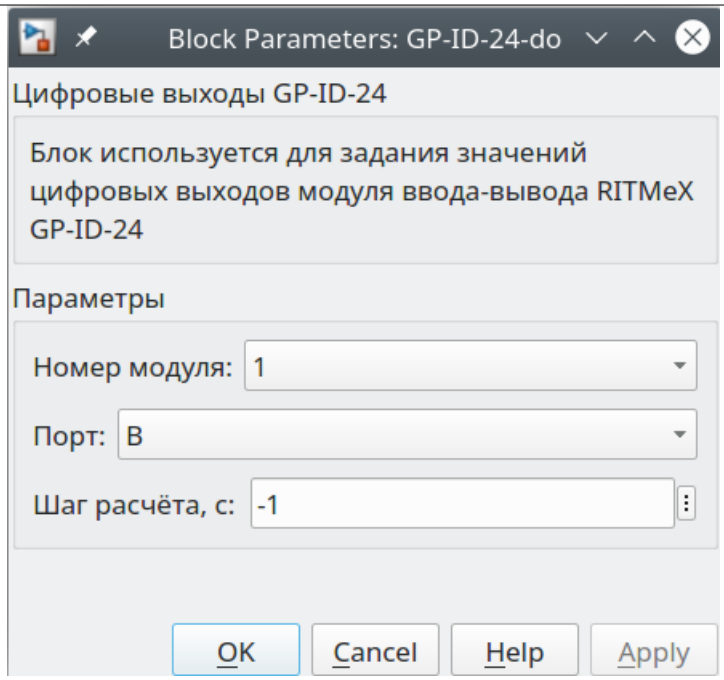
Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Порт**: Обозначение порта, со входов которого блок будет считывать значения.
- **Шаг расчета**: Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Блок “Цифровые выходы” в Simulink

Блок используется для задания значений цифровых выходов модуля ввода-вывода RITMeX GP-ID-24.

Маска блока



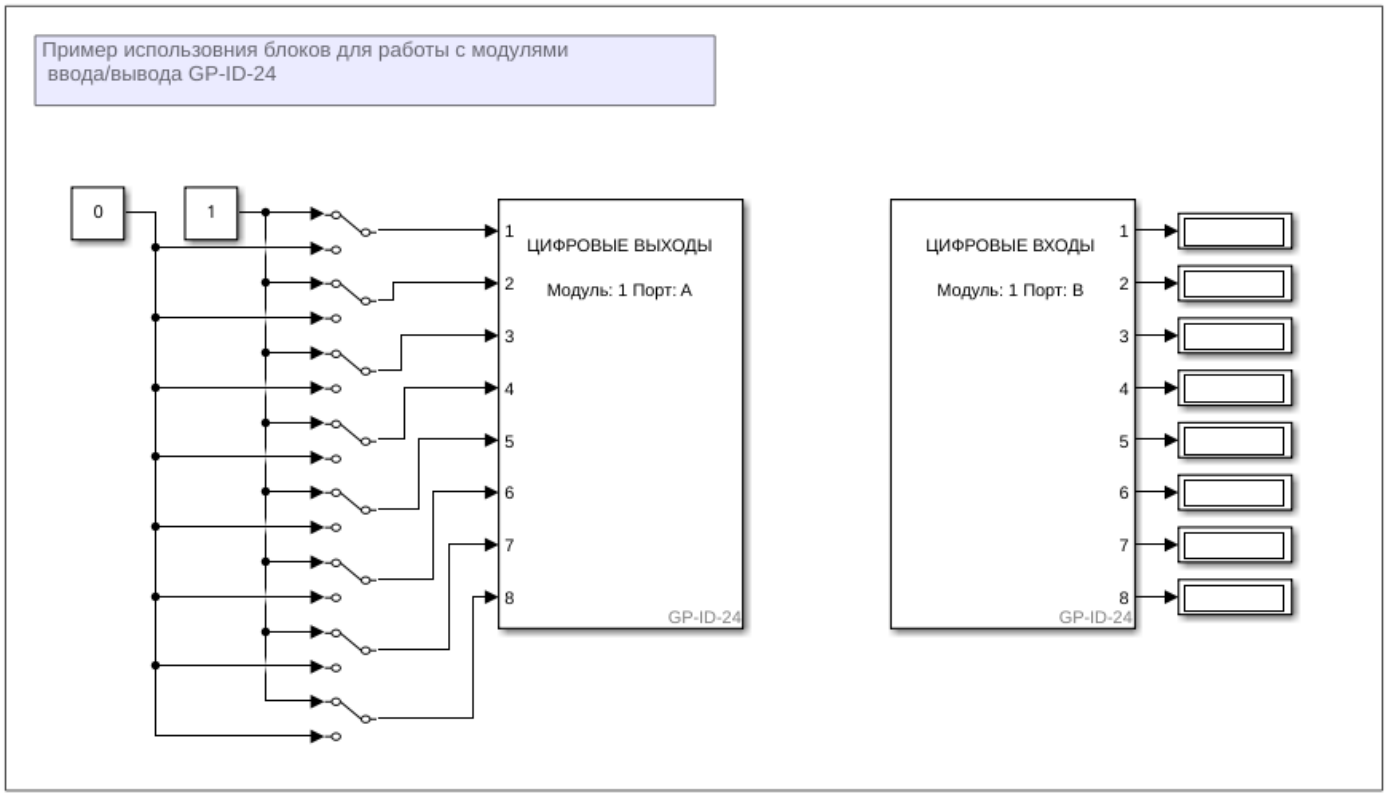
Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Порт**: Обозначение порта, на выходы которого блок будет подавать значения.
- **Шаг расчета**: Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Примеры и демонстрации GP-ID-24

Пример использования блоков GP-ID-24.

```
open_system('ritmdemo_block_gpid24.slx')
```



Назначение пинов GP-ID-24

Содержание

- Цифровые входы и цифровые выходы

Цифровые входы и цифровые выходы

Назначение контактов на поставляемом клеммном модуле:

Контакт	Порт	Сигнал	Контакт	Порт	Сигнал
1	-	-	20	-	+5V
2	-	-	21	-	GND
3	B	DIO8	22	C	DIO8
4	B	DIO7	23	C	DIO7
5	B	DIO6	24	C	DIO6
6	B	DIO5	25	C	DIO5
7	B	DIO4	26	C	DIO4
8	B	DIO3	27	C	DIO3
9	B	DIO2	28	C	DIO2
10	B	DIO1	29	C	DIO1
11	-	GND	30	A	DIO8
12	-	-	31	A	DIO7
13	-	GND	32	A	DIO6
14	-	-	33	A	DIO5
15	-	GND	34	A	DIO4
16	-	-	35	A	DIO3
17	-	GND	36	A	DIO2
18	-	+5V	37	A	DIO1
19	-	GND			

Обозначения:
DIO1..DIO8: Цифровые входы и выходы с 1 до 8 на указанном порту.
GND: Цепь “общего провода” для цифровых входов и выходов.
+5V: Выход +5 В питания внешних цифровых узлов.

Модуль RITMeX GP-ID-34

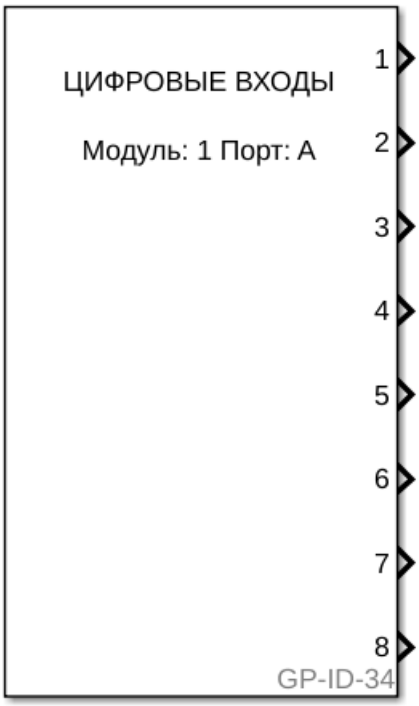
Содержание

- Описание
- Библиотека блоков
- Цифровые входы
  - Характеристики цифровых входов
  - Блок “Цифровые входы” в Simulink
- Цифровые выходы
  - Характеристики цифровых выходов
  - Блок “Цифровые выходы” в Simulink

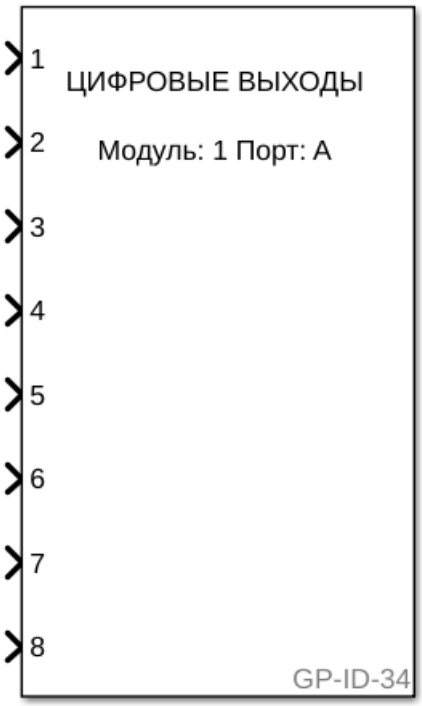
Описание

Цифровой модуль ввода-вывода RITMeX GP-ID-34, обеспечивающий поддержку цифровых входов и цифровых выходов в КПМ “РИТМ”.

Библиотека блоков



GP-ID-34-di



GP-ID-34-do

Цифровые входы

Характеристики цифровых входов

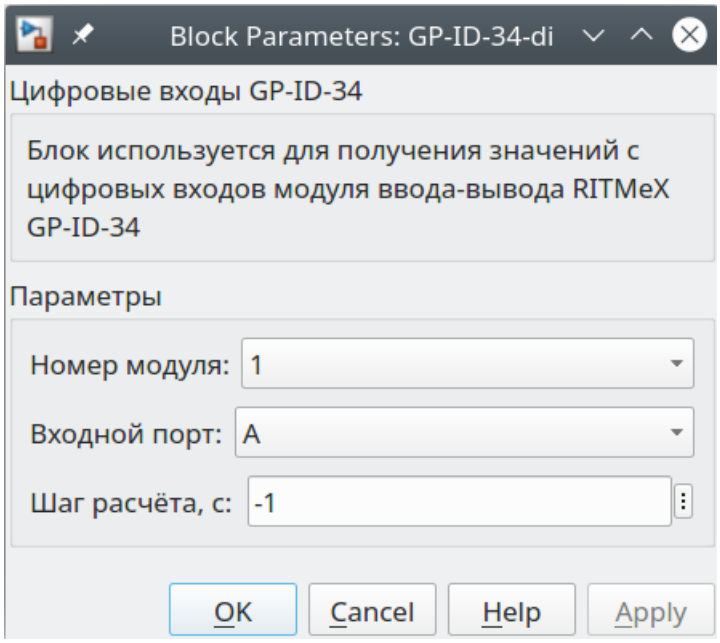
Количество каналов: 32 канала, сгруппированных в 4 порта: **A**, **B**, **C**, **D**.
Диапазон напряжений: 0…+1 В (“ноль”), +9,0…+24,0 В (“единица”).

Блок “Цифровые входы” в Simulink

Блок используется для получения значений с цифровых входов модуля ввода-вывода

RITMeX GP-ID-34.

Маска блока



Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Входной порт**: Обозначение порта, со входов которого блок будет считывать значения.
- **Шаг расчета**: Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Цифровые выходы

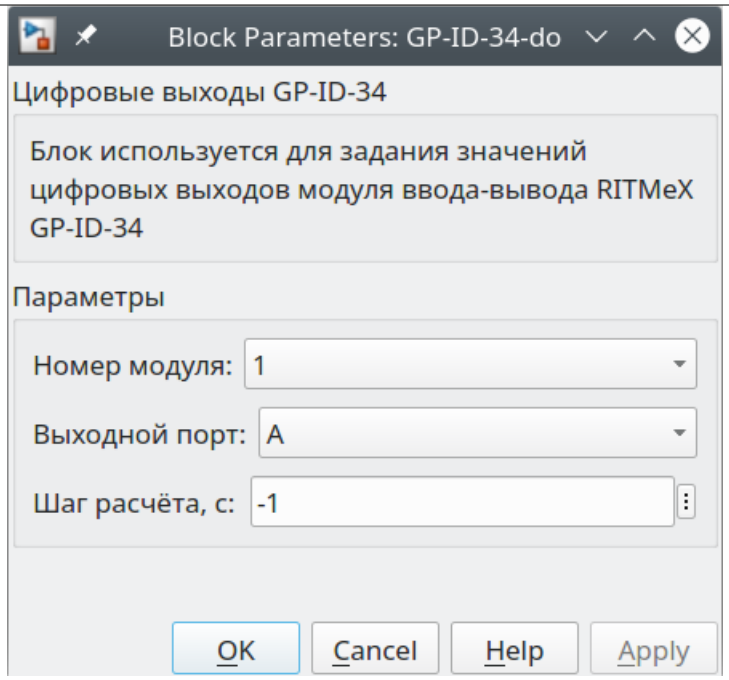
Характеристики цифровых выходов

Количество каналов: 32 канала, сгруппированных в 4 порта: **E**, **F**, **G**, **H**.
Диапазон напряжений: 0…+1 В (“ноль”), +9,0…+24,0 В (“единица”).

Блок “Цифровые выходы” в Simulink

Блок используется для задания значений цифровых выходов модуля ввода-вывода RITMeX GP-ID-34.

Маска блока



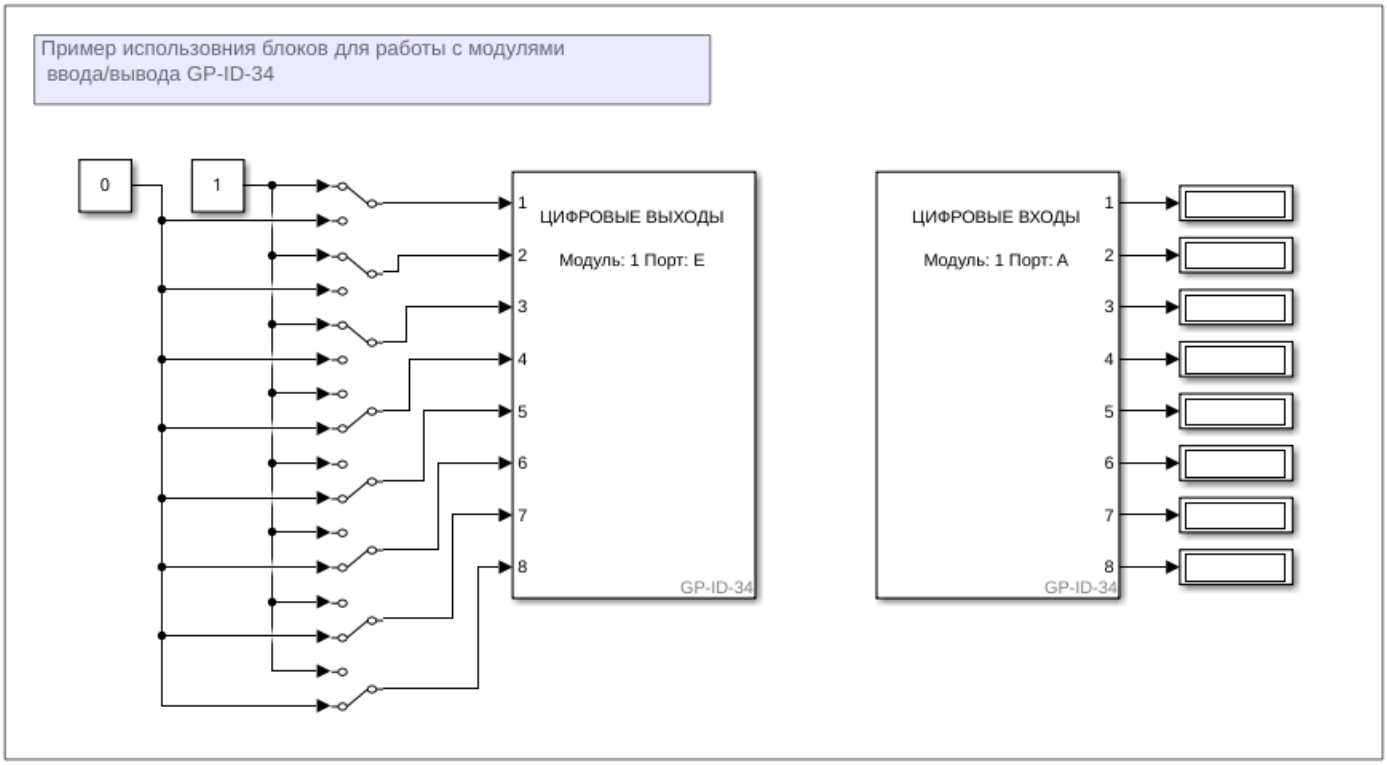
Параметры блока.

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Выходной порт**: Обозначение порта, на выходы которого блок будет подавать значения.
- **Шаг расчета**: Позволяет задать шаг расчета в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Примеры и демонстрации GP-ID-34

Пример использования блоков GP-ID-34.

```
open_system('ritmdemo_block_gpid34.slx')
```



Назначение пинов GP-ID-34

Содержание

- Цифровые входы и цифровые выходы
  - Схема подключения входного сигнала
  - Схема подключения выходного сигнала

Цифровые входы и цифровые выходы

Назначение контактов на *первом* поставляемом клеммном модуле для конфигурации модуля с 16 цифровыми входами и 16 цифровыми выходами:

Контакт	Порт	Сигнал	Контакт	Порт	Сигнал
1	-	EXT.GND	21	E	DO1
2	A	DI1	22	E	DO2
3	A	DI2	23	E	DO3
4	A	DI3	24	E	DO4
5	A	DI4	25	E	DO5
6	A	DI5	26	E	DO6
7	A	DI6	27	E	DO7
8	A	DI7	28	E	DO8
9	A	DI8	29	F	DO1
10	B	DI1	30	F	DO2
11	B	DI2	31	F	DO3
12	B	DI3	32	F	DO4
13	B	DI4	33	F	DO5
14	B	DI5	34	F	DO6
15	B	DI6	35	F	DO7
16	B	DI7	36	F	DO8
17	B	DI8	37	-	EXT.PWR
18	-	INT.GND			
19	-	N/A			
20	-	EXT.GND			

Назначение контактов на *втором* поставляемом клеммном модуле для конфигурации модуля с 32 цифровыми входами и 32 цифровыми выходами:

Контакт	Порт	Сигнал	Контакт	Порт	Сигнал
1	-	EXT.GND	21	G	DO1
2	C	DI1	22	G	DO2
3	C	DI2	23	G	DO3
4	C	DI3	24	G	DO4
5	C	DI4	25	G	DO5

Контакт	Порт	Сигнал	Контакт	Порт	Сигнал
6	C	DI5	26	G	DO6
7	C	DI6	27	G	DO7
8	C	DI7	28	G	DO8
9	C	DI8	29	H	DO1
10	D	DI1	30	H	DO2
11	D	DI2	31	H	DO3
12	D	DI3	32	H	DO4
13	D	DI4	33	H	DO5
14	D	DI5	34	H	DO6
15	D	DI6	35	H	DO7
16	D	DI7	36	H	DO8
17	D	DI8	37	-	EXT.PWR
18	-	INT.GND			
19	-	N/A			
20	-	EXT.GND			

Обозначения:
DI1..DI8: Цифровые входы с 1 до 8 на указанном порту.
DO1..DO8: Цифровые выходы с 1 до 8 на указанном порту.
INT.GND: Земля внутреннего источника питания.
ETX.GND: Земля внешнего источника питания.
ETX.PWR: Плюс внешнего источника питания.

Важно: Для работы модуля необходимо подключить внешнее питание. Для этого необходимо подключить порты ***GND*** и ***INT.GND*** к ***Земле*** внешнего источника питания, а на контакт ***EXT.PWR*** подать напряжение 9..24 В.

Схема подключения входного сигнала

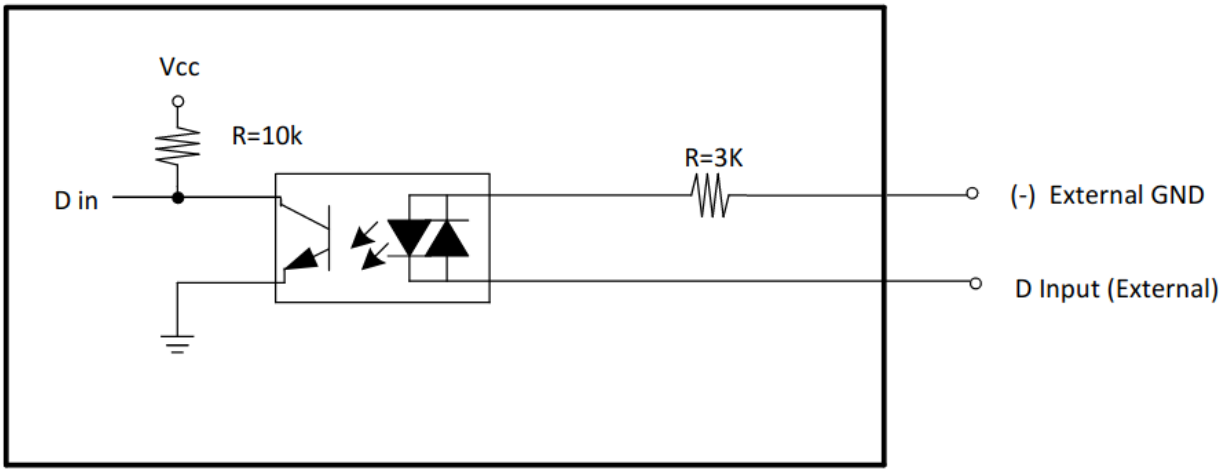
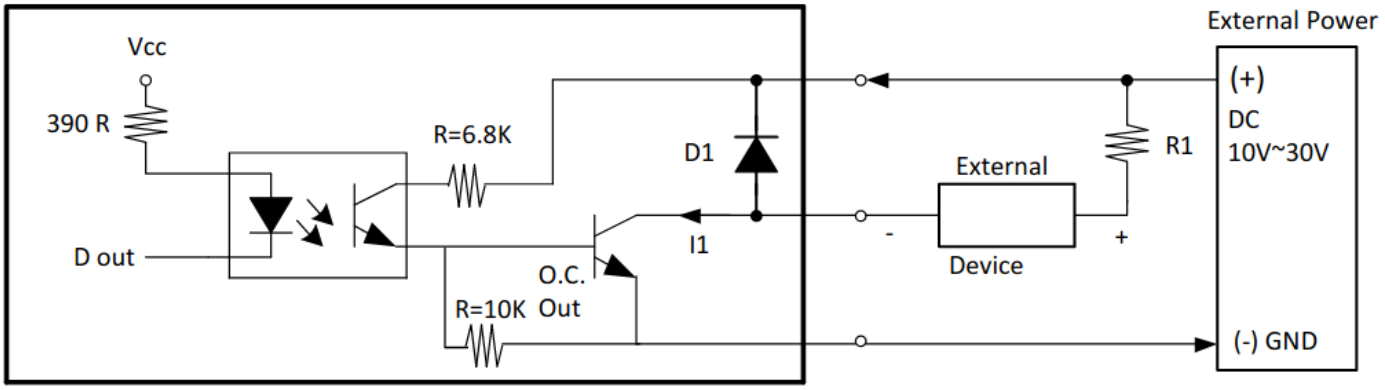


Схема подключения выходного сигнала


Vcc
390 R
D out
R=6.8K
D1
I1
O.C.
Out
R=10K
External
Device
-
+
R1
External Power
(+)
DC
10V~30V
(-) GND

Модуль RITMeX GP-EL-64

Содержание

- Описание
- Библиотека блоков
- Цифровые входы
  - Характеристики цифровых входов с открытым коллектором
  - Блок “Цифровые входы открытый коллектор” в Simulink
  - Характеристики цифровых входов TTL уровня
  - Блок “Цифровые входы TTL” в Simulink
- Цифровые выходы
  - Характеристики цифровых выходов с открытым коллектором
  - Блок “Цифровые выходы открытый коллектор” в Simulink
  - Характеристики цифровых выходов TTL уровня
  - Блок “Цифровые выходы открытый коллектор” в Simulink

Описание

Цифровой модуль ввода-вывода RITMeX GP-EL-64, обеспечивающий поддержку цифровых входов и цифровых выходов в КПМ “РИТМ”.

Библиотека блоков

Цифровые входы

Характеристики цифровых входов с открытым коллектором

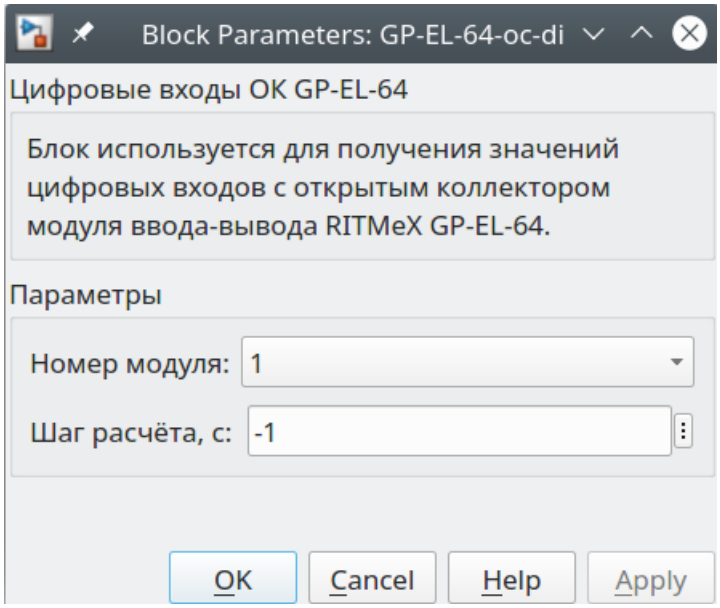
Количество каналов: 16 входных каналов с открытым коллектором.
Диапазон напряжений: 0...+1 В (“ноль”), +2,4...+35,0 В (“единица”).

Блок “Цифровые входы открытый коллектор” в Simulink

Блок используется для получения значений с цифровых входов с открытым коллектором

модуля ввода-вывода RITMeX GP-EL-64.

Маска блока



Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

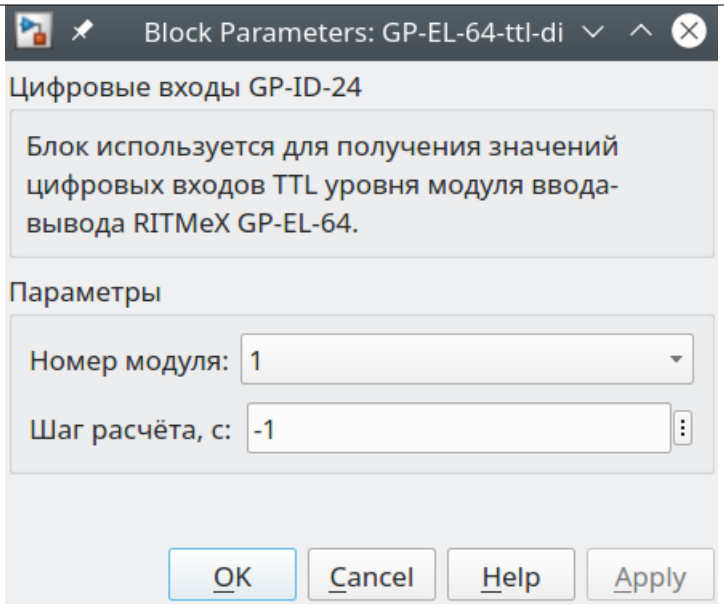
Характеристики цифровых входов TTL уровня

Количество каналов: 16 входных каналов TTL уровня.
Диапазон напряжений: 0...+1 В (“ноль”), +5 В (“единица”).

Блок “Цифровые входы TTL” в Simulink

Блок используется для получения значений с цифровых входов TTL уровня модуля ввода-вывода RITMeX GP-EL-64.

Маска блока



Параметры блока

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

Цифровые выходы

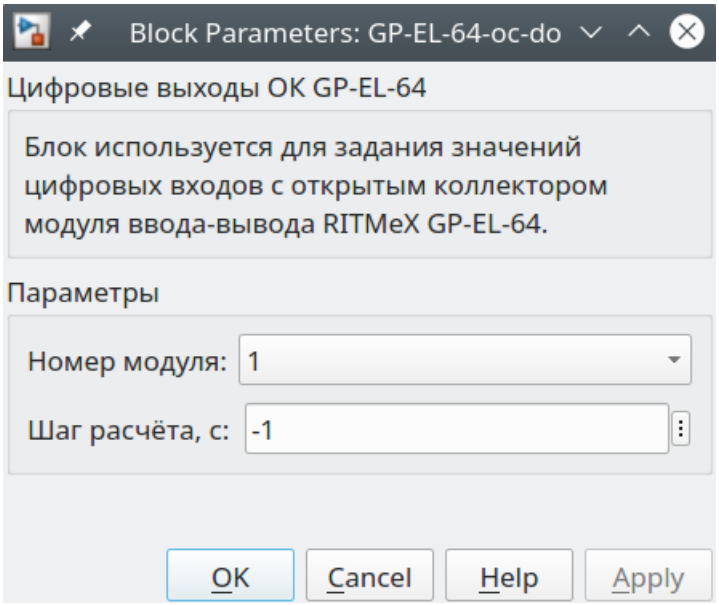
Характеристики цифровых выходов с открытым коллектором

Количество каналов: 16 выходных каналов с открытым коллектором.
Допустимое внешнее напряжение: до 35,0 В.

Блок “Цифровые выходы открытый коллектор” в Simulink

Блок используется для задания значений цифровых выходов с открытым коллектором модуля ввода-вывода RITMeX GP-EL-64.

Маска блока



Параметры блока.

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для установления значений цифровых каналов выхода из приложения реального времени.

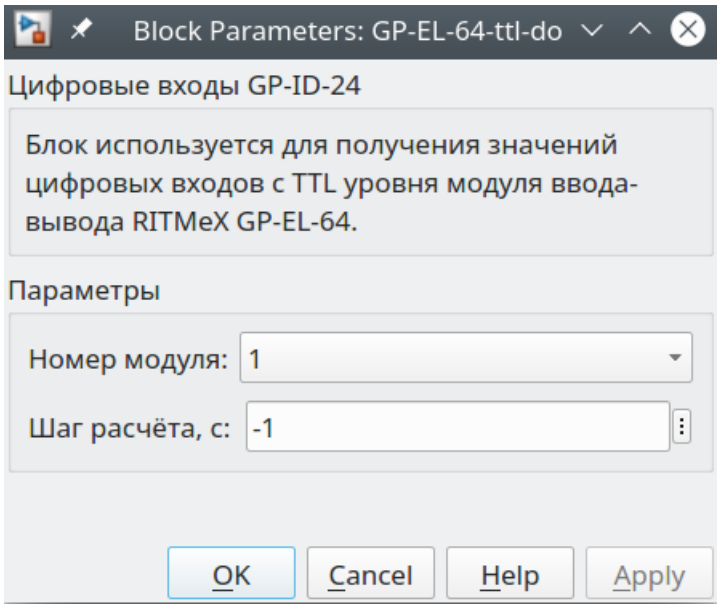
Характеристики цифровых выходов TTL уровня

Количество каналов: 16 выходных каналов TTL уровня.
Диапазон напряжений: 0…+1 В (“ноль”), +5 В (“единица”).

Блок “Цифровые выходы открытый коллектор” в Simulink

Блок используется для задания значений цифровых выходов TTL уровня модуля ввода-вывода RITMeX GP-EL-64.

Маска блока



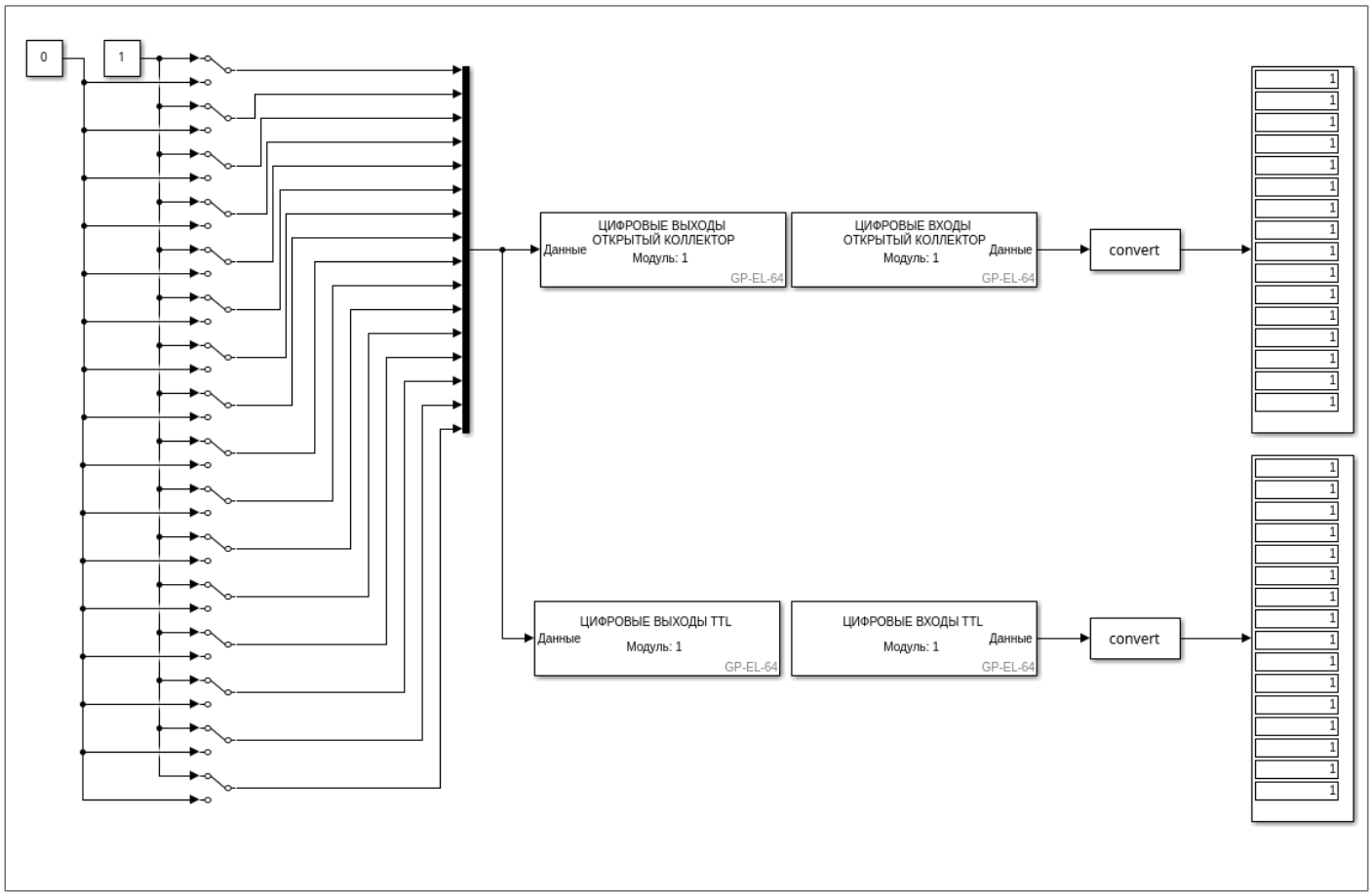
Параметры блока.

- **Номер модуля**: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для установления значений цифровых каналов выхода из приложения реального времени.

Примеры и демонстрации GP-EL-64

Пример использования блоков GP-EL-64.

```
open_system('ritmdemo_block_gpel64.slx')
```



Назначение пинов GP-EL-64

Содержание

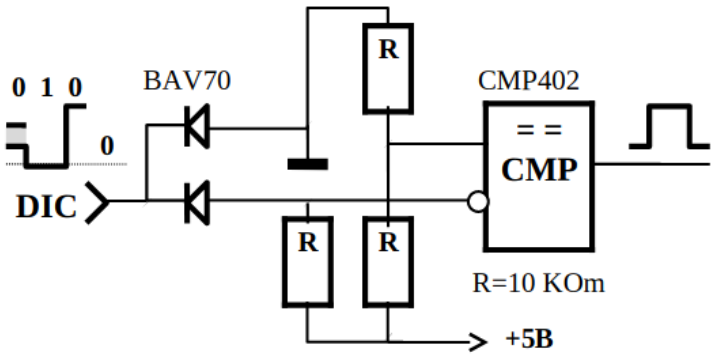
- Цифровые входы и цифровые выходы
  - Схема каналов с открытым коллектором
  - Схема каналов TTL уровня

Цифровые входы и цифровые выходы

Контакт	Цепь	Контакт	Цепь	Контакт	Цепь
1	DIT3	27	DOT4	53	DOC14
2	DIT5	28	DOT6	54	DOC15
3	DIT7	29	DOT8	55	DOC16
4	DIT9	30	DOT10	56	DIC13
5	DIT11	31	DOC1	57	DIC11
6	DOT1	32	DOC3	58	DIC9
7	DOT3	33	DOC5	59	DIC8
8	DOT5	34	DOC7	60	GND
9	DOT7	35	DOC9	61	-
10	DOT9	36	DOC11	62	-
11	DOT11	37	DIC2	63	GND
12	DOC2	38	DIC4	64	GND
13	DOC4	39	DIC6	65	GND
14	DOC6	40	DIT1	66	GND
15	DOC8	41	DIT2	67	GND
16	DOC10	42	DIT13	68	GND
17	DIC1	43	DIT14	69	GND
18	DIC3	44	DIT15	70	GND
19	DIC5	45	DIT16	71	GND
20	DIC7	46	DOT12	72	GND
21	DIT4	47	DOT13	73	DIC16
22	DIT6	48	DOT14	74	DIC15
23	DIT8	49	DOT15	75	DIC14
24	DIT10	50	DOT16	76	DIC12
25	DIT12	51	DOC12	77	DIC10
26	DOT2	52	DOC13	78	GND

Обозначения:
DIC1..DIC16: Цифровые входы с открытым коллектором 1 до 16.
DIT1..DIT16: Цифровые входы TTL уровня 1 до 16.
DOC1..DOC16: Цифровые выходы с открытым коллектором 1 до 16.
DOT1..DOT16: Цифровые выходы TTL уровня коллектором 1 до 16.

Схема каналов с открытым коллектором



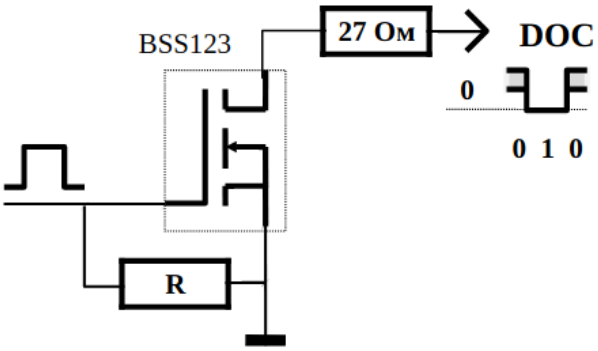
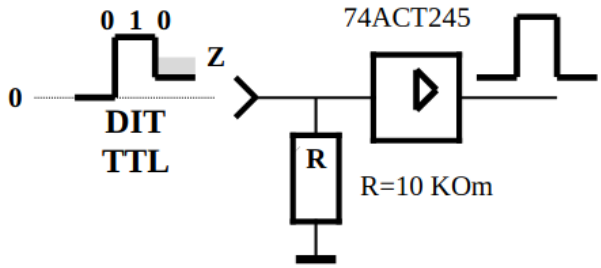
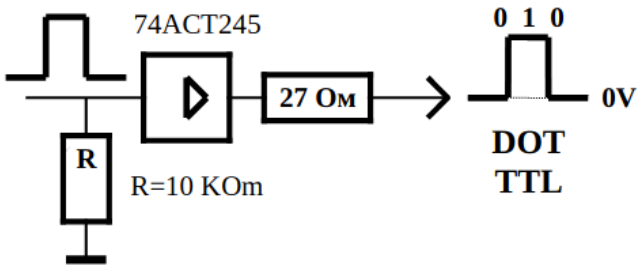


Схема каналов TTL уровня





Модули RS-232/RS-422/RS-485

Содержание

- Модули RITMeX RS-MO-45 и RITMeX RS-MO-35
- Библиотека блоков RITM-COM

Модули RITMeX RS-MO-45 и RITMeX RS-MO-35

Модули последовательных интерфейсов обеспечивают поддержку 16 и 8 портов RS-232/RS-422/RS-485 соответственно.

Библиотека блоков RITM-COM

Библиотека блоков для работы со встроенными COM-портами материнской платы и адаптерами USB-COM.

Модули RITMeX RS-MO-45 и RITMeX RS-MO-35

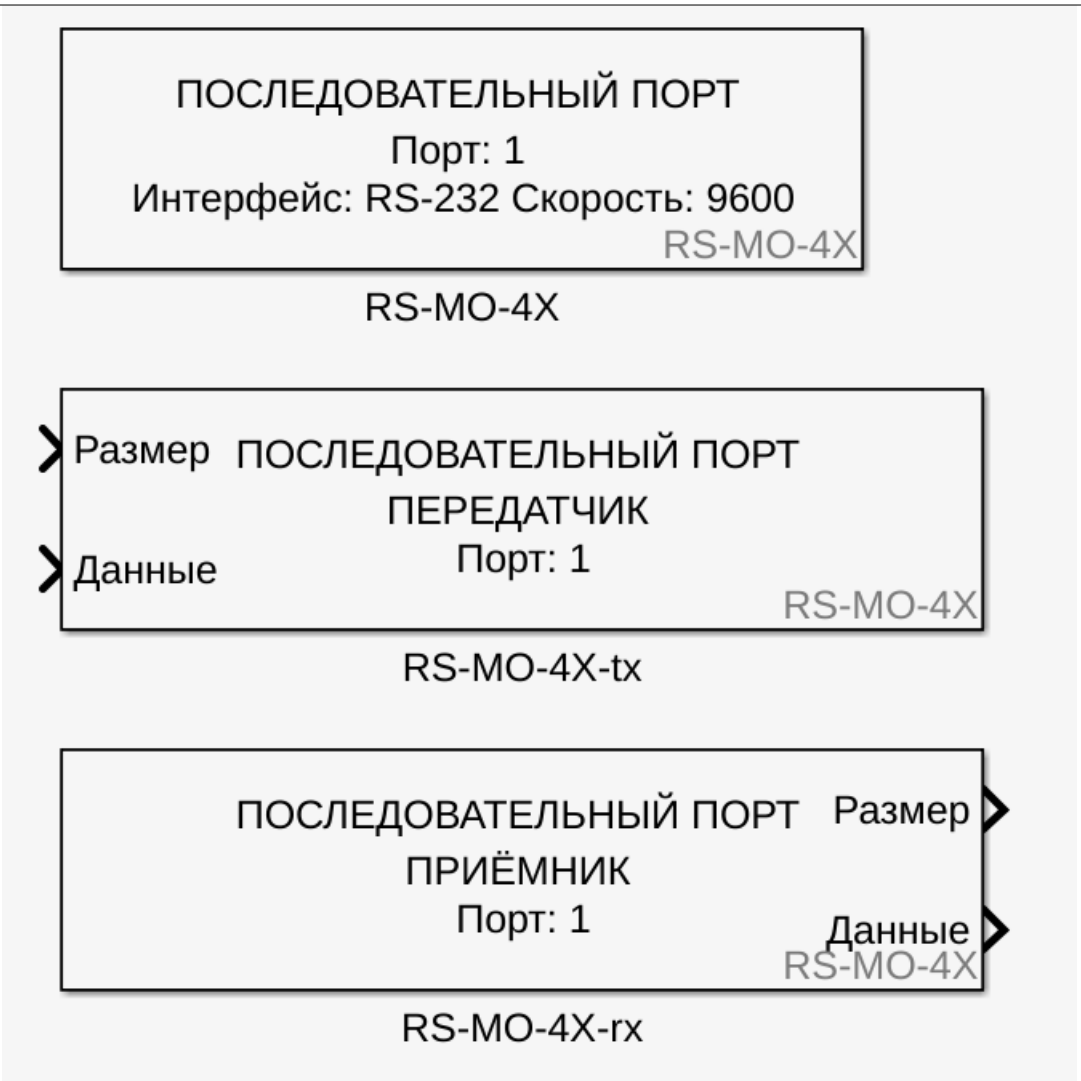
Содержание

- Описание
- Библиотека блоков
- Характеристики модуля
- Блок Последовательный порт
  - Маска блока
  - Параметры блока
- Блок Передатчик
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок Приёмник
  - Маска блока
  - Выходы блока

Описание

Модули последовательных интерфейсов RITMeX RS-MO-45 и RITMeX RS-MO-35, обеспечивающие поддержку интерфейсов RS-232/RS-422/RS-485 в КПМ “РИТМ”.

Библиотека блоков



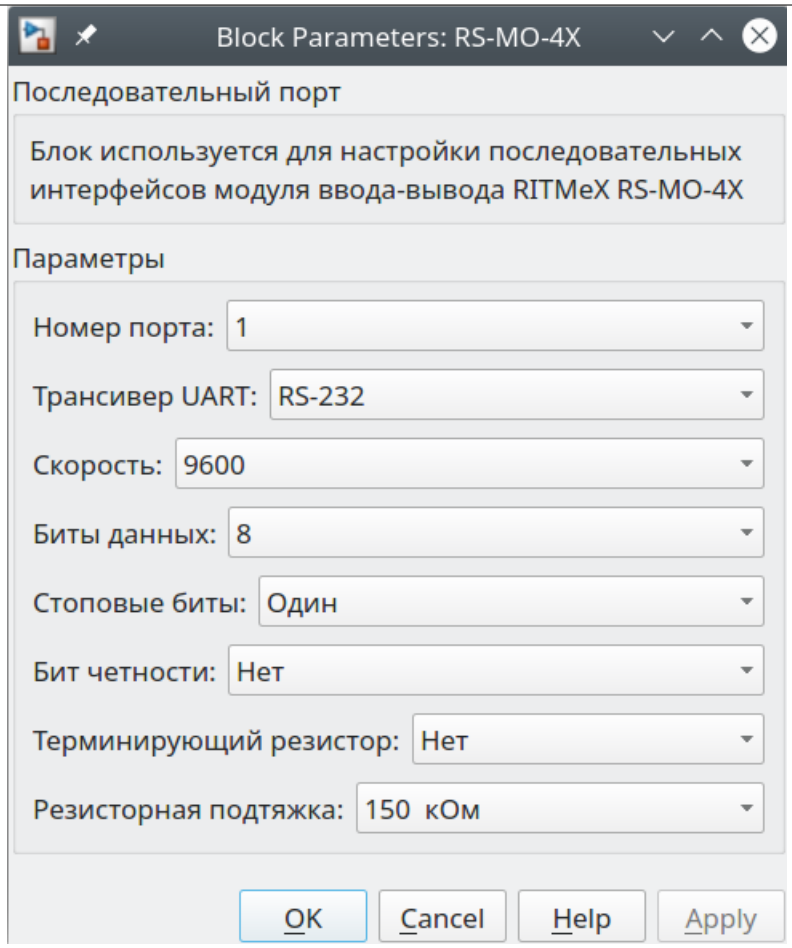
Характеристики модуля

Количество интерфейсов: 16 портов (RS-MO-45) или 8 портов (RS-MO-35) с возможностью выбора типа приемопередатчика RS-232/RS-422/RS-485.
Максимальная скорость: 921600 бод.
Гальваническая изоляция: 4 кВ.

Блок Последовательный порт

Блок используется для настройки последовательных интерфейсов модуля ввода-вывода RITMeX RS-MO-4X.

Маска блока



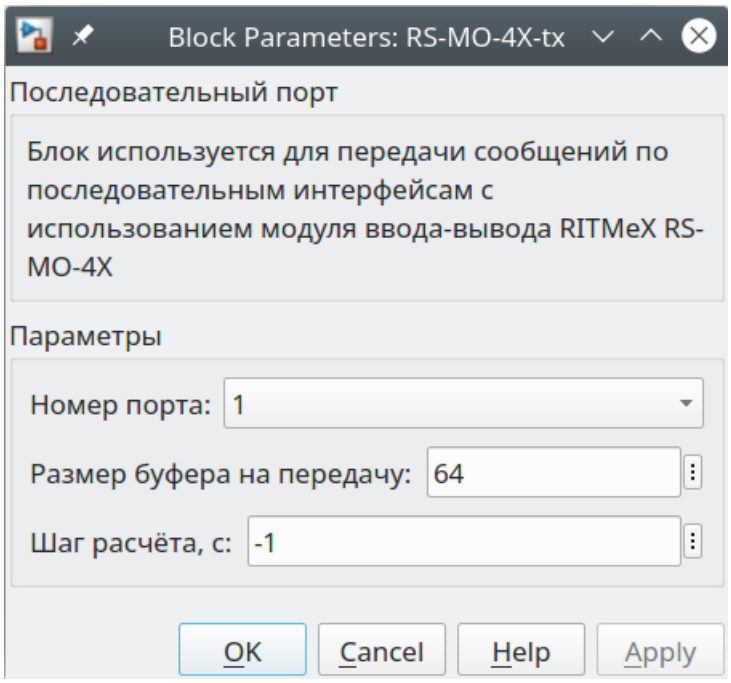
Параметры блока

- **Номер порта**: Задает номер порта для настройки. Номера портов обозначены на корпусе КПМ “РИТМ” и соединительных кабелях.
- **Трансивер UART**: Задает конфигурацию приемопередатчика последовательного интерфейса. Возможные варианты: *RS-232*, *RS-422*, *RS-485-2W* (полудуплекс), *RS-485-4W* (полный дуплекс).
- **Скорость**: Задает скорость приемопередачи. Возможные варианты: *1200*, *2400*, *4800*, *9600*, *19200*, *28800*, *38400*, *57600*, *115200*, *230400*, *460800*, *912600* бод.
- **Биты данных**: Задает количество бит данных приемопередачи. Возможные значения: *5-бит*, *6-бит*, *7-бит*, *8-бит*.
- **Стоповые биты**: Задает количество стоповых бит приемопередачи. Возможные значения: *одиночный*, *двойной*.
- **Бит четности**: Задает метод определения ошибок приемопередачи. Возможные значения: нет, проверка на четность (*Odd*), проверка на нечетность (*Even*), бит четности равен 0 (*Space*), бит четности равен 1 (*Mark*).
- **Терминирующий резистор**: Задает включение и отключение терминирующего резистора на линиях RS-422/RS-485 (120 Ом резистор). Опция недоступна при использовании интерфейса RS-232.
- **Резисторная подтяжка**: Задает значение резисторной подтяжки 1 кОм или 150 кОм. Установка резисторной подтяжки в 1 кОм доступна только при использовании интерфейсов RS-422/RS-485 и только для модуля RITMeX RS-MO-45.

Блок Передатчик

Блок используется для передачи сообщений по последовательным интерфейсам с использованием модуля ввода-вывода RITMeX RS-MO-4X.

Маска блока



Входы блока

- ***Размер***: Количество элементов из массива данных для передачи.
- ***Данные***: Данные для передачи. Массив 8-битных беззнаковых элементов (uint8).

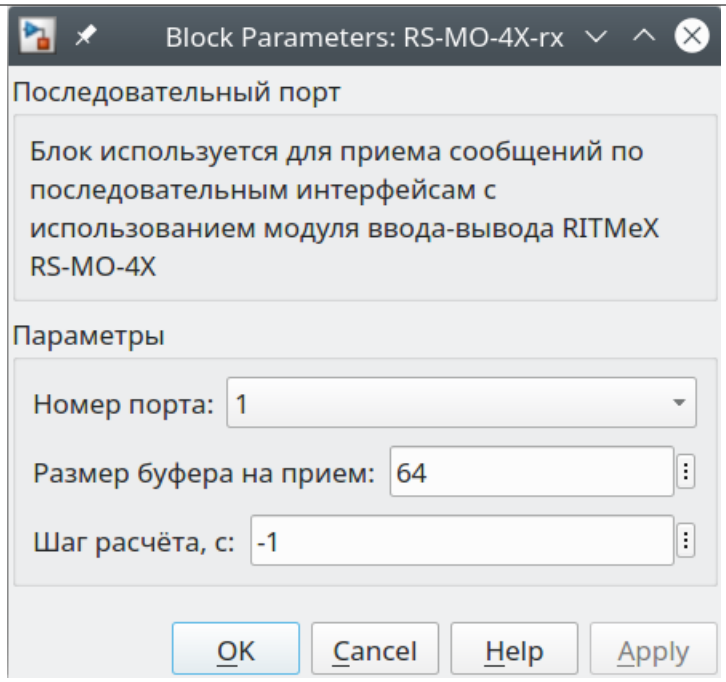
Параметры блока

- **Номер порта**: Задает номер порта для передачи.
- **Размер буфера на передачу**: Задает максимальный размер передаваемого массива.
- ***Шаг расчёта.*** Позволяет задать шаг расчета в секундах для передачи данных из приложение реального времени.

Блок Приёмник

Блок используется для приема сообщений по последовательным интерфейсам с использованием модуля ввода-вывода RITMeX RS-MO-4X.

Маска блока



Выходы блока

- ***Размер*** - количество принятых байт. Тип данных - uint32.
- ***Данные*** - принятые данные. Массив 8-битных беззнаковых элементов (uint8).

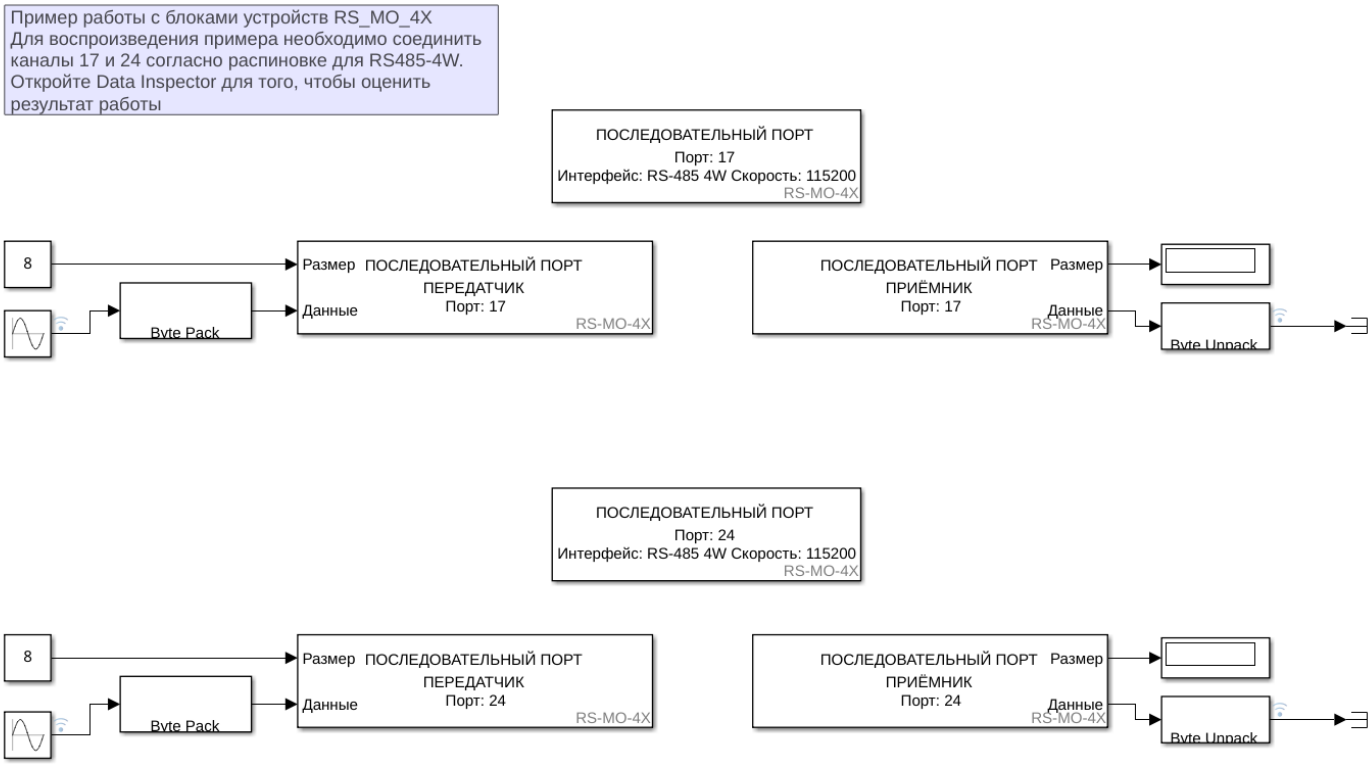
Параметры блока

- **Номер порта**: Задает номер порта для передачи.
- **Размер буфера на прием**: Задает максимальный размер принимаемого массива.
- ***Шаг расчёта.*** Позволяет задать шаг расчета в секундах для приема данных в приложение реального времени.

Примеры и демонстрации RS-MO-45 и RS-MO-35

Пример использования COM портов модуля RS-MO-45 и RS-MO-35.

```
open_system('ritmdemo_block_rsmo4x.slx')
```

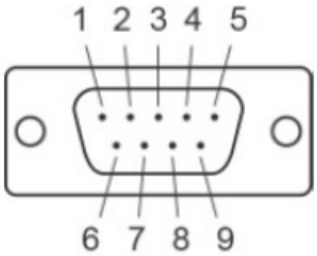


Назначение пинов RS-MO-45 и RS-MO-35

Модуль RS-MO-45 поставляется с переходниками с двух разъемов VHDCI 68 на 16 разъемов DB-9 Male.

Модуль RS-MO-35 поставляется с переходниками с одного разъема VHDCI 68 на 8 разъемов DB-9 Male.

Ниже приводится распиновка разъема DB-9:



Контакт	RS-232	RS-422/485-4W	RS485-2W
1	DCD	TxD- (A)	-
2	RxD	TxD+ (B)	-
3	TxD	RxD+ (B)	Data+ (B)
4	DTR	RxD- (A)	Data- (A)
5	GND	GND	GND
6	DSR	-	-
7	RTS	-	-
8	CTS	-	-

Библиотека блоков RITM-COM

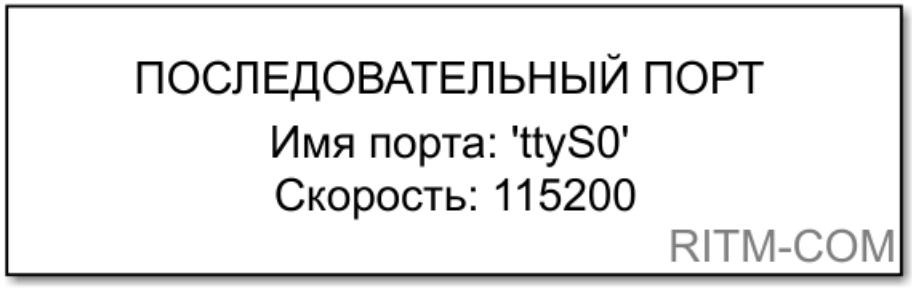
Содержание

- Описание
- Библиотека блоков
- Блок Последовательный порт
  - Маска блока
  - Параметры блока
- Блок Передатчик
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок Приёмник
  - Маска блока
  - Выходы блока

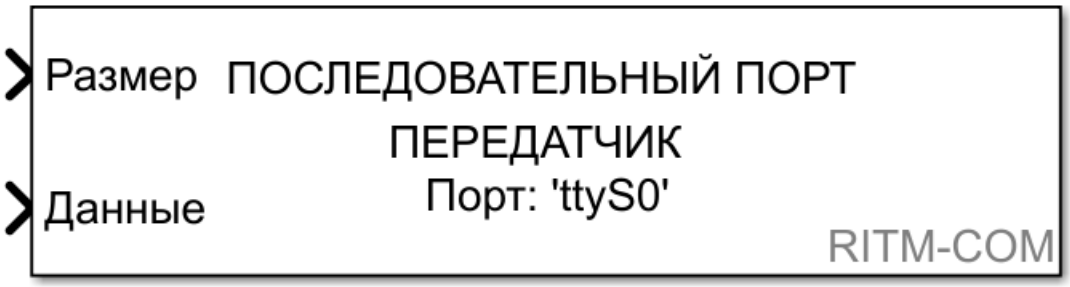
Описание

Библиотека блоков для работы со встроенным COM материнской платы и адаптерами USB-COM.

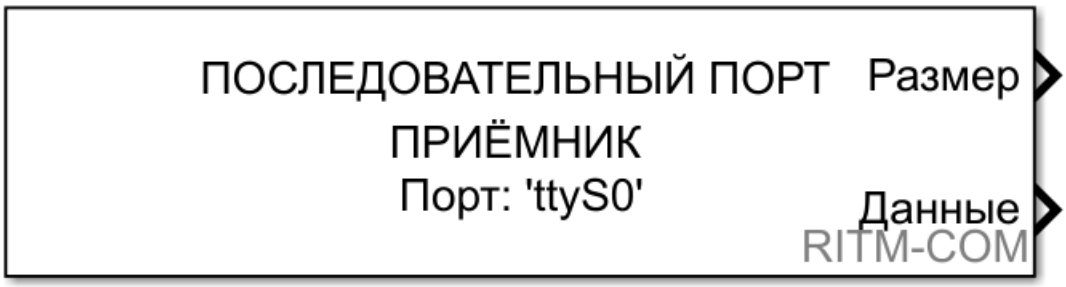
Библиотека блоков



RITM-COM-setup



RITM-COM-tx

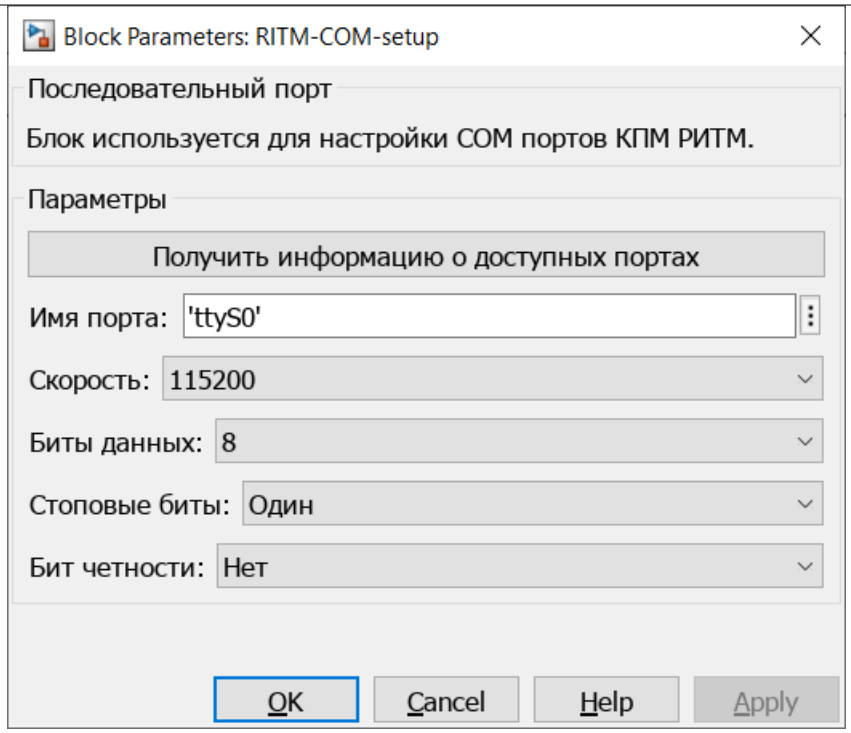


RITM-COM-rx

Блок Последовательный порт

Блок используется для настройки последовательных портов.

Маска блока



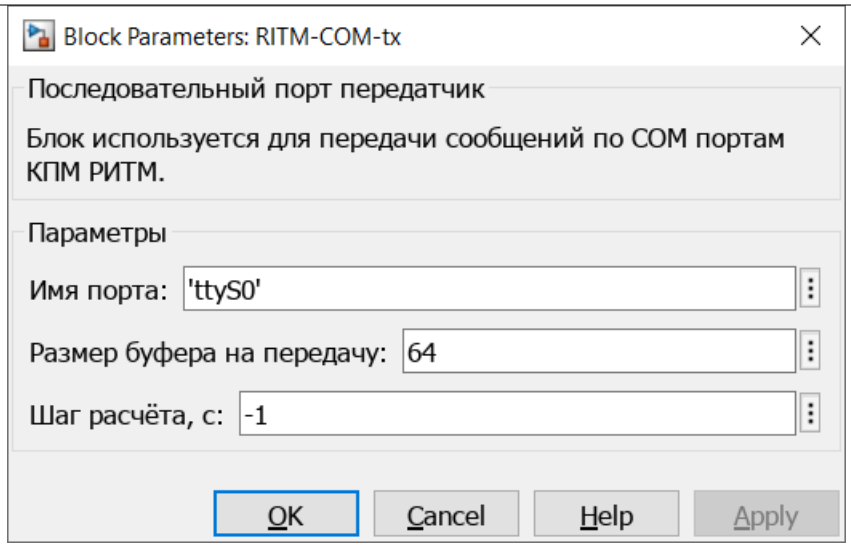
Параметры блока

- **Имя порта**: Задает имя порта для настройки. Имена доступных портов могут быть получены во
- **Скорость**: Задает скорость приемопередачи. Возможные варианты: *1200*, *2400*, *4800*, *9600*, *19200*, *28800*, *38400*, *57600*, *115200* бод.
- **Биты данных**: Задает количество бит данных приемопередачи. Возможные значения: *5-бит*, *6-бит*, *7-бит*, *8-бит*.
- **Стоповые биты**: Задает количество стоповых бит приемопередачи. Возможные значения: *одиночный*, *двойной*.
- **Бит четности**: Задает метод определения ошибок приемопередачи. Возможные значения: нет, проверка на четность (*Odd*), проверка на нечетность (*Even*), бит четности равен 0 (*Space*), бит четности равен 1 (*Mark*).

Блок Передатчик

Блок используется для передачи сообщений по COM портам КПМ РИТМ.

Маска блока



Входы блока

- ***Размер***: Количество элементов из массива данных для передачи.
- ***Данные***: Данные для передачи. Массив 8-битных беззнаковых элементов (uint8).

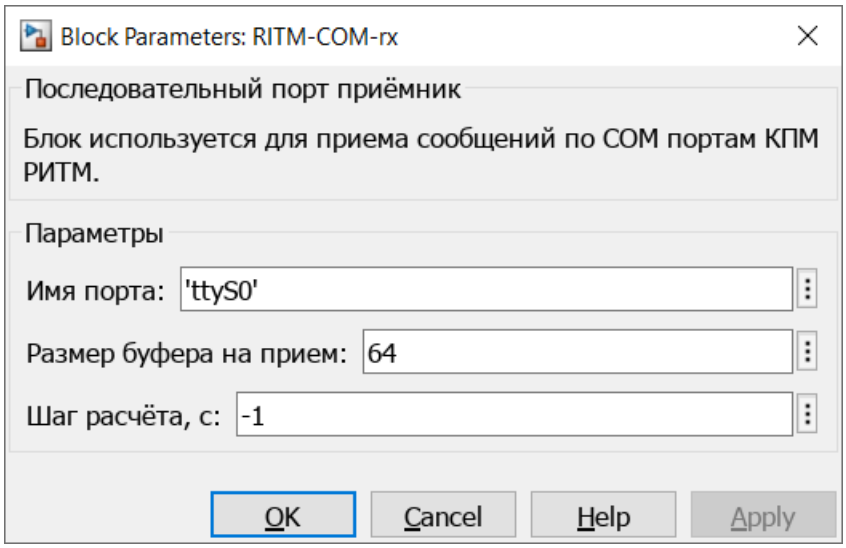
Параметры блока

- **Имя порта**: Задает имя порта для передачи.
- **Размер буфера на передачу**: Задает максимальный размер передаваемого массива.
- ***Шаг расчёта.*** Позволяет задать шаг расчета в секундах для передачи данных из приложение реального времени.

Блок Приёмник

Блок используется для приёма сообщений по по COM портам КПМ РИТМ.

Маска блока



Выходы блока

- ***Размер*** - количество принятых байт. Тип данных - uint32.
- ***Данные*** - принятые данные. Массив 8-битных беззнаковых элементов (uint8).

Параметры блока

- **Имя порта**: Задает имя порта для передачи.
- **Размер буфера на прием**: Задает максимальный размер принимаемого массива.
- ***Шаг расчёта.*** Позволяет задать шаг расчета в секундах для приема данных в приложение реального времени.

Примеры и демонстрации RITM-COM

Пример использования блоков RITM-COM.

```
open_system('ritmdemo_block_com.slx')
```

Блоки поддержки Ethernet

Содержание

- Библиотека UDP
- Библиотека сырого Ethernet
- Блоки и API для работы с PTP на РИТМ

Библиотеки для работы с протоколами на базе Ethernet в КПМ “РИТМ”

Библиотека UDP

Обеспечивает поддержку UDP для Ethernet интерфейсов КПМ “РИТМ”.

Библиотека сырого Ethernet

Библиотека блоков для отправки и приёма сырых пакетов по Ethernet.

Блоки и API для работы с PTP на РИТМ

Библиотека блоков программный интерфейс для работы с PTP на КПМ “РИТМ”.

Библиотека блоков RITM-UDP

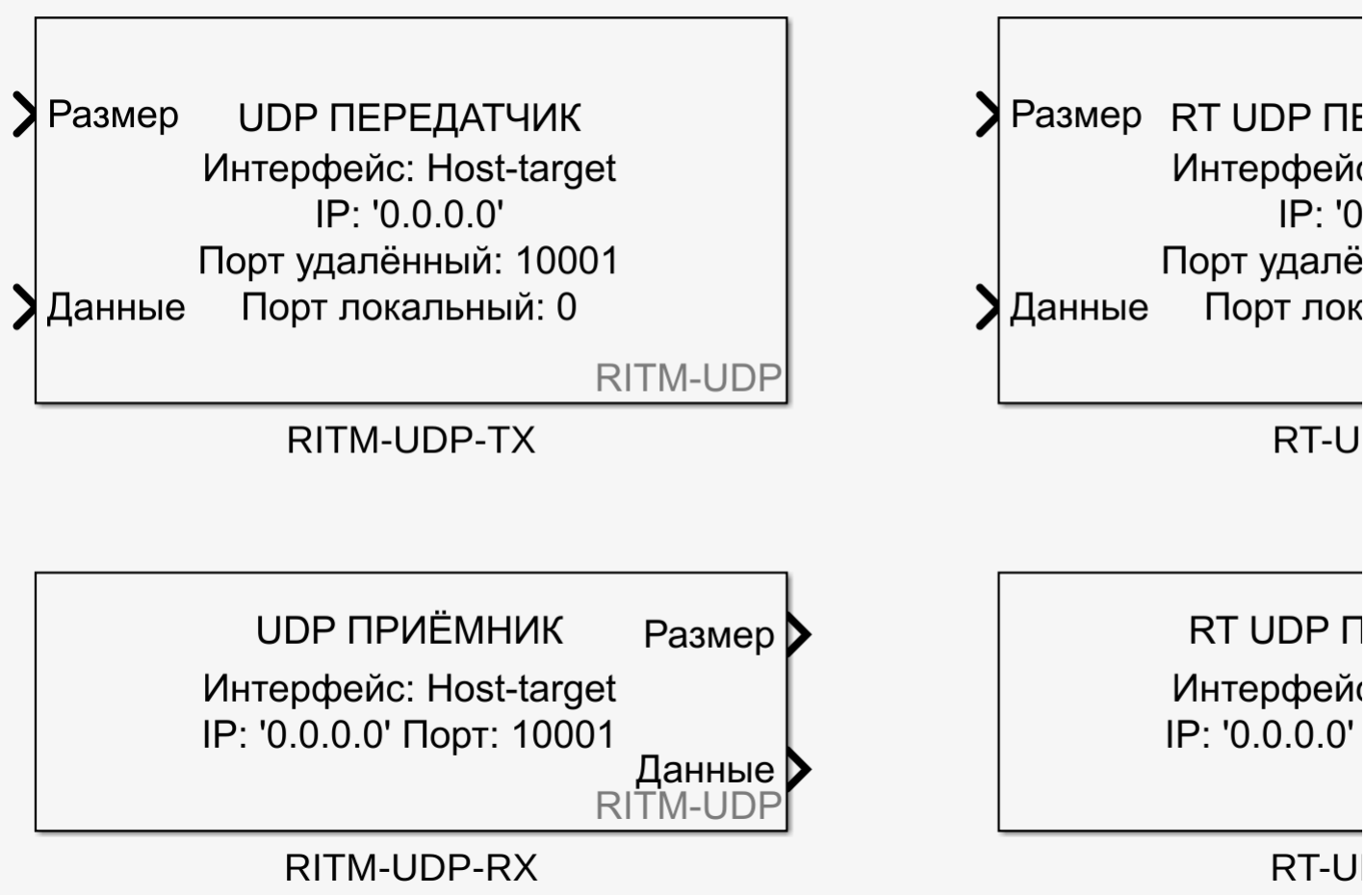
Содержание

- Описание
- Библиотека блоков
- Блок UDP-Передатчик
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок UDP-Приёмник
  - Маска блока
  - Выходы блока
  - Параметры блока

Описание

Библиотека блоков для отправки и приёма UDP пакетов по Ethernet. Блоки с префиксом RT не отличаются по своим интерфейсам, за исключением того, что для них нет возможности использовать интерфейс HostTarget. Они работают стабильней (с точки зрения времени выполнения шага, TET) и производительней, чем обычные. Перед использованием необходимо произвести настройку IP адресов Ethernet портов КПМ “РИТМ”, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

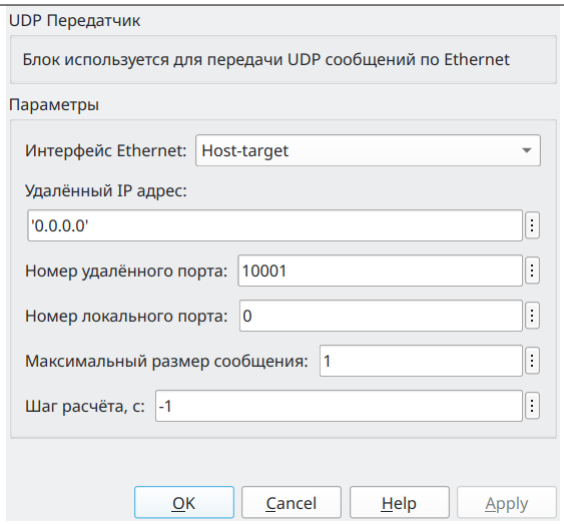
Библиотека блоков



Блок UDP-Передатчик

Блок используется для передачи UDP сообщений по Ethernet.

Маска блока



Входы блока

- **Размер**: Число байт для передачи из вектора **“Данные”**, тип данных *uint32*.
- **Данные**: Массив байт для передачи.

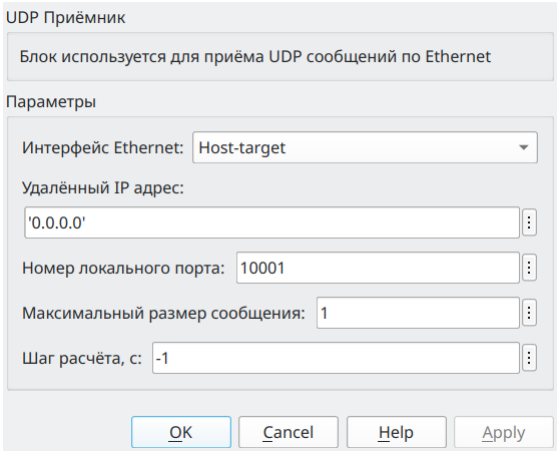
Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ “РИТМ”.
- **Удалённый IP адрес**: IPv4 адрес получателя сообщений. *0.0.0.0* это IP адрес для отправки сообщений на локальную машину.
- **Номер удалённого порта**: Порт удаленного получателя, на который будет происходить отправка сообщений.
- **Номер локального порта**: Локальный порт UDP, с которого будет производиться отправка сообщений. При заданном в маске нулевом значении номер порта будет определен операционной системой самостоятельно из доступных.
- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины вектора **“Данные”**.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для отправки сообщений UDP из приложения реального времени.

Блок UDP-Приёмник

Блок используется для приёма UDP сообщений по Ethernet.

Маска блока



Выходы блока

- **Размер**: Число принятых байт, тип данных *uint32*. Не более значения параметра **“Максимальный размер сообщения”**. Если в буфере операционной системы находится больше данных, они будут считаны при следующем шаге расчета.
- **Данные**: Массив принятых байт.

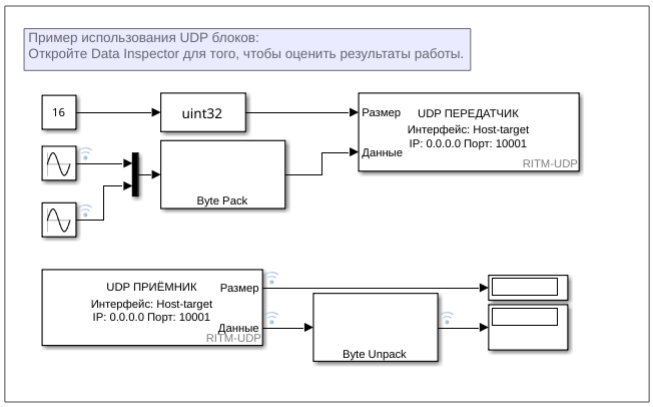
Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для получения сообщений. Номера портов указаны на корпусе КПМ “РИТМ”.
- **Удалённый IP адрес**: IPv4 адрес отправителя сообщений. *0.0.0.0* это IP адрес для приема сообщений от любого отправителя.
- **Номер локального порта**: Порт, который будет прослушиваться приёмником.
- **Максимальный размер сообщения**: Максимально возможная длина принятого сообщения.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для приема сообщений UDP в приложение реального времени.

Примеры и демонстрации RITM-UDP

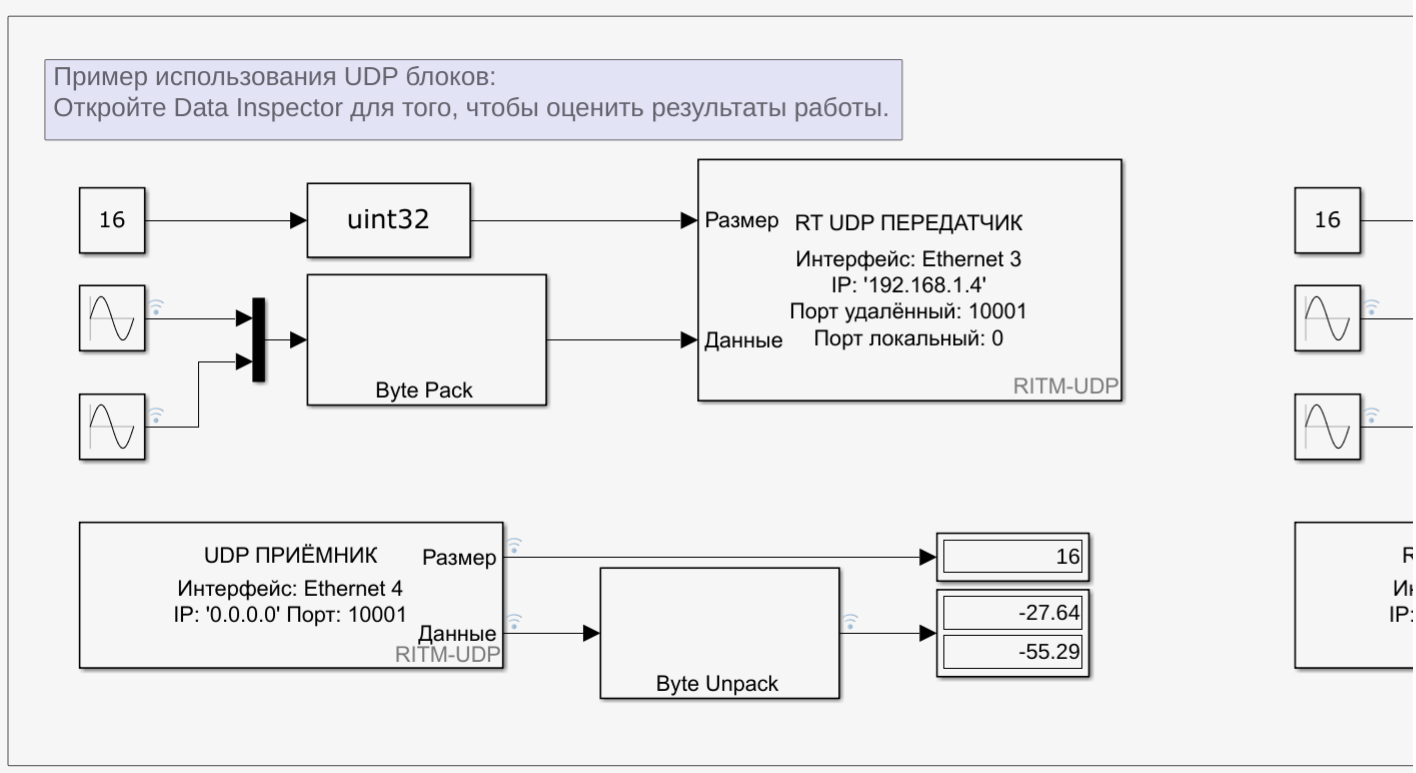
Пример использования UDP блоков.

```
open_system('ritmdemo_block_udp.slx')
```



Пример использования RT UDP блоков.

```
open_system('ritmdemo_block_rtudp.slx')
```



Библиотека блоков RITM-TCP

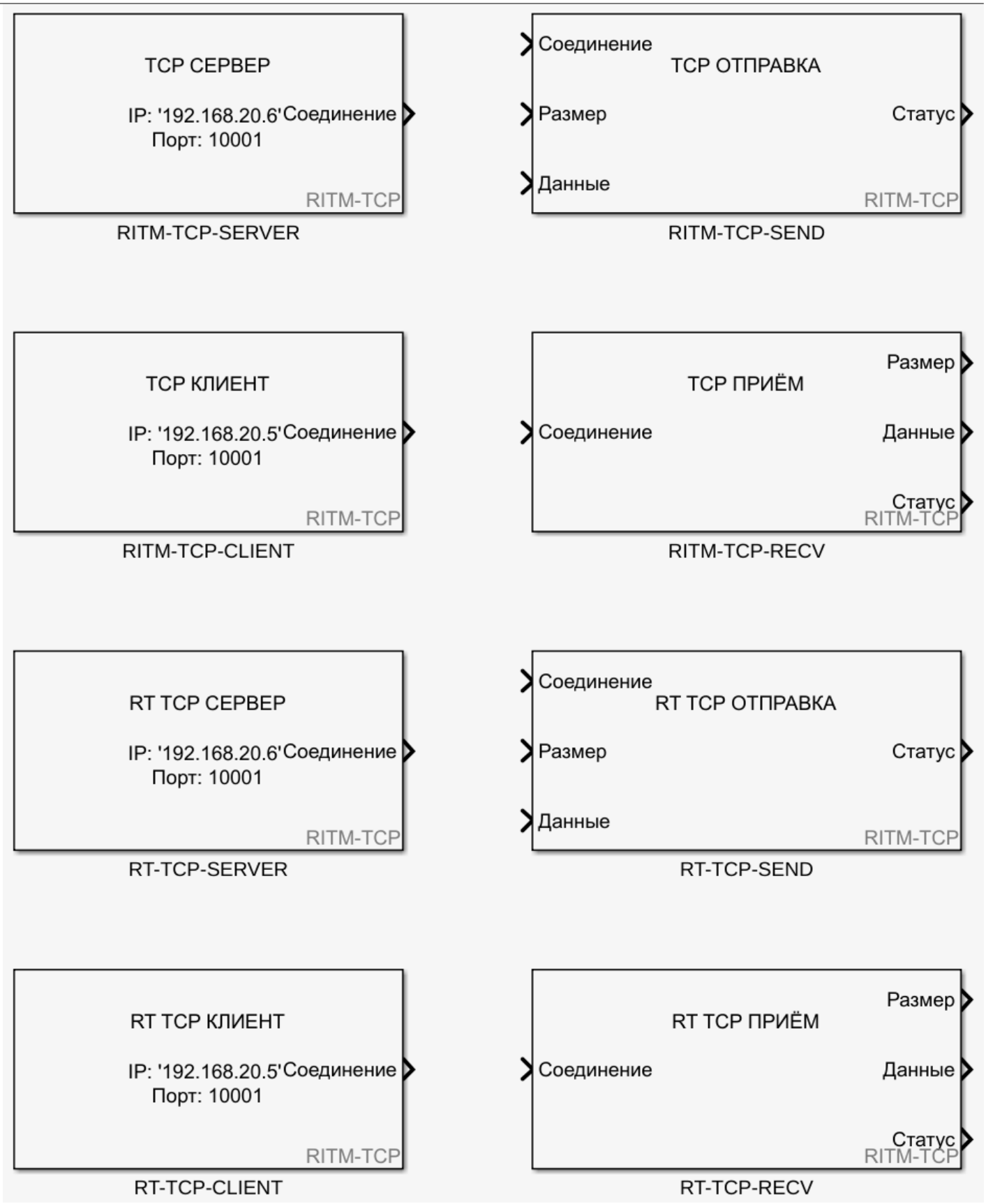
Содержание

- Описание
- Библиотека блоков
- Блок TCP Сервер
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок TCP Клиент
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок TCP Отправка
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок TCP Приём
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока

Описание

Библиотека блоков для использования TCP.
Блоки с префиксом RT не отличаются по своим интерфейсам. Они работают стабильней (с точки зрения времени выполнения шага, TET) и производительней, чем обычные блоки TCP. Перед использованием необходимо произвести настройку IP адресов для сетевых портов КПМ “РИТМ”, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

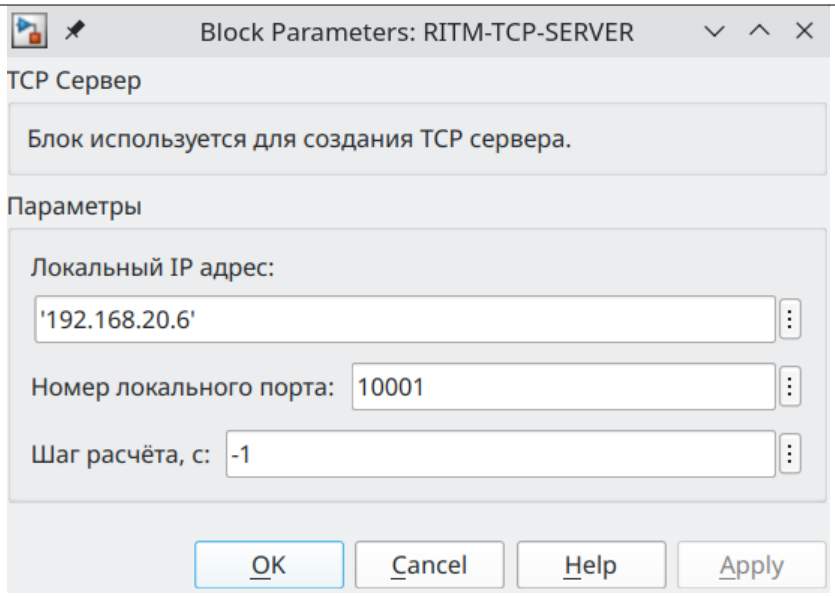
Библиотека блоков



Блок TCP Сервер

Блок используется для создания TCP сервера.

Маска блока



Выходы блока

- **Соединение**: Информация о соединении необходимая для блоков “TCP Приём” и “TCP Отправка”.

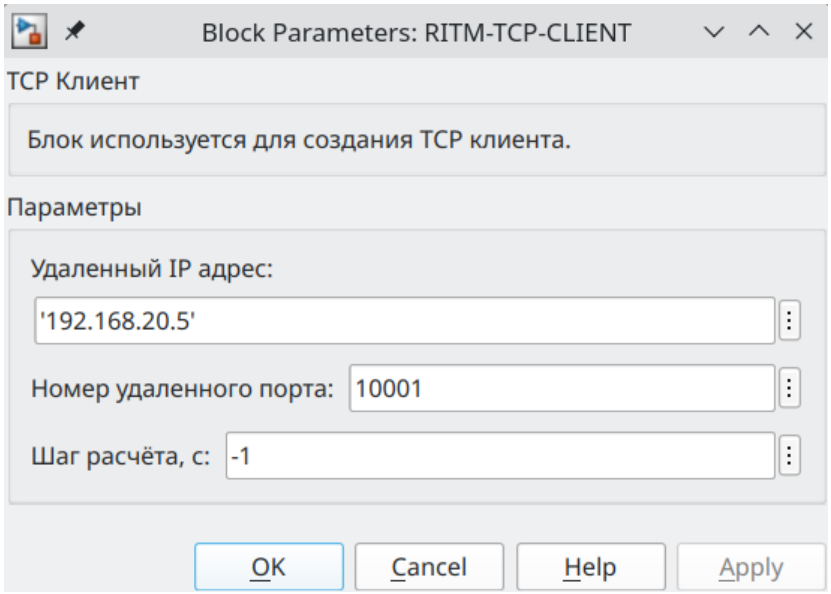
Параметры блока

- **IP адрес**: IPv4 адрес заданный устройству с поддержкой, который будет прослушиваться сервером.
- **Номер локального порта**: Порт, который будет прослушиваться сервером.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для обновления информации о состоянии соединения.

Блок TCP Клиент

Блок используется для

Маска блока



Выходы блока

- **Соединение**: Информация о соединении необходимая для блоков “TCP Приём” и “TCP

Передача”.

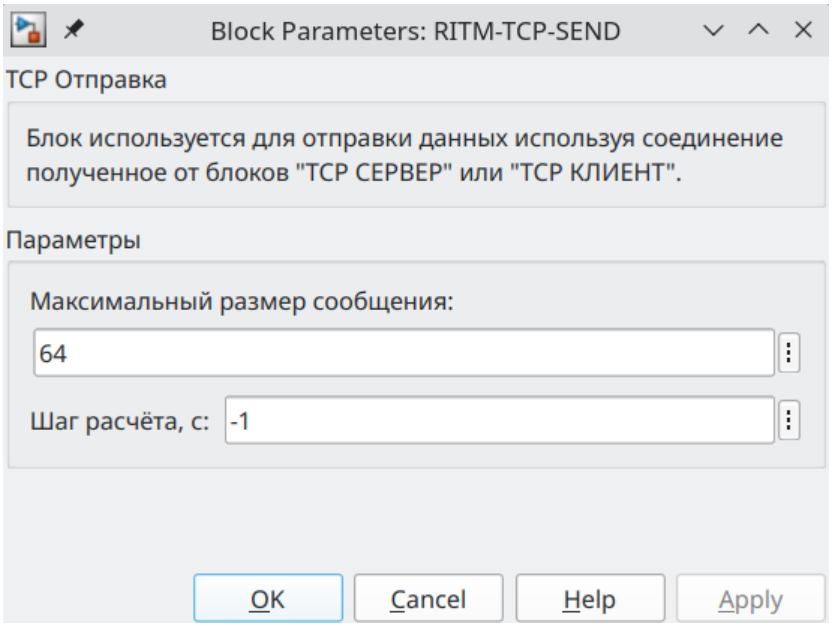
Параметры блока

- **IP адрес**: IPv4 адрес заданный устройству с поддержкой, к которому будет производиться подключение.
- **Номер локального порта**: Порт, к которому будет производиться подключение.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для обновления информации о состоянии соединения.

Блок TCP Отправка

Блок используется для передачи данных, используя соединение полученное от блоков “TCP Сервер” или “TCP Клиент”.

Маска блока



Входы блока

- **Соединение**: Информаци о соединении полученная от блока “TCP Сервер” или “TCP Клиент”.
- **Размер**: Число байт для передачи из вектора **“Данные”**, тип данных *uint32*.
- **Данные**: Массив байт для передачи.

Выходы блока

- **Статус**: Информация о статусе завершения передачи - ненулевой статус сообщает об ошибке.

Параметры блока

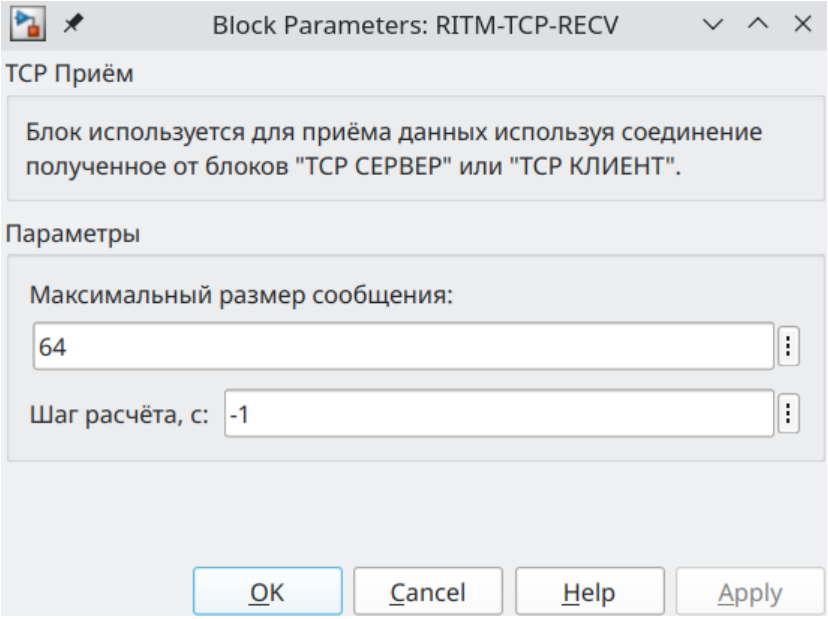
- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины вектора **“Данные”**.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для отправки сообщений из приложения реального времени.

Блок TCP Приём

Блок используется для приёма данных, используя соединение полученное от блоков “TCP

Сервер” или “TCP Клиент”.

Маска блока



Входы блока

- **Соедиение**: Информаци о соединении полученная от блока “TCP Сервер” или “TCP Клиент”.

Выходы блока

- **Статус**: Информация о статусе завершения передачи - ненулевой статус сообщает об ошибке.
- **Размер**: Число принятых байт, тип данных *uint32*. Не более значения параметра **“Максимальный размер сообщения”**. Если в буфере операционной системы находится больше данных, они будут считаны при следующем шаге расчета.
- **Данные**: Массив принятых байт.

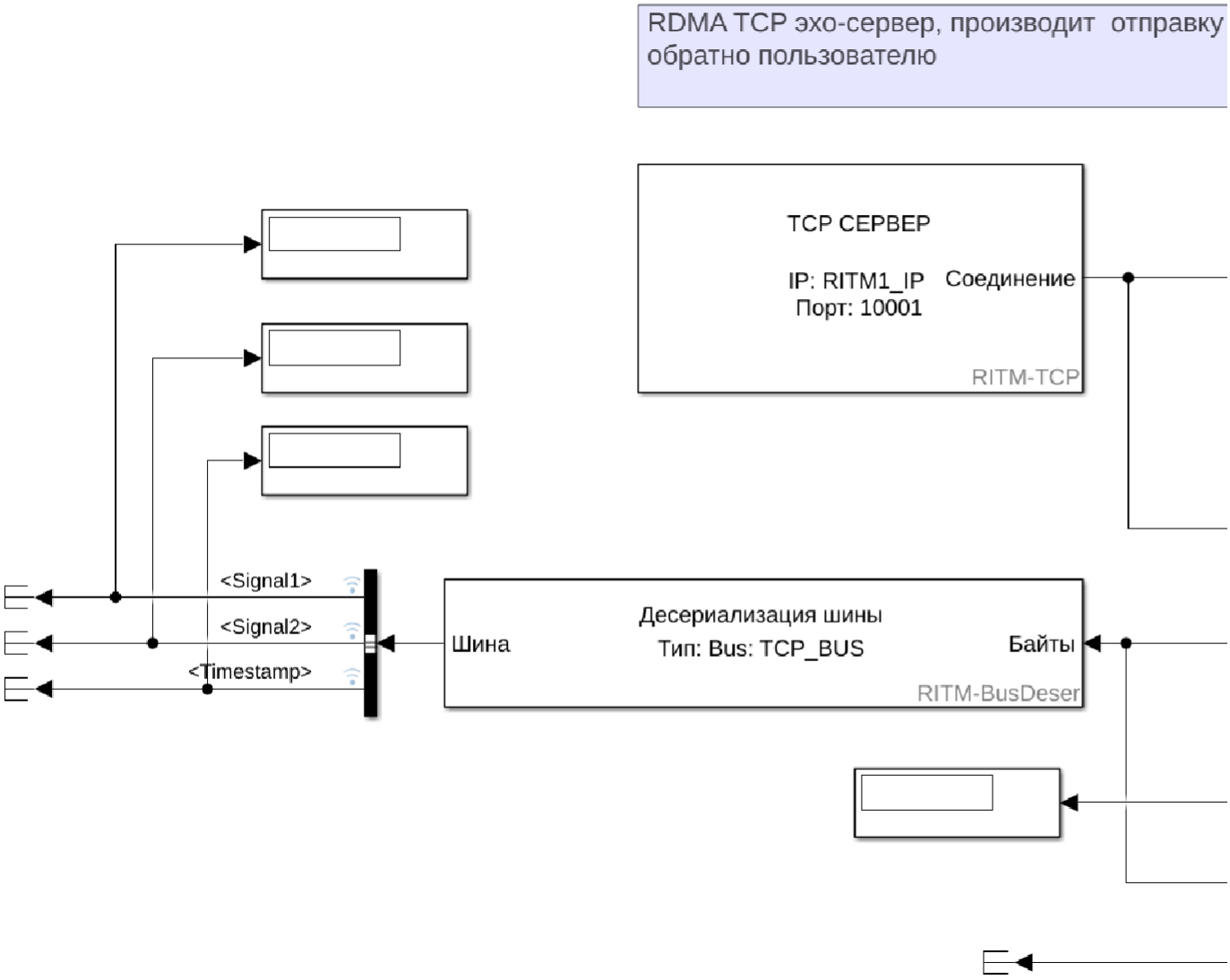
Параметры блока

- **Максимальный размер сообщения**: Максимально возможная длина принятого сообщения.
- **Максимальное время ожидания сообщения, мкс**: Максимальное время ожидания сообщения, если число принятых данных меньше значения параметра **“Максимальный размер сообщения”**.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для приема сообщений в приложение реального времени.

Примеры и демонстрации RITM-TCP

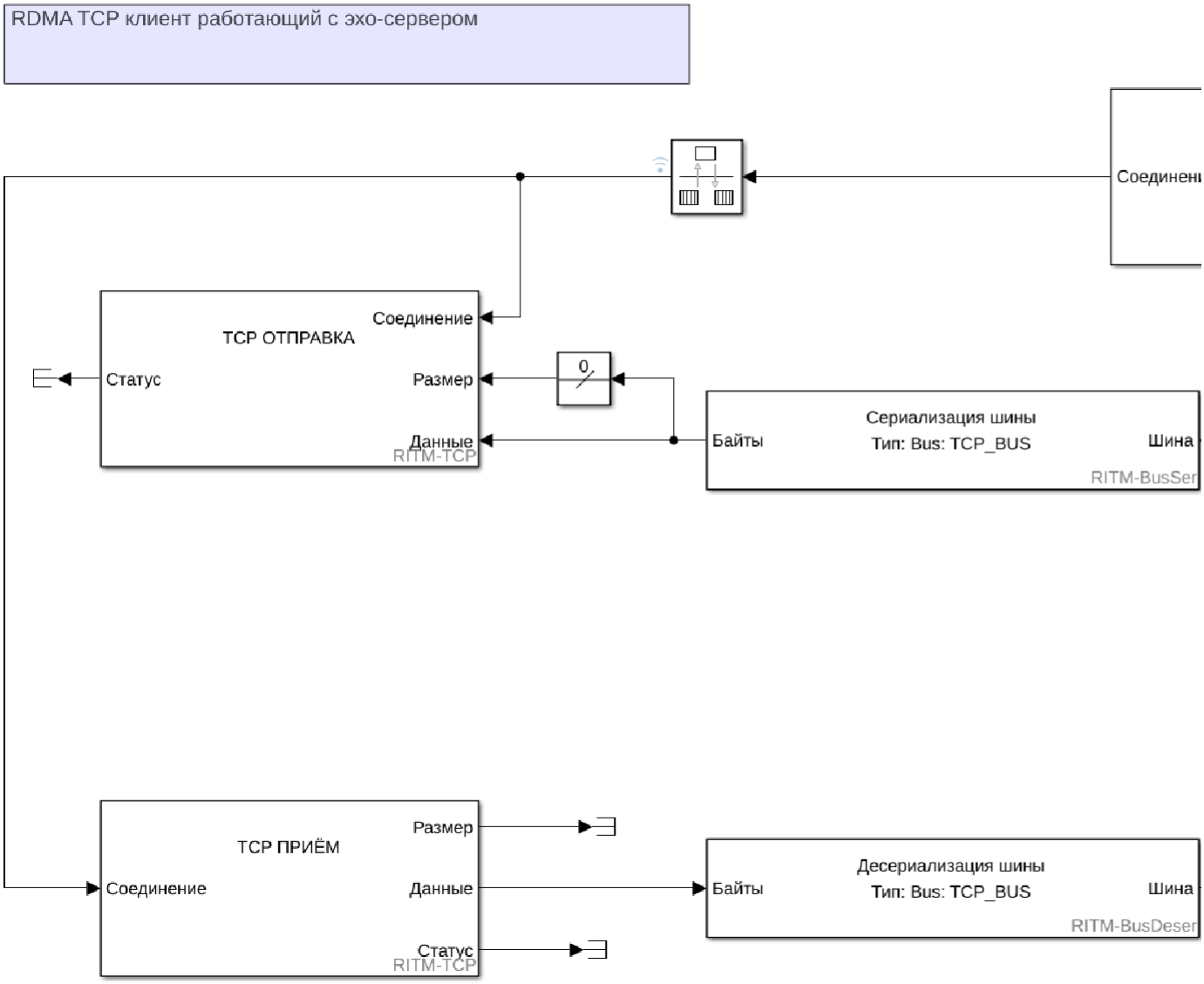
Пример сервера с использованием TCP блоков.

```
open_system('ritmdemo_block_tcp_server.slx')
```



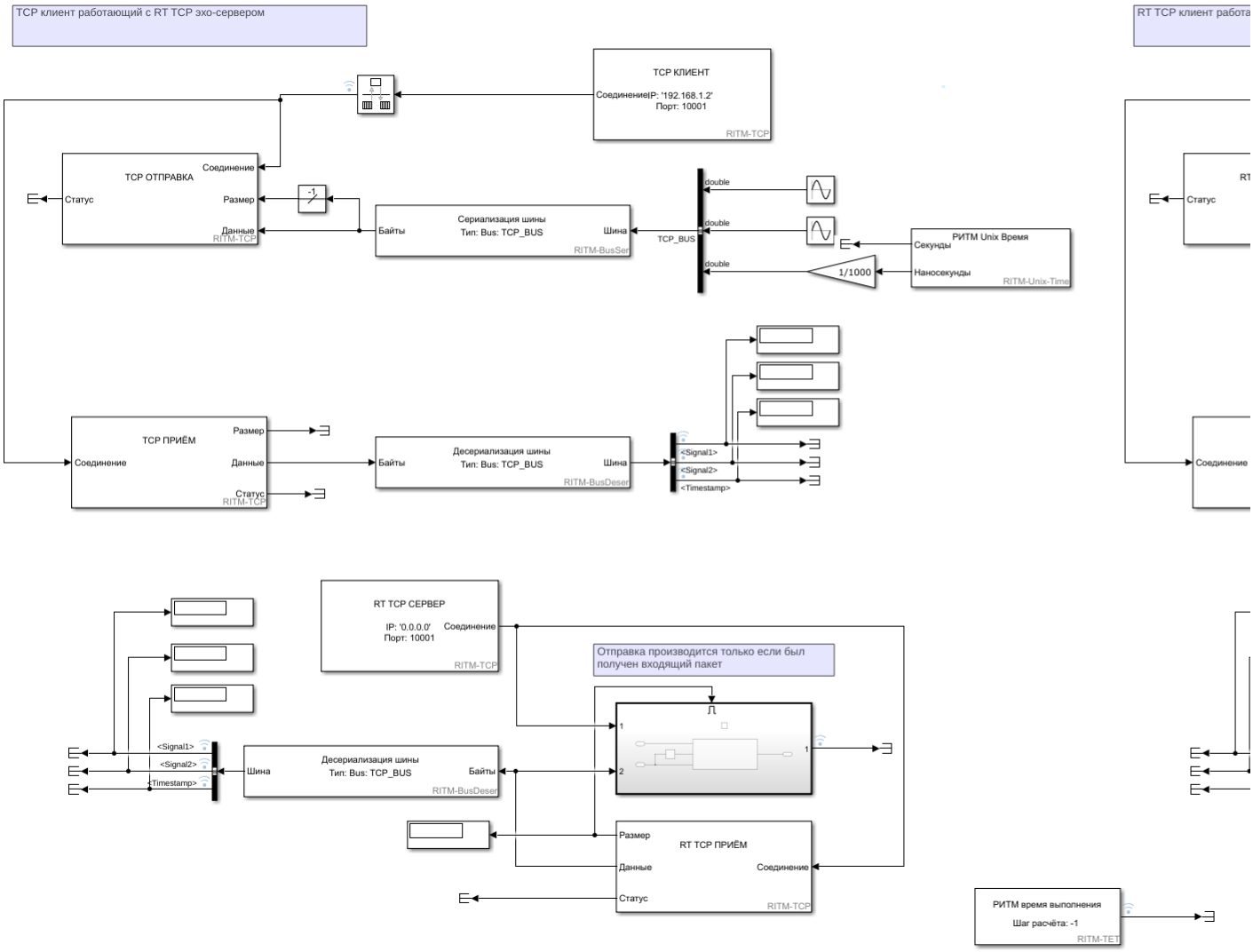
Пример клиента с использованием TCP блоков.

```
open_system('ritmdemo_block_tcp_client.slx')
```



Пример клиента с использованием RT TCP блоков.

```
open_system('ritmdemo_block_rttcp.slx')
```


TCP клиент работающий с RT TCP эхо-сервером
RT TCP клиент работа
TCP КЛИЕНТ
Соединение
IP: '192.168.1.2'
Порт: 10001
RITM-TCP
TCP ОТПРАВКА
Соединение
Статус
Размер
Данные
RITM-TCP
-1
Сериализация шины
Тип: Bus: TCP_BUS
Байты
Шина
RITM-BusSer
TCP_BUS
double
double
double
РИТМ Unix Время
Секунды
Наносекунды
1/1000
RITM-Unix-Time
ТСР ПРИЁМ
Соединение
Размер
Данные
Статус
RITM-TCP
Десериализация шины
Тип: Bus: TCP_BUS
Байты
Шина
RITM-BusDeser
<Signal1>
<Signal2>
<Timestamp>
RT TCP СЕРВЕР
IP: '0.0.0.0'
Порт: 10001
Соединение
RITM-TCP
Отправка производится только если был получен входящий пакет
RT TCP ПРИЁМ
Размер
Данные
Статус
Соединение
RITM-TCP
РИТМ время выполнения
Шаг расчёта: -1
RITM-TET

Библиотека блоков RITM Raw Ethernet

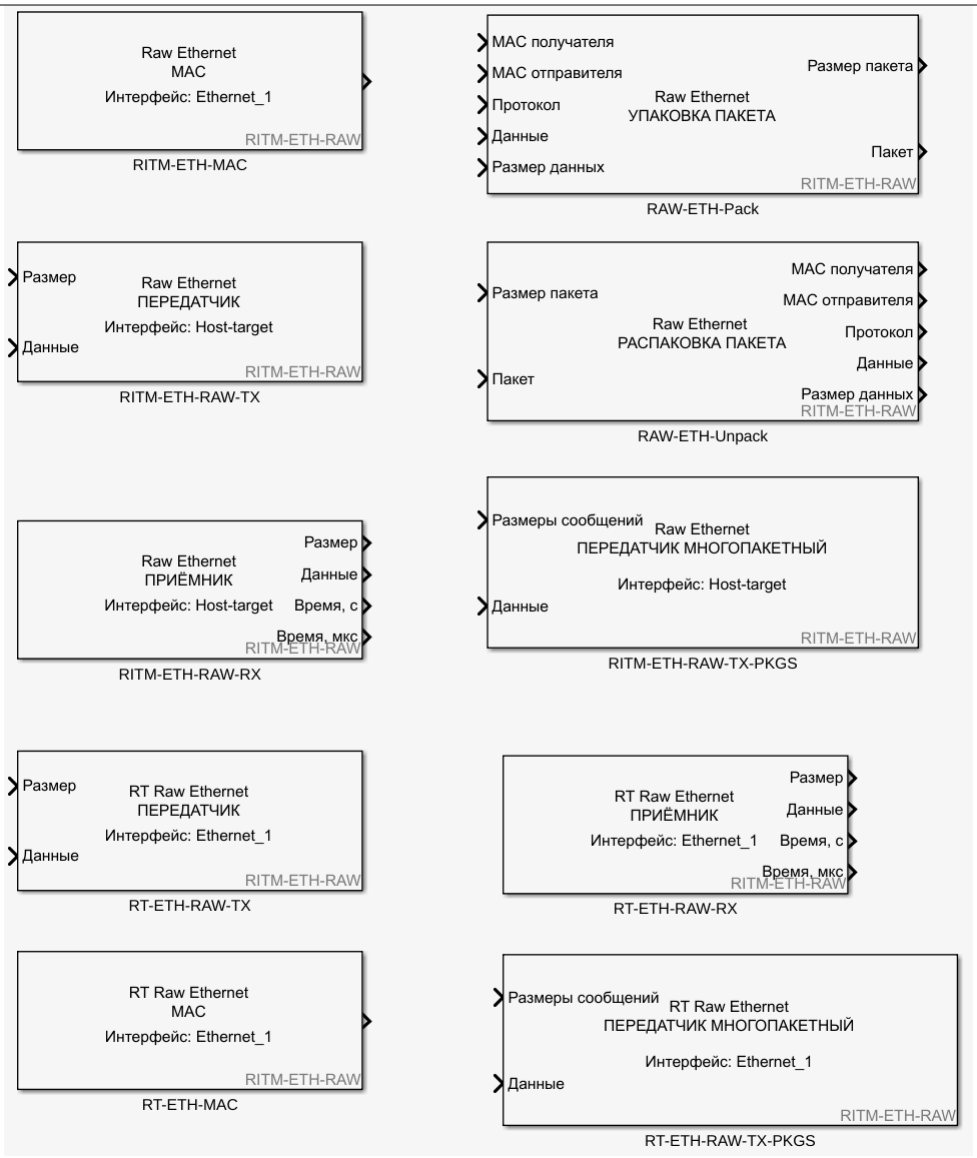
Содержание

- Описание
- Библиотека блоков
- Блок Raw Ethernet Передатчик
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок Raw Ethernet Передатчик Многопакетный
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок Raw Ethernet Приёмник
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок Raw Ethernet MAC
  - Выходы блока
  - Параметры блока
- Блок Raw Ethernet Упаковка пакета
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок Raw Ethernet Распаковка пакета
  - Входы блока
  - Выходы блока
  - Параметры блока
- Фильтр сообщений в блоке Raw Ethernet Приёмник

Описание

Библиотека блоков для отправки и приёма сырых пакетов по Ethernet. Блоки с префиксом RT не отличаются по своим интерфейсам, за исключением того, что для них нет возможности использовать интерфейс HostTarget. Они работают стабильней (с точки зрения времени выполнения шага, TET) и производительней, чем обычные. Блоки “Raw Ethernet УПАКОВКА ПАКЕТА” и “Raw Ethernet РАСПАКОВКА ПАКЕТА” могут универсально использоваться и для RT блоков и для обычных. Перед использованием необходимо произвести настройку Ethernet портов КПМ РИТМ, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

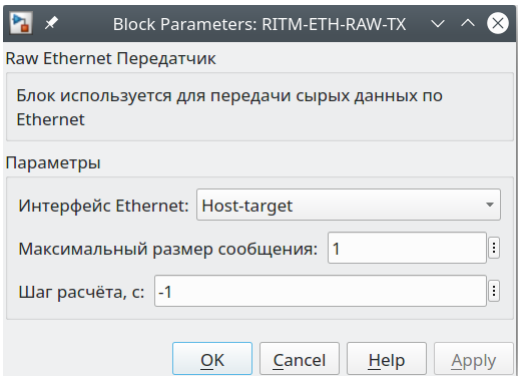
Библиотека блоков



Блок Raw Ethernet Передатчик

Блок используется для передачи сырых данных по Ethernet.

Маска блока



Входы блока

- **Размер**: Число байт для передачи из вектора **Данные**, тип данных *uint32*.
- **Данные**: Массив байт для передачи.

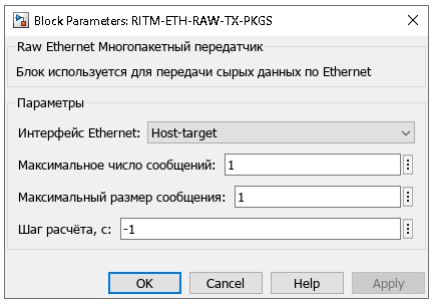
Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ РИТМ.
- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины вектора **Данные**.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для отправки пакетов.

Блок Raw Ethernet Передатчик Многопакетный

Блок используется для передачи массивов сырых данных по Ethernet.

Маска блока



Входы блока

- **Размеры сообщений**: Массив размеров сообщений для передачи.
- **Данные**: Массив векторов с данными сообщений.

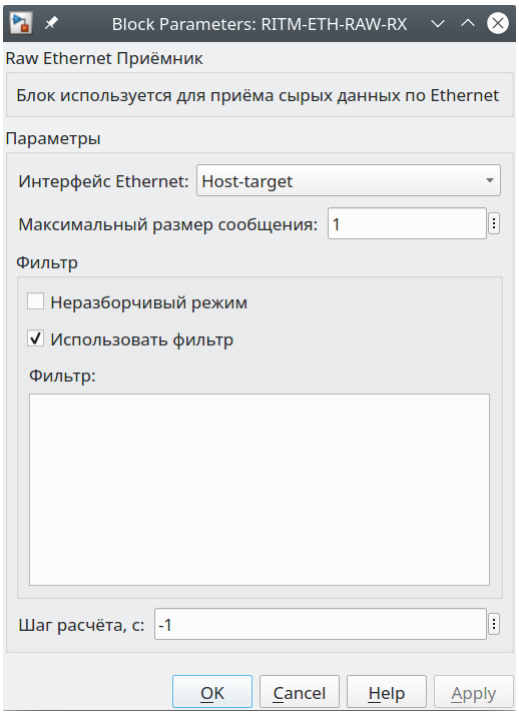
Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ РИТМ.
- **Максимальное число сообщений**: Длина векторов **Размер сообщений** и **Данные**.
- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины векторов **Данные**.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для отправки пакетов.

Блок Raw Ethernet Приёмник

Блок используется для приёма сырых данных по Ethernet. Возвращает данные последнего принятого пакета.

Маска блока



Выходы блока

- **Размер**: Число принятых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения**.
- **Данные**: Принятый массив байт.
- **Время, с**: Время в секундах с начала эпохи UNIX, в которое был принят пакет.
- **Время, мкс**: Дробная часть времени. Число микросекунд, прошедшее с начала секунды.

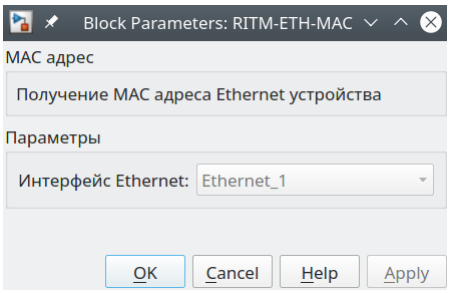
Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для получения сообщений. Номера портов указаны на корпусе КПМ РИТМ.
- **Максимальный размер сообщения**: Максимально возможная длина принятого сообщения.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для приема Ethernet пакетов. За каждый шаг расчёта из буфера операционной системы считывается один пакет. Если в буфере операционной системы находится больше одного пакета, то на следующем шаге расчёта будет считан один следующий пакет.
- **Фильтр**: Меню для настройки фильтров принимаемых пакетов.

- **Неразборчивый режим**: Позволяет переключить порт в неразборчивый режим, в котором сетевая плата принимает все пакеты независимо от того, кому они адресованы.
- **Использовать фильтр**: Флаг, позволяющий включить фильтр принимаемых сообщений. Подробнее о работе фильтра написано в разделе Фильтр сообщений в блоке Raw Ethernet Приёмник.

Блок Raw Ethernet MAC

Блок используется для получения MAC адреса выбранного Ethernet интерфейса.



Выходы блока

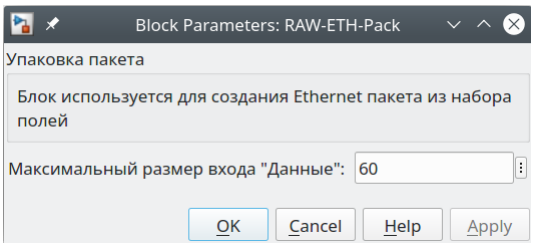
- **MAC**: Массив из 6 байт - MAC адрес выбранного интерфейса.

Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, MAC адрес которого требуется получить.

Блок Raw Ethernet Упаковка пакета

Блок используется для создания Ethernet пакета из набора полей.



Входы блока

- **MAC назначения**: MAC адрес устройства назначения.
- **MAC отправителя**: MAC адрес, который будет указан как MAC отправителя.
- **Протокол**: 2 байта, метка протокола, которую будет содержать пакет.
- **Данные**: Массив, содержащий данные, которые будут помещены в пакет.
- **Размер данных**: Размер сегмента данных. Число, которое будет использоваться для определения передаваемого размера пакета.

Выходы блока

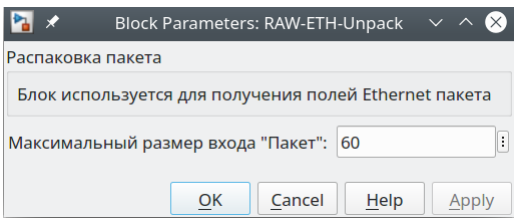
- **Размер пакета**: Число байт, которое необходимо передать из массива **Пакет**.
- **Пакет**: Массив из последовательных полей **MAC назначения**,**MAC отправителя**, метки **Протокол** и массива **Данные**.

Параметры блока

- **Максимальный размер входа “Данные”**: Наибольший размер массива данных, который может быть подан на вход блока.

Блок Raw Ethernet Распаковка пакета

Блок используется для получения полей Ethernet пакета.



Входы блока

- **Размер пакета**: Размер принятого пакета.
- **Пакет**: Массив байт принятого пакета.

Выходы блока

- **MAC назначения**: MAC адрес устройства назначения.
- **MAC отправителя**: MAC адрес устройства отправителя.

- **Протокол**: Метка протокола, которую содержит пакет.
- **Данные**: Сегмент данных пакета.
- **Размер данных**: Размер сегмента данных.

Параметры блока

- **Максимальный размер входа “Пакет”**: Наибольший размер пакета, который может быть подан на вход блока.

Фильтр сообщений в блоке Raw Ethernet Приёмник

Для приёма пакетов используется библиотека **PCAP**. Окно **Фильтр** блока **Raw Ethernet Приёмник** используется для ввода выражения, которое будет использоваться для компиляции фильтров библиотеки.

Фильтр **PCAP** представляет собой строку, состоящую из набора примитивов. Набор примитивов определяет то, какие пакеты будут приняты. Примитив содержит **идентификатор** и один или несколько **квалификаторов**.

Определено три типа квалификаторов:

- **type**: Квалификатор определяет тип имени или номера идентификатора.

Возможные значения: **host** (хост), **net** (сеть), **port** (порт) или **portrange** (диапазон портов). При отсутствии квалификатора, по умолчанию принимается значение **host**.

- **dir**: Квалификатор определяет направление передачи информации относительно объекта определенного идентификатором.

Возможные значения: **src** (отправитель), **dst** (получатель), **src and dst** (отправитель и получатель), **src or dst** (отправитель или получатель). Значение по умолчанию - **src or dst**.

- **proto**: Квалификатор определяет соответствие определенному протоколу.

Возможные значения: **ether**, **fddi**, **ip**, **arp**, **rarp**, **decnet**, **lat**, **sca**, **moprc**, **mopdl**, **tcp** и **udp**.

Например:

ether src 0A:0A:0A:0A:0A:0A - захват сообщений от отправителя с MAC: **0A:0A:0A:0A:0A:0A**.

dst net 192.168.56.0/24 - захват сообщений предназначенных для сети **192.168.56.0/24**.

host 192.168.56.3 - захват сообщений предназначенных либо отправленных устройством с адресом **192.168.56.3**.

ip icmp – захват ICMP пакетов.

Кроме идентификаторов и классификаторов объекта фильтры могут содержать ключевые слова **gateway**, **broadcast**, **multicast**, **less**, **greater**, а так же арифметические выражения.

Например:

ip multicast – захват ip пакетов, содержащих адреса из класса D.
less 1000 – захват кадров, у которых размер менее 1000 байт.

Примитивы можно компоновать используя логические операции:

- И - **and (&)**
- НЕ - **not (!)**
- ИЛИ - **or (||)**.

```
host foo and not port ftp and not port ftp-data
```

Условия могут быть объединены при помощи скобок

```
not (host ip1 or ip2)
```

Дополнительные примеры выражений

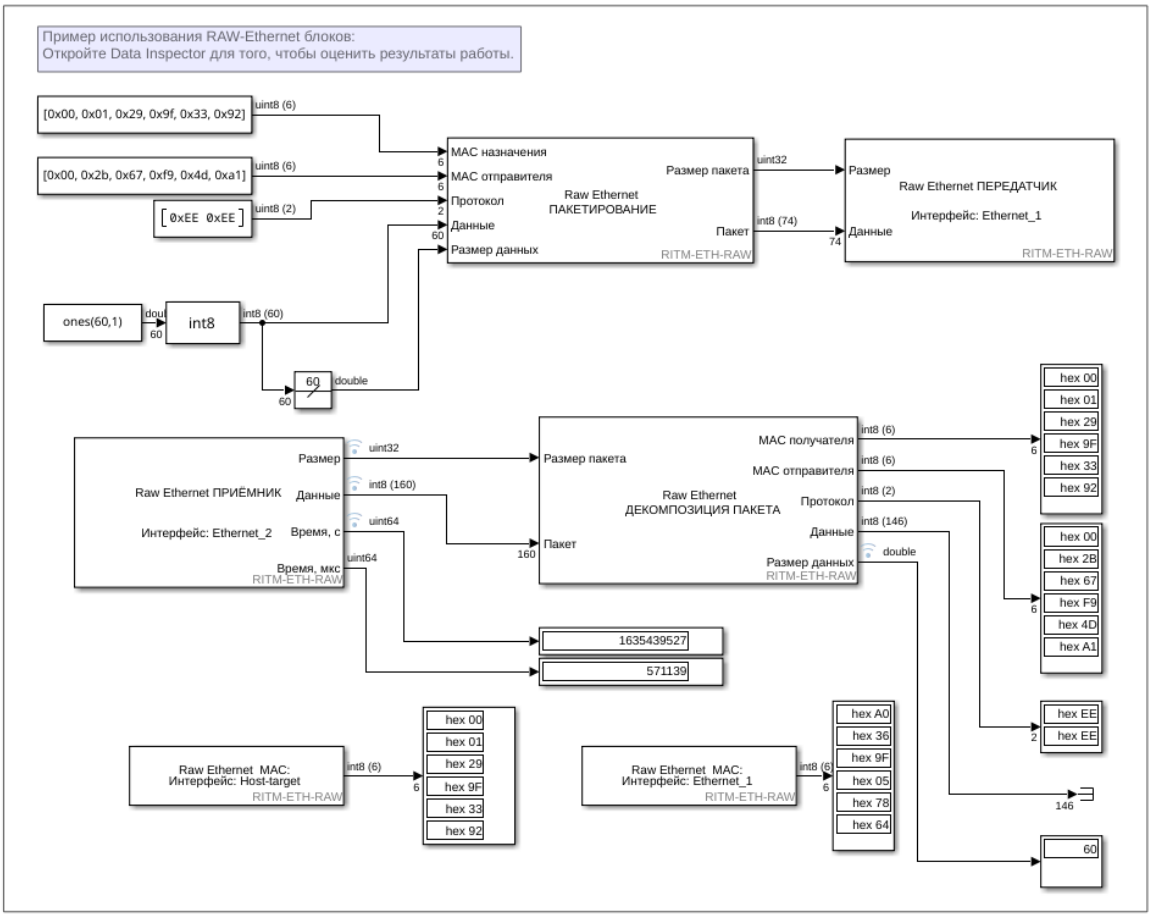
```
not (host 192.168.56.3 or 192.168.56.2) - Для отбрасывания пакетов, принадлежащих устройствам с ip  адресами 192.168.56.3 или 192.168.56.2.

host 192.168.0.1 or host 192.168.0.2 and tcp port 22 – Для захвата или любого трафика, принадлежащего хосту 192.168.0.1, или трафика протокол

(host 192.168.0.1 or host 192.168.0.2) and tcp port 22 – Для захвата трафика протокола TCP и использующего порт 22, принадлежащего хосту 192.

ip host 192.168.56.2 and not 192.168.56.3 - Захват всех пакетов, идущих от или к 192.168.56.2, кроме пакетов от или к 192.168.56.3
```

Более подробную информацию об использовании фильтров можно получить из официальной документации PCAP.

Примеры и демонстрации RITM-Raw-Ethernet

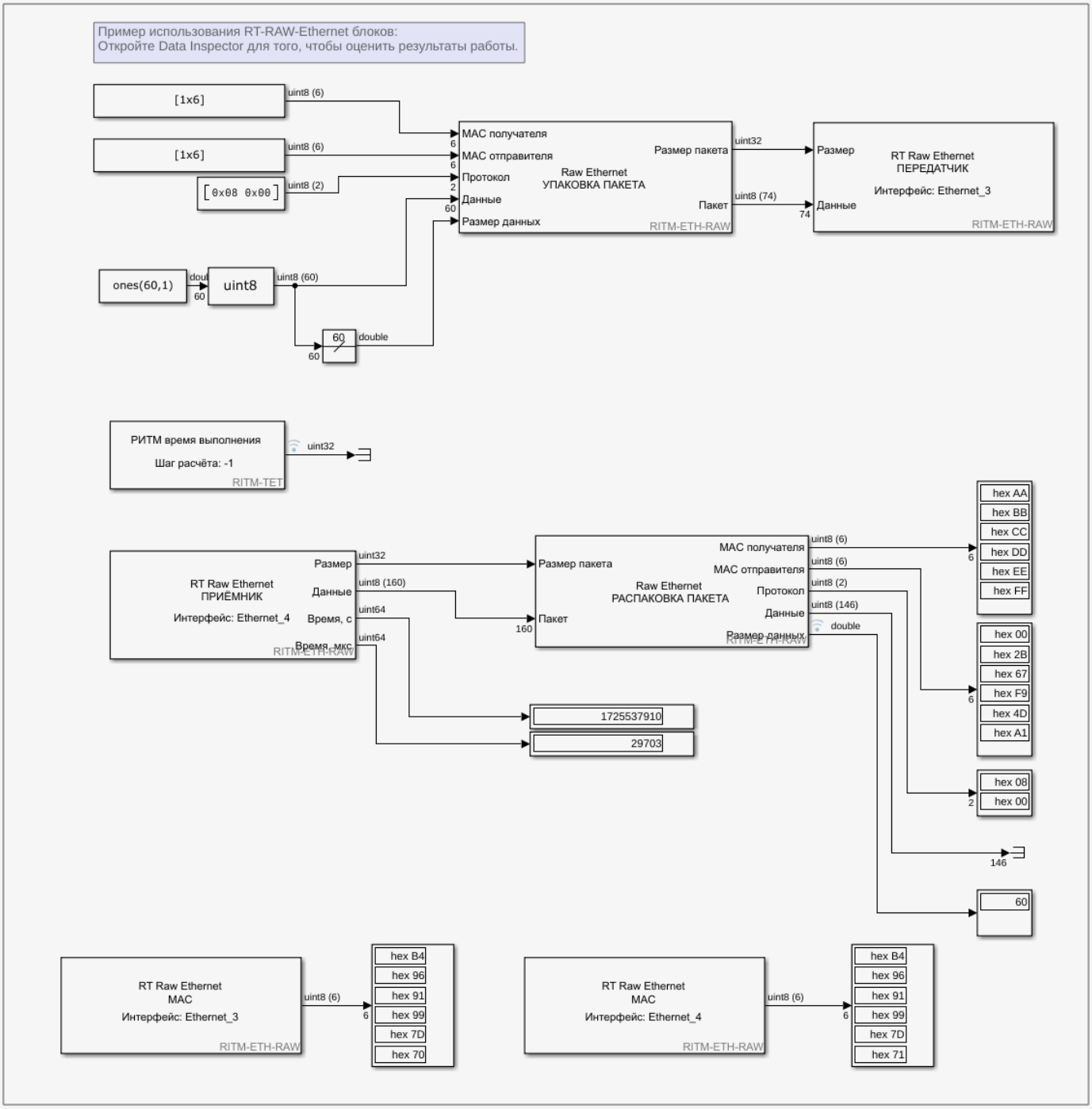
Пример использования блоков Raw Ethernet.

```
open_system('ritmdemo_block_eth_raw.slx')
```



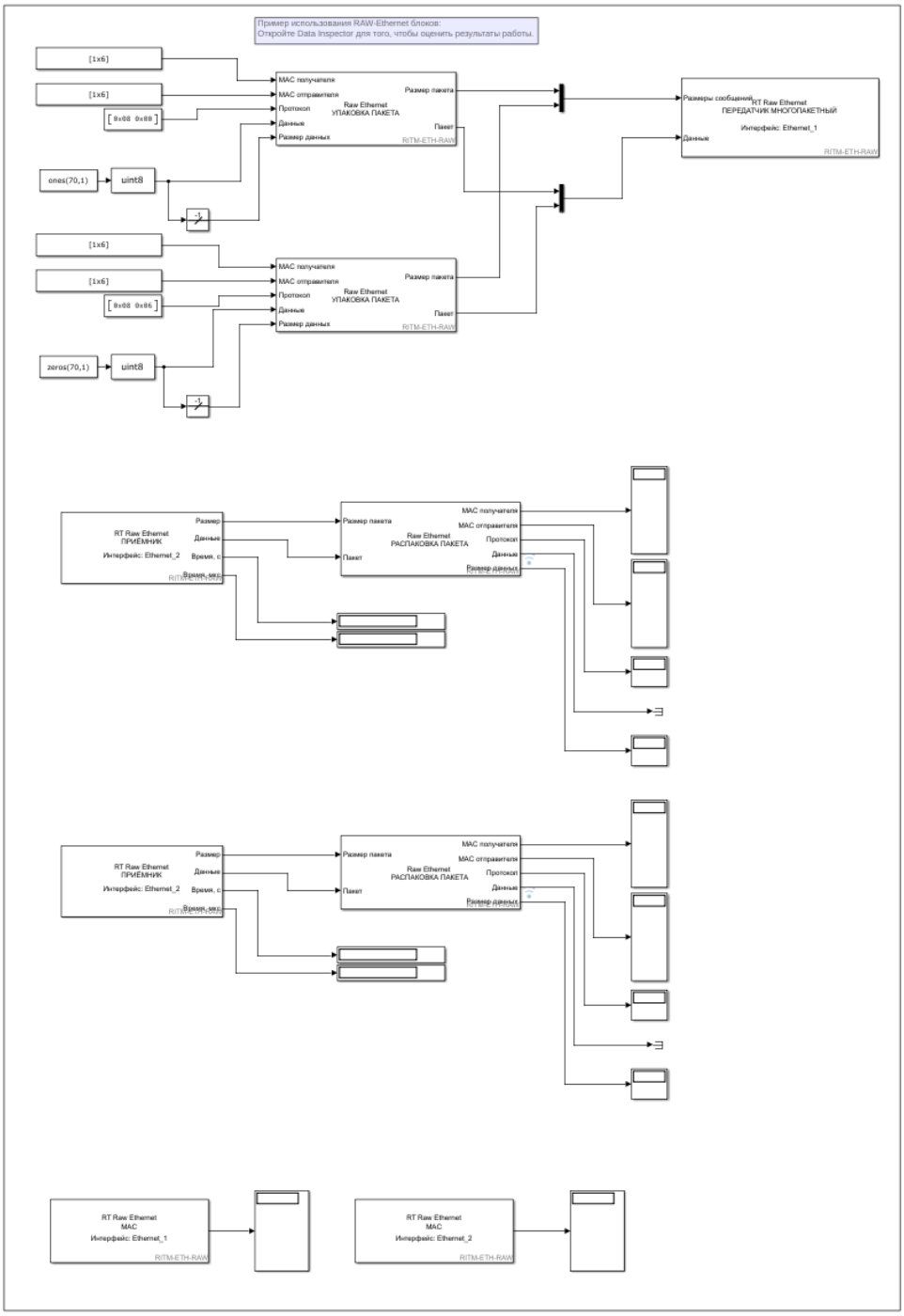
Пример использования блоков RT Raw Ethernet.

```
open_system('ritmdemo_block_rt_eth_raw.slx')
```



Пример использования многопакетного передатчика RT Raw Ethernet.

```
open_system('ritmdemo_block_rt_eth_raw_pkgs.slx')
```


Пример использования RAW-Ethernet блоков:
Откройте Data Inspector для того, чтобы оценить результаты работы.
[1x6]
[1x6]
[0x08 0x00]
ones(70,1)
uint8
MAC получателя
MAC отправителя
Протокол
Данные
Размер данных
Raw Ethernet
УПАКОВКА ПАКЕТА
Размер пакета
Пакет
RITM-ETH-RAW
Размеры сообщений
RT Raw Ethernet
ПЕРЕДАТЧИК МНОГОПАКЕТНЫЙ
Интерфейс: Ethernet_1
[0x08 0x86]
zeros(70,1)
RT Raw Ethernet
ПРИЁМНИК
Интерфейс: Ethernet_2
Размер
Время, с
Время, мкс
Raw Ethernet
РАСПАКОВКА ПАКЕТА
RT Raw Ethernet
MAC
Интерфейс: Ethernet_1
Интерфейс: Ethernet_2

Блоки и API для работы с PTP на РИТМ

В данном разделе приводится информация о программном интерфейсе предоставляемом пакетом поддержки КПМ “РИТМ” для настройки PTP соединения, а так-же блоках Simulink для получения информации о текущем состоянии работы PTP в процессе моделирования.

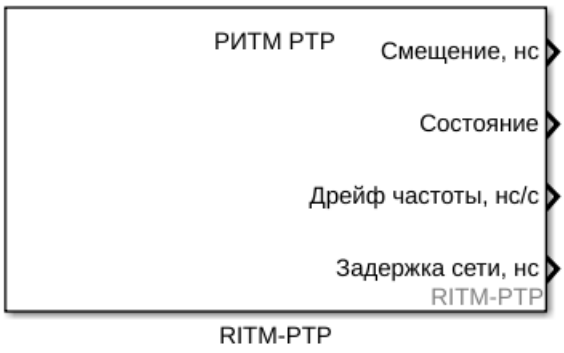
- **Библиотека блоков РИТМ PTP**: Блоки для получения информации о PTP соединении.
- **ritm.util.Machine.PTP**: API для настройки PTP соединения.

Библиотека блоков РИТМ PTP

Содержание

- Библиотека блоков
- Блок РИТМ PTP
  - Маска блока
  - Параметры блока
  - Выходы блока

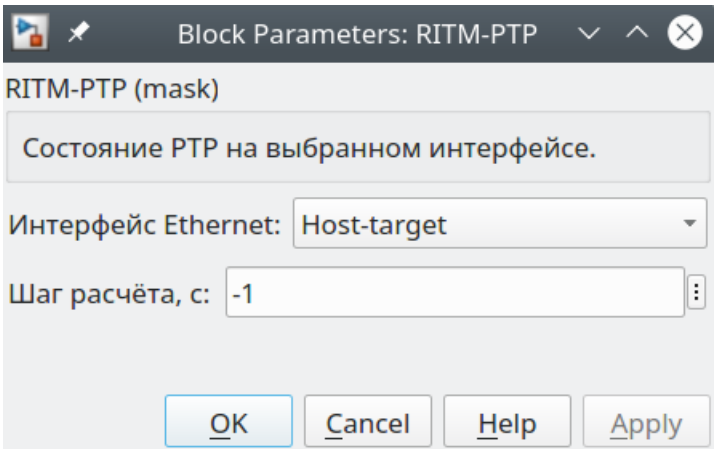
Библиотека блоков



Блок РИТМ PTP

Блок используется для получения состояния PTP на выбранном интерфейсе.

Маска блока



Параметры блока

- **Интерфейс Ethernet**: Порт Ethernet, который будет использоваться для получения сообщений. Номера портов указаны на корпусе КПМ РИТМ.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Выходы блока

- **Смещение**: Смещения часов Slave относительно Master в наносекундах. Для Master всегда равен “0”.Тип *int64*
- **Состояние**: Код текущего состояния PTP. Тип *int32*. Расшифровка кодов:

    - **0**: Slave обнаружил мастера на линии.
    - **1**: Смещение часов Slave.
    - **2**: Часы Slave синхронизированы с Master.
    - **3**: Master работает.
    - **4**: PTP соединение в процессе инициализации.
    - **5**: ошибка работы PTP.

- **Дрейф частоты**: Смещение часов Slave относительно часов Master в наносекундах за секунду. Тип *int32*
- **Задержка сети**: Оценка задержки передачи пакетов по сети от Master к Slave в наносекундах. Тип *int32*

Набор функции для настройки PTP соединения

Содержание

- Функция PTP.setupEthPtp
- Функция PTP.startPtp
- Функция PTP.stopPtp
- Функция PTP.resetSettings
- Функция PTP.getState
- Функция PTP.getSettings

Функция PTP.setupEthPtp

Функция используется для настройки работы PTP на выбранном Ethernet КПМ “РИТМ”.

Синтаксис

```
PTP.setupEthPtp(ethNumber, varargin)
```

Входные параметры

- 'ethNumber': номер Ethernet интерфейса. Возможные значения "HostTarget" или "1" - "max ethernet number". *Формат - строковый*
- 'role': роль КПМ “РИТМ” при синхронизации. Возможные значения "Master", "Slave". Значение по умолчанию "Master". *Формат - строковый*
- 'delayMech': режим работы PTP. Возможные значения "E2E", "P2P" или "Auto". Значение по умолчанию "E2E". *Формат - строковый*
- 'networkTr': сетевой протокол. Возможные значения "UDPv4", "UDPv6" and "L2". Значение по умолчанию "UDPv4". *Формат - строковый*
- 'priority1': атрибут локальных часов “Приоритет 1”. Возможный диапазон 0 до 255. Значение по умолчанию 128. *Формат - числовой*
- 'priority2': атрибут локальных часов “Приоритет 2”. Возможный диапазон 0 до 255. Значение по умолчанию 128. *Формат - числовой*
- 'clockClass': атрибут локальных часов “Класс”. Возможный диапазон 0 до 255. Значение по умолчанию 248. *Формат - числовой*
- 'clockAccuracy': атрибут локальных часов “Точность”. Возможный диапазон 0 до 255. Значение по умолчанию 254. *Формат - числовой*

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.setupEthPtp('HostTarget','role','Slave', 'networkTr', 'UDPv4', 'clockClass', 255, 'clockAccuracy', 35);
```

Функция PTP.startPtp

Функция используется для запуска PTP на выбранном Ethernet КПМ “РИТМ”.

Синтаксис

```
PTP.startPtp(ethNumber)
```

Входные параметры

- 'ethNumber': номер Ethernet интерфейса. Возможные значения "HostTarget" или "1" - "max ethernet number". *Формат - строковый*

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.startPtp('1');
```

Функция PTP.stopPtp

Функция используется для остановки PTP на выбранном Ethernet КПМ “РИТМ”.

Синтаксис

```
PTP.stopPtp(ethNumber)
```

Входные параметры

- 'ethNumber': номер Ethernet интерфейса. Возможные значения "HostTarget" или "1" - "max ethernet number". *Формат - строковый*

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.stopPtp('1');
```

Функция PTP.resetSettings

Функция используется для сброса настроек PTP к значениям по умолчанию.

Синтаксис

```
PTP.resetSettings()
```

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.resetSettings();
```

Функция PTP.getState

Функция используется для получения информации о текущем состоянии КПМ “РИТМ”.

Синтаксис

```
PTP.getState()
```

Выходные параметры

- 'T': Массив строк с информацией о состоянии всех запущенных интерфейсах. *Формат - строковый*

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.getState()

ans =

  6×1 string array

    "interface: HostTarget"
    "role: Master"
    "state: "Master active""
    "interface: Ethernet 5"
    "role: Slave"
    "state: "PTP initialization""
```

Функция PTP.getSettings

Функция используется для получения информации о текущих настройках PTP КПМ “РИТМ”.

Синтаксис

```
PTP.getSettings()
```

Выходные параметры

- 'T': Массив строк с информацией о настройках интерфейсов. *Формат - строковый*

Пример использования

```
PTP = ritm.util.getPtpHandler('RITM 1');
PTP.getSettings()

ptpSettings =

  16×1 string array

    "interface: Ethernet 1"
    "delayMech: E2E"
    "networkTr: UDPv4"
    "role: Master"
    "priority1: 128"
    "priority2: 128"
    "clockClass: 255"
    "clockAccuracy: 35"
    "interface: Ethernet 5"
    "delayMech: E2E"
    "networkTr: L2"
    "role: Slave"
    "priority1: 128"
    "priority2: 128"
    "clockClass: 255"
    "clockAccuracy: 35"
```

Сетевые карты для КПМ РИТМ

В данном разделе приводится список поддерживаемых сетевых карт на КПМ РИТМ

- **ET-IN-XX**: Сетевые карты 1 Гбит Ethernet с аппаратной поддержкой IEEE 1588
- **ET-MX-X5**: Сетевые карты 25 Гбит Ethernet с аппаратной поддержкой RDMA

Модули 1 Гбит Ethernet

Содержание

- Сетевой адаптер RITMeX ET-IN-55
- Сетевой адаптер RITMeX ET-IN-65
- Поддерживаемые блоки

Сетевой адаптер RITMeX ET-IN-55

- Количество портов: 2
- Скорость передачи данных: 1 Гбит/с
- Разъём: RJ45

Сетевой адаптер RITMeX ET-IN-65

- Количество портов: 4
- Скорость передачи данных: 1 Гбит/с
- Разъём: RJ45

Поддерживаемые блоки

- Библиотека UDP
- Библиотека сырого Ethernet
- Блоки и API для работы с PTP на РИТМ

Модуль RITMeX ET-MX-X5

Содержание

- Описание
- Поддерживаемые блоки

Описание

Сетевой адаптер RITMeX ET-MX-X5

- Количество портов: 2
- Скорость передачи данных: 25 Гбит/с
- Разъём: SFP28

Поддерживаемые блоки

- Библиотека UDP
- Библиотека сырого Ethernet
- Блоки и API для работы с PTP на РИТМ
- Библиотека RDMA TCP
- Библиотека RDMA Буфер

Блоки поддержки RDMA

Содержание

- Библиотека RDMA TCP
- Библиотека RDMA Буфер

Библиотеки для работы с протоколами на базе RDMA (удалённого прямого доступа к памяти) в КПМ “РИТМ”

Библиотека RDMA TCP

Обеспечивает поддержку TCP используя прямой доступ к памяти КПМ “РИТМ”.

Библиотека RDMA Буфер

Позволяет создавать буферы для для прямого доступа к памяти удаленной машины КПМ “РИТМ”.

Библиотека блоков RITM-RDMA-TCP

Содержание

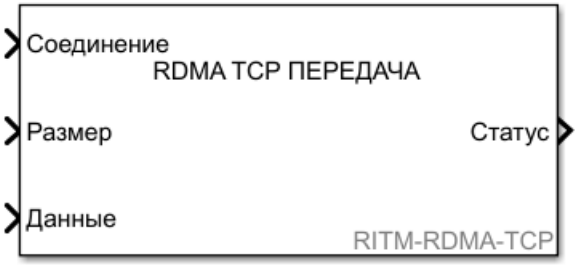
- Описание
- Библиотека блоков
- Блок RDMA TCP Сервер
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок RDMA TCP Клиент
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок RDMA TCP Отправка
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок RDMA TCP Приём
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока

Описание

Библиотека блоков для использования TCP через прямой удаленный доступ к памяти (RDMA). Перед использованием необходимо произвести настройку IP адресов для сетевых портов КПМ “РИТМ”, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

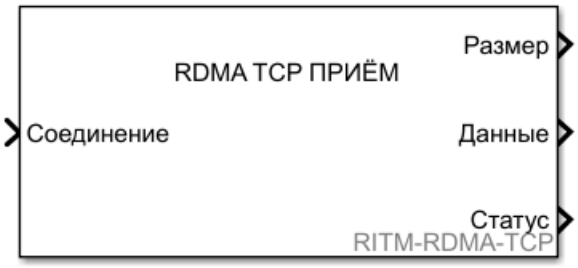
Библиотека блоков

RITM-RDMA-TCP-SERVER



RITM-RDMA-TCP-SEND

RITM-RDMA-TCP-CLIENT

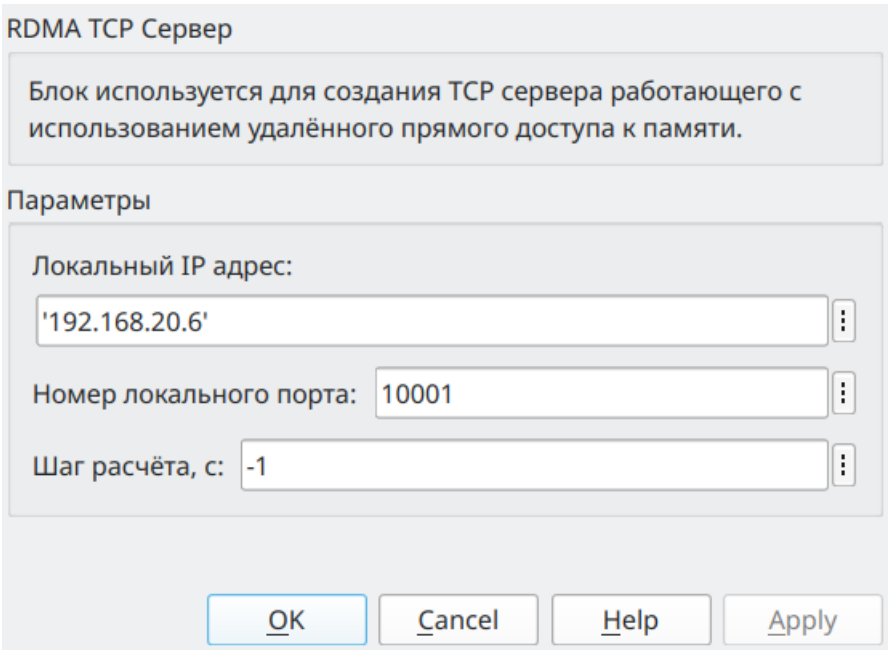


RITM-RDMA-TCP-RECV

Блок RDMA TCP Сервер

Блок используется для создания TCP сервера, работающего с использованием удаленного доступа к памяти.

Маска блока



Выходы блока

- **Соединение**: Информация о соединении необходимая для блоков “RDMA TCP Приём” и “RDMA TCP Передача”.

Параметры блока

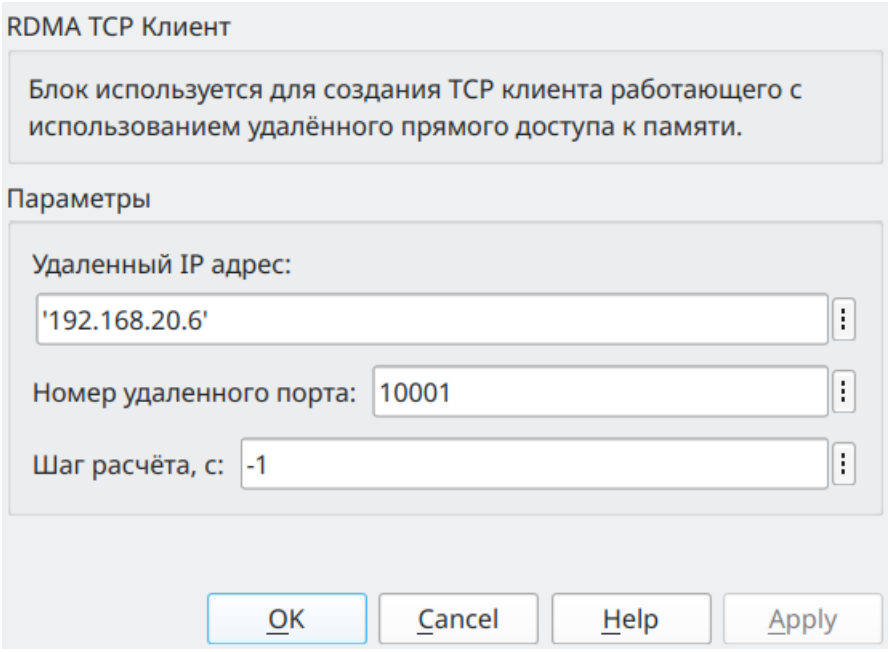
- **IP адрес**: IPv4 адрес заданный устройству с поддержкой RDMA, который будет прослушиваться сервером.

- **Номер локального порта**: Порт, который будет прослушиваться сервером.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для обновления информации о состоянии соединения.

Блок RDMA TCP Клиент

Блок используется для

Маска блока



Выходы блока

- **Соединение**: Информация о соединении необходимая для блоков “RDMA TCP Приём” и “RDMA TCP Передача”.

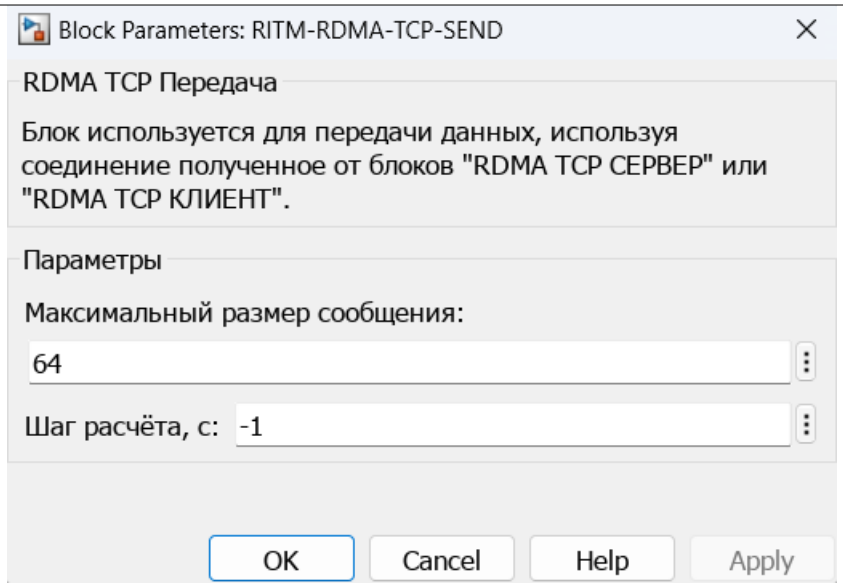
Параметры блока

- **IP адрес**: IPv4 адрес заданный устройству с поддержкой RDMA, к которому будет производиться подключение.
- **Номер локального порта**: Порт, к которому будет производиться подключение.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для обновления информации о состоянии соединения.

Блок RDMA TCP Отправка

Блок используется для передачи данных, используя соединение полученное от блоков “RDMA TCP Сервер” или “RDMA TCP Клиент”.

Маска блока



Входы блока

- **Соедиение**: Информаци о соединении полученная от блока “RDMA TCP Сервер” или “RDMA TCP Клиент”.
- **Размер**: Число байт для передачи из вектора **“Данные”**, тип данных *uint32*.
- **Данные**: Массив байт для передачи.

Выходы блока

- **Статус**: Информация о статусе завершения передачи - ненулевой статус сообщает об ошибке.

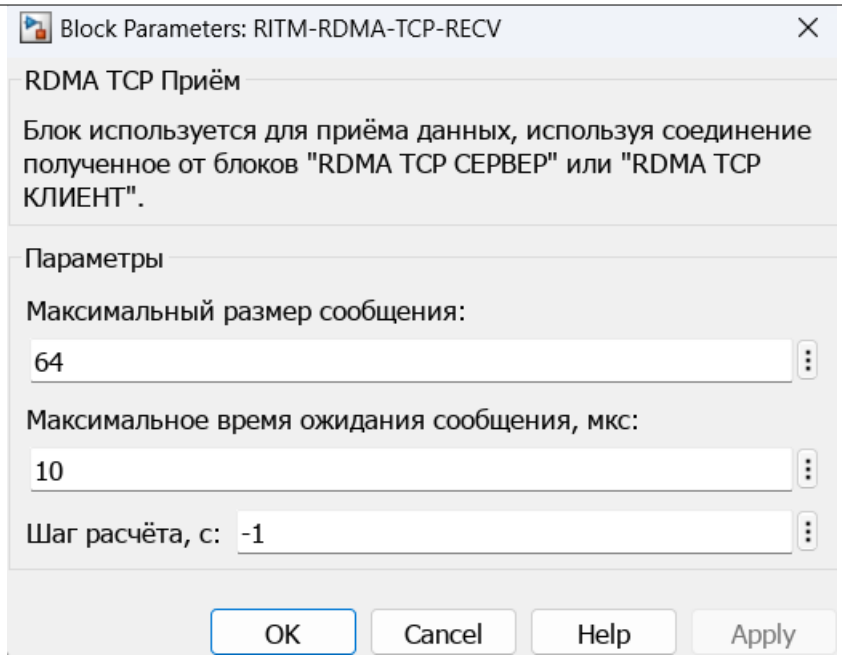
Параметры блока

- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины вектора **“Данные”**.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для отправки сообщений из приложения реального времени.

Блок RDMA TCP Приём

Блок используется для приёма данных, используя соединение полученное от блоков “RDMA TCP Сервер” или “RDMA TCP Клиент”.

Маска блока



Входы блока

- **Соедиение**: Информаци о соединении полученная от блока “RDMA TCP Сервер” или “RDMA TCP Клиент”.

Выходы блока

- **Статус**: Информация о статусе завершения передачи - ненулевой статус сообщает об ошибке.
- **Размер**: Число принятых байт, тип данных *uint32*. Не более значения параметра **“Максимальный размер сообщения”**. Если в буфере операционной системы находится больше данных, они будут считаны при следующем шаге расчета.
- **Данные**: Массив принятых байт.

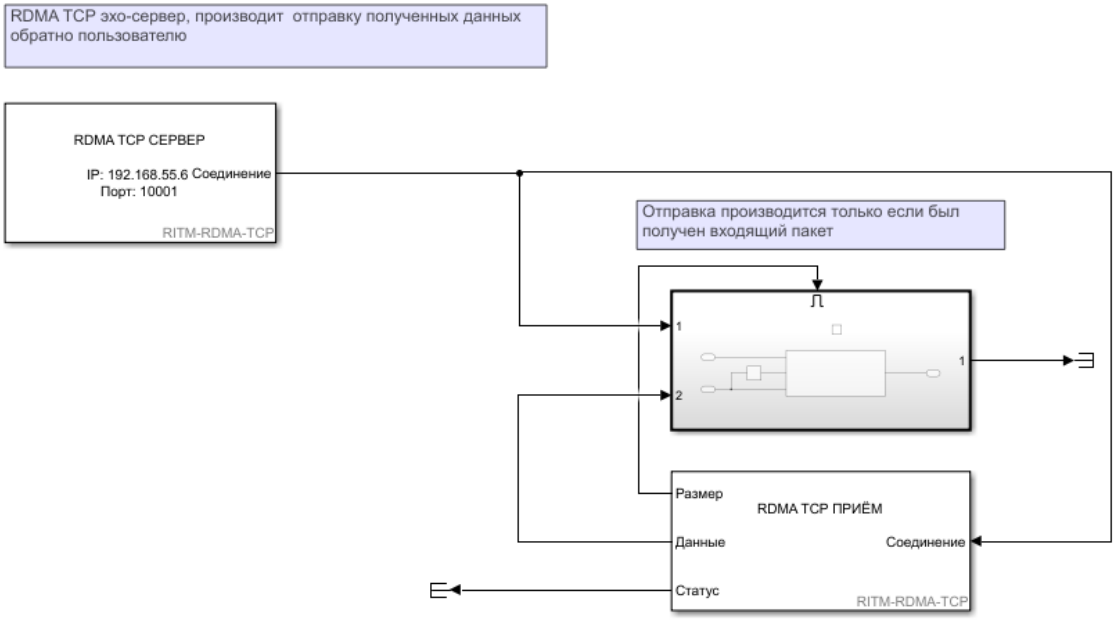
Параметры блока

- **Максимальный размер сообщения**: Максимально возможная длина принятого сообщения.
- **Максимальное время ожидания сообщения, мкс**: Максимальное время ожидания сообщения, если число принятых данных меньше значения параметра **“Максимальный размер сообщения”**.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для приема сообщений в приложение реального времени.

Примеры и демонстрации RITM-RDMA-TCP

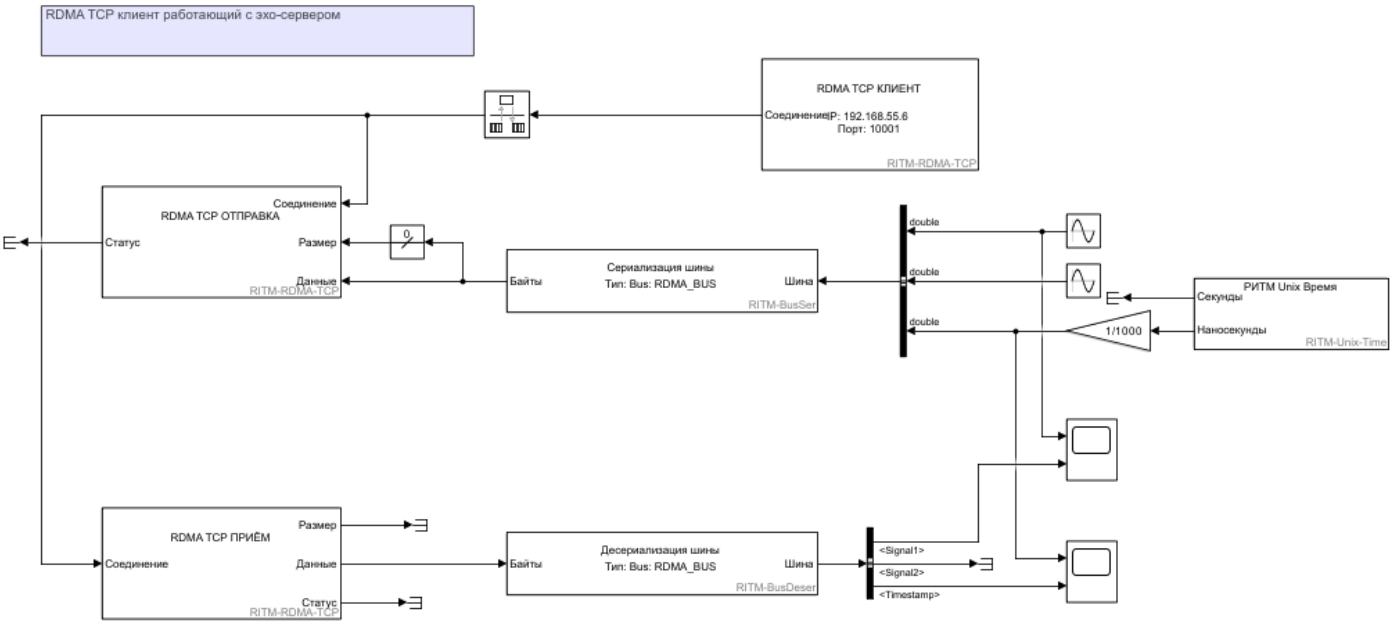
Пример сервера с использованием RDMA-TCP блоков.

```
open_system('ritmdemo_block_rdmatcp_server.slx')
```



Пример клиента с использованием RDMA-TCP блоков.

```
open_system('ritmdemo_block_rdmatcp_client.slx')
```



Библиотека блоков RITM-RDMA-BUF

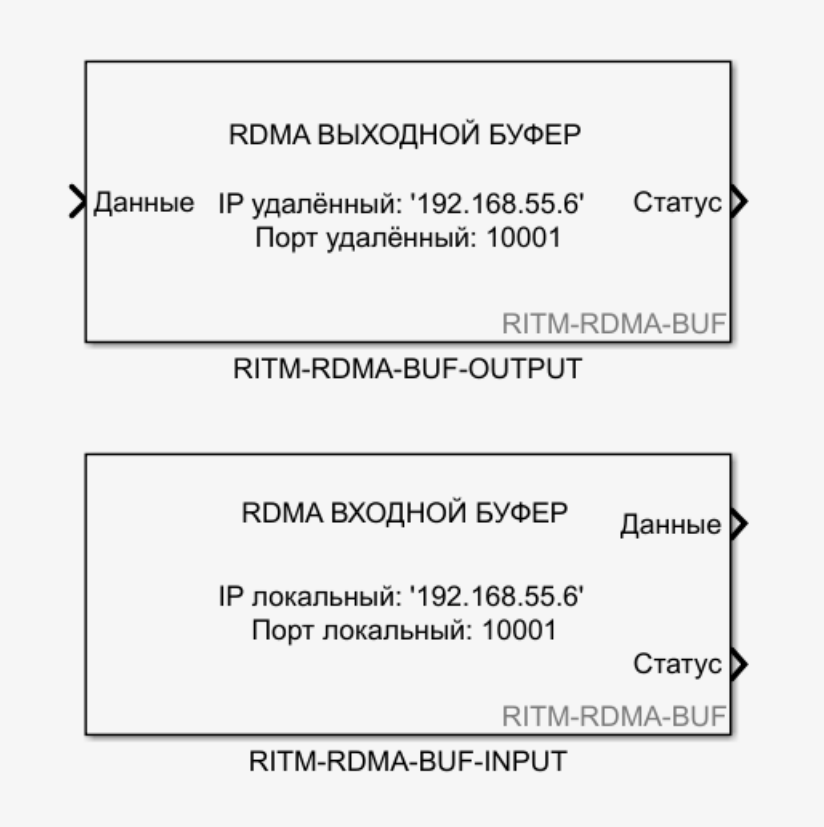
Содержание

- Описание
- Библиотека блоков
- Блок RDMA BUF Входной буфер
  - Маска блока
  - Выходы блока
  - Параметры блока
- Блок RDMA BUF Выходной буфер
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Тип **ritm_rdma_buf_status**

Описание

Библиотека блоков для создания буферов памяти с возможностью прямого удаленного доступа к ним со сторонней машины. Перед использованием необходимо произвести настройку IP адресов для сетевых портов КПМ “РИТМ”, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

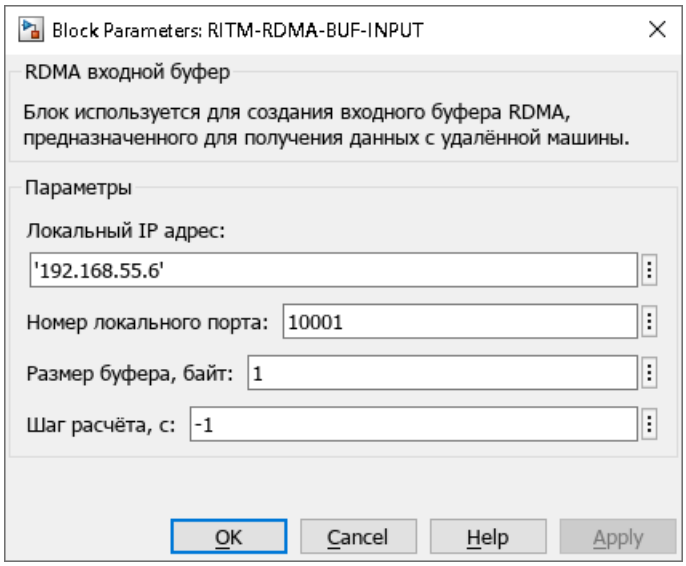
Библиотека блоков



Блок RDMA BUF Входной буфер

Блок используется для создания буфера RDMA, предназначенного для получения данных с удаленной машины.

Маска блока



Выходы блока

- **Данные**: Массив принятых байт.
- **Статус**: Информация о состоянии соединения. Тип **ritm_rdma_buf_status**

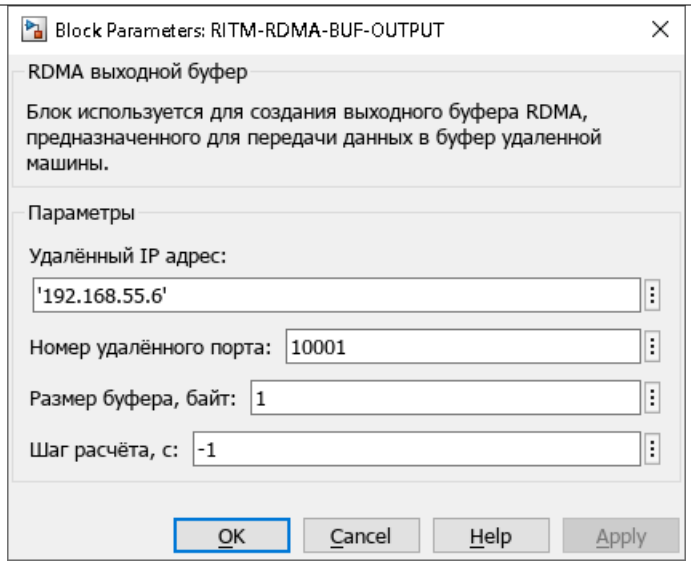
Параметры блока

- **Локальный IP адрес**: IPv4 адрес заданный устройству с поддержкой RDMA, на котором будет прослушиваться запросы на соединение с буфером от внешних устройств.
- **Номер локального порта**: Порт, который будет прослушиваться.
- **Размер буфера**: Размер входного буфера в байтах.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах, с которым будет происходить обновление выходных портов.

Блок RDMA BUF Выходной буфер

Блок используется для создания буфера RDMA, предназначенного для передачи данных на удаленную машину.

Маска блока



Входы блока

- **Данные**: Массив байт для передачи.

Выходы блока

- **Статус**: Информация о состоянии соединения. Тип **ritm_rdma_buf_status**

Параметры блока

- **Удалённый IP адрес**: IPv4 адрес заданный устройству с поддержкой RDMA, к которому будет производиться подключение.
- **Номер удаленного порта**: Порт, к которому будет производиться подключение.
- **Размер буфера**: Размер выходного буфера в байтах.
- **Шаг расчёта**: Позволяет задать шаг расчета в секундах для обновления информации о состоянии соединения.

Тип **ritm_rdma_buf_status**

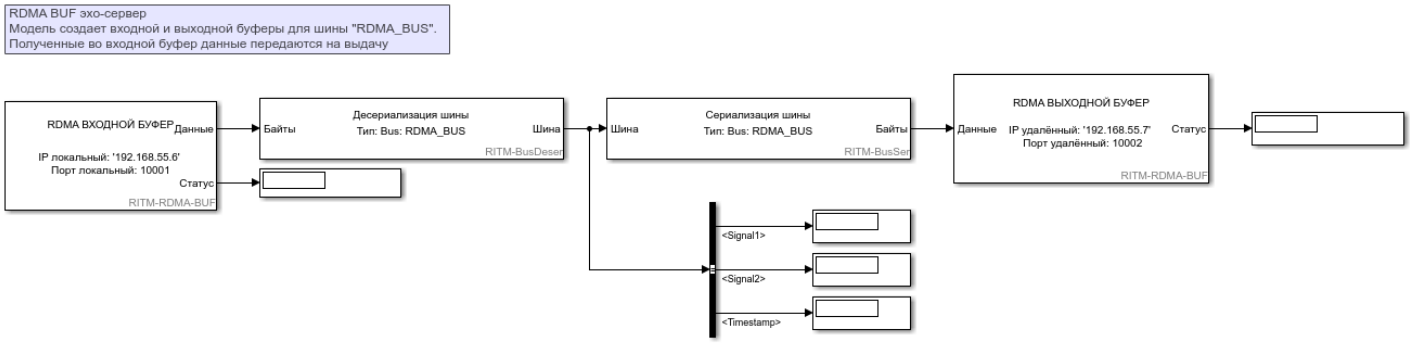
Перечисление типа Simulink.IntEnumType, содержит код состояния соединения.

- **RITM_RDMA_BUF_OK(0)**: соединение установлено, данные передаются.
- **RITM_RDMA_BUF_NO_CONN(1)**: соединение не установлено.
- **RITM_RDMA_BUF_CONN_ERR(2)**: произошла ошибка на установленном соединении.

Примеры и демонстрации RITM-RDMA-BUF

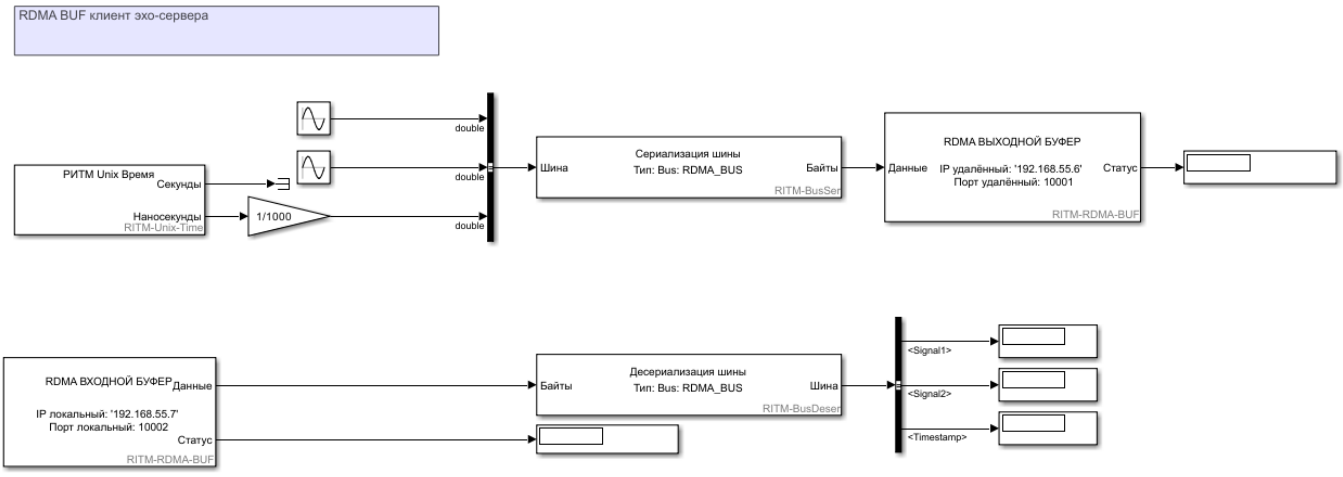
Пример сервера с использованием RDMA-BUF блоков.

```
open_system('ritmdemo_block_rdmabuf_server.slx')
```



Пример клиента с использованием RDMA-BUF блоков.

```
open_system('ritmdemo_block_rdmabuf_client.slx')
```



Блоки поддержки Modbus

Содержание

- Общие сведения
- Особенности работы
- Поддерживаемые функции протокола Modbus
- Блоки для работы с Modbus
- Руководство по использованию библиотеки

Общие сведения

Библиотека блоков поддержки протокола Modbus предназначена для связи приложения реального времени, запущенного на КПМ РИТМ, с устройствами АСУ ТП по протоколу Modbus.

Особенности работы

При применении данной библиотеки КПМ РИТМ, на котором запущено приложение реального времени, выступает как клиент сети Modbus. Поддерживается одновременная работа с 16 серверами (слейвами) сети Modbus.

Поддерживаемые функции протокола Modbus

Поддерживаются следующие функции протокола Modbus:

- 1 (0x01) — чтение значений из нескольких регистров флагов (Read Coil Status).
- 2 (0x02) — чтение значений из нескольких дискретных входов (Read Discrete Inputs).
- 3 (0x03) — чтение значений из нескольких регистров хранения (Read Holding Registers).
- 4 (0x04) — чтение значений из нескольких регистров ввода (Read Input Registers).
- 5 (0x05) — запись значения одного флага (Force Single Coil).
- 6 (0x06) — запись значения в один регистр хранения (Preset Single Register).
- 15 (0x0F) — запись значений в несколько регистров флагов (Force Multiple Coils).
- 16 (0x10) — запись значений в несколько регистров хранения (Preset Multiple Registers).

Блоки для работы с Modbus

- Блок настройки соединения Modbus TCP
- Блок настройки соединения Modbus RTU
- Блоки приема данных
- Блоки отправки данных
- Блоки для обработки данных

Руководство по использованию библиотеки

Для начала работы с библиотекой рекомендуется ознакомиться с разделами

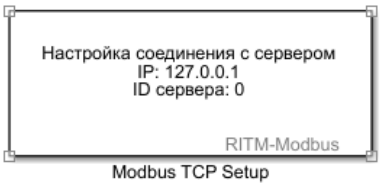
документации Работа с библиотекой блоков Modbus TCP и Работа с библиотекой блоков Modbus RTU

Блок настройки соединения Modbus TCP

Содержание

- Описание
- Маска блока
- Параметры блока

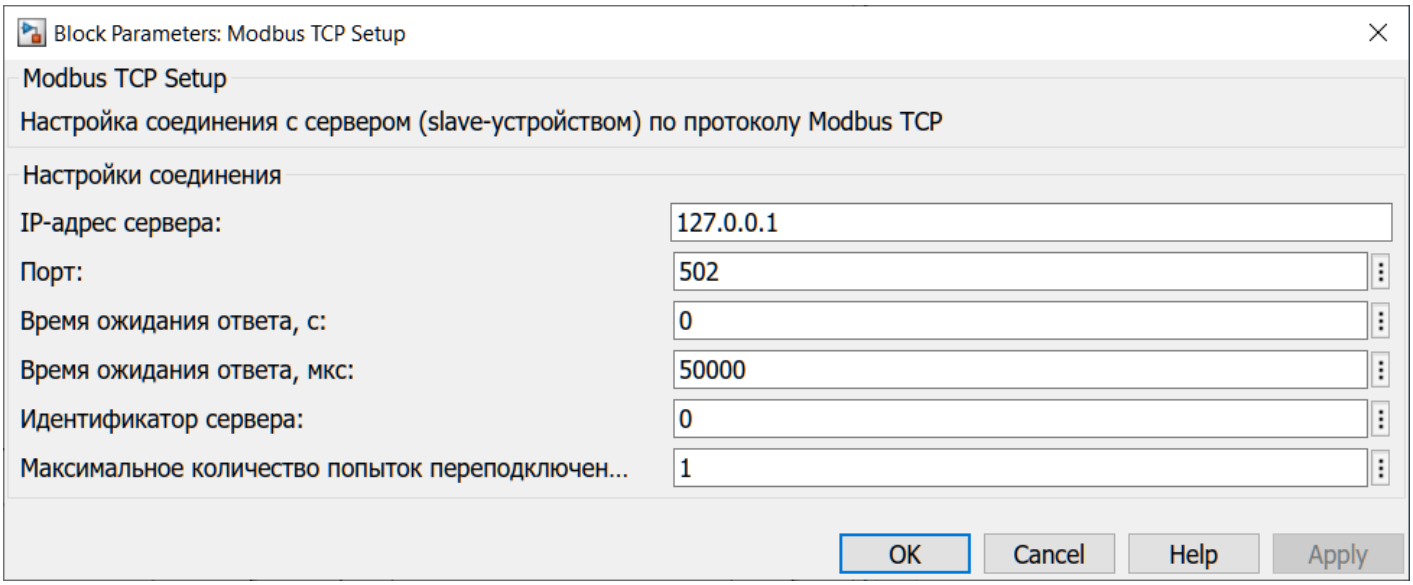
Блок настройки соединения (Modbus TCP Setup).



Описание

Настройка соединения с сервером (slave-устройством) по протоколу Modbus TCP.

Маска блока



Параметры блока

- ***IP адрес сервера***: IP адрес для соединения.

  ВАЖНО: необходимо указать IP адрес. *FQDN имя хоста не поддерживается.*

- ***Порт***: Порт сервера. Стандартное значение - *502*.

- ***Время ожидания ответа, с***: Таймаут ответа от слейв-устройства, в секундах.

- ***Время ожидания ответа, мкс***: Таймаут ответа от слейв-устройства, в микросекундах.

  Значения таймаутов указываются для каждого уникального идентификатора сервера!

- ***Идентификатор сервера***: Идентификатор соединения. Данный идентификатор используется в блоках приема и передачи. Допустимые значения: от 0 до 15.

- ***Максимальное количество попыток переподключения***: Количество попыток

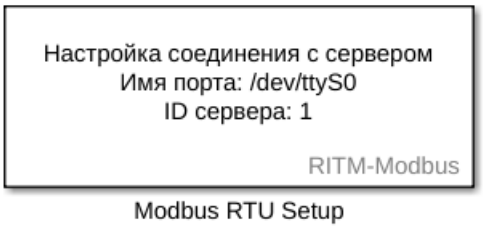
переподключения к серверу при инициализации модели.

Блок настройки соединения Modbus RTU

Содержание

- Описание
- Маска блока
- Параметры блока

Блок настройки соединения (Modbus RTU Setup).

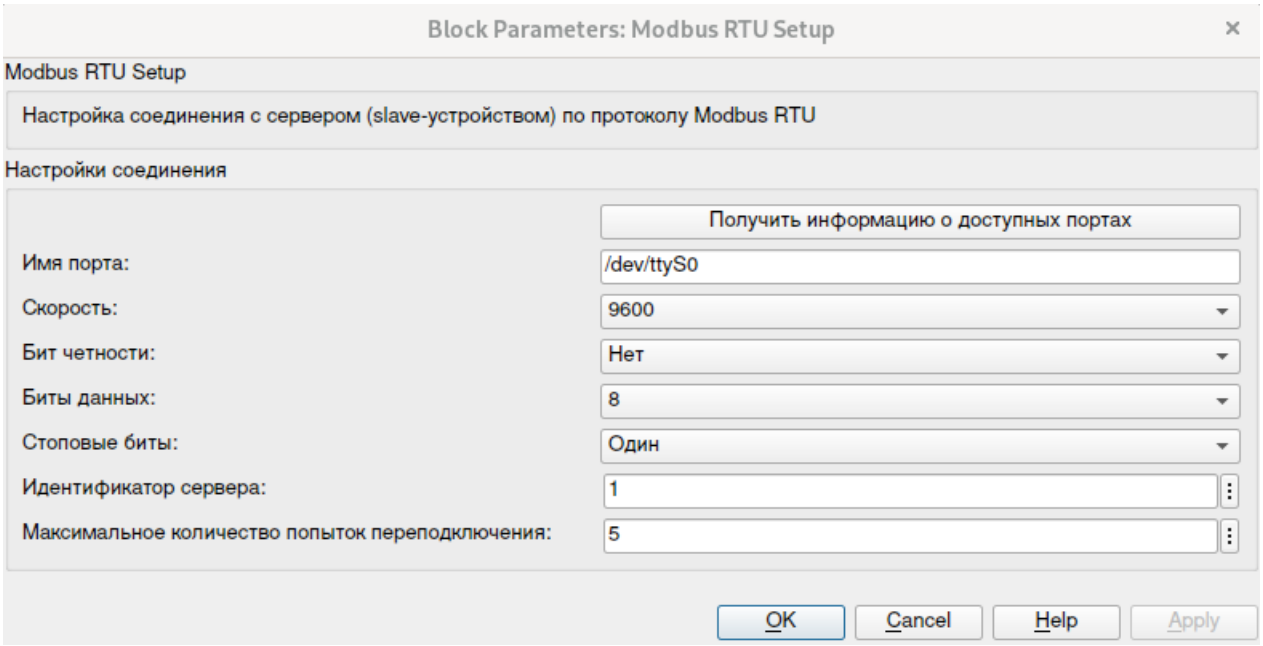


Modbus RTU Setup

Описание

Настройка соединения с сервером (slave-устройством) по протоколу Modbus RTU.

Маска блока



Параметры блока

- ***Имя порта***: Задает имя порта для настройки. Имена доступных портов могут быть получены во всплывающем окне при нажатии на кнопку “Получить информацию о доступных портах” в маске блока.
- ***Скорость***: Задает скорость приемопередачи. Возможные варианты: *9600*, *19200*, *38400*, *57600*, *115200* бод.
- ***Бит четности***: Задает метод определения ошибок приемопередачи. Возможные значения: нет, проверка на четность (*Odd*), проверка на нечетность (*Even*).

- ***Биты данных***: Задает количество бит данных приемопередачи. Возможные значения: *5-бит*, *6-бит*, *7-бит*, *8-бит*.
- ***Стоповые биты***: Задает количество стоповых бит приемопередачи. Возможные значения: *один*, *два*.
- ***Идентификатор сервера***: Идентификатор соединения. Данный идентификатор используется в блоках приема и передачи. Допустимые значения: от *0* до *15*.
- ***Максимальное количество попыток переподключения***: Количество попыток переподключения к серверу при инициализации модели.

Блоки приема данных

Содержание

- Блок Modbus Read Registers
  - Описание
  - Маска блока
  - Параметры блока
  - Выходы блока
- Блок Modbus Read Coils
  - Описание
  - Маска блока
  - Параметры блока
  - Выходы блока

Modbus Read Registers

Modbus Read Coils

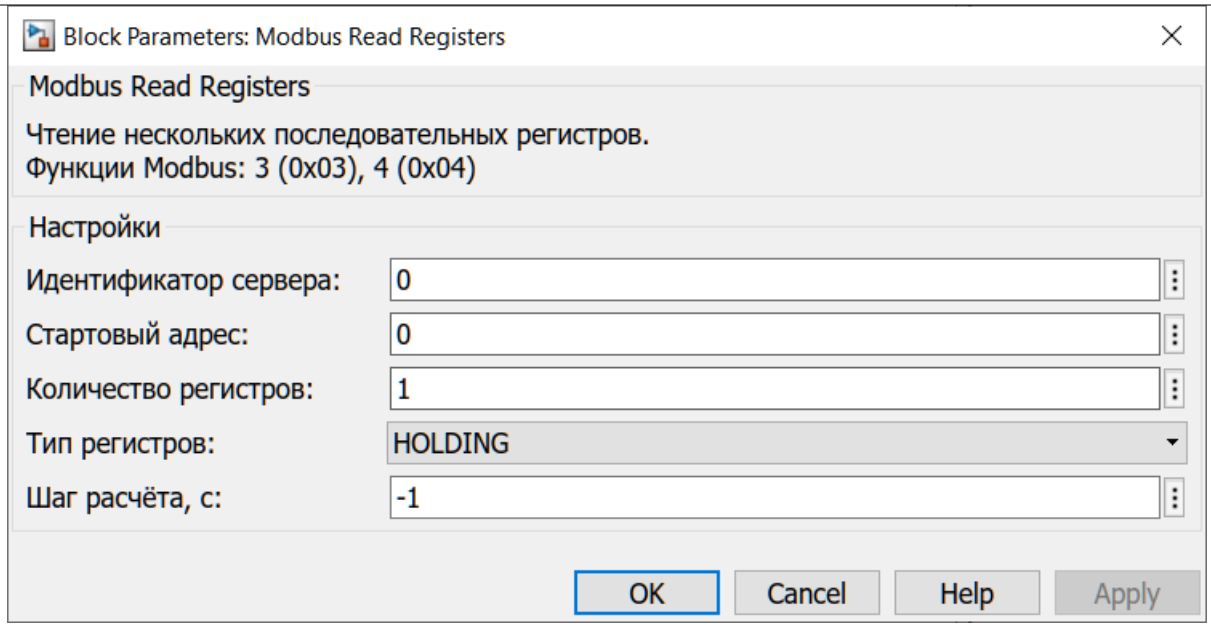
Блок Modbus Read Registers

Описание

Чтение нескольких последовательных регистров.

Функции Modbus: 3 (0x03), 4 (0x04).

Маска блока



Параметры блока

- ***Идентификатор сервера***: Идентификатор слейв-устройства.
- ***Стартовый адрес***: Начальный адрес для чтения, *без учета смещения*.
- ***Количество регистров***: Количество регистров для чтения.
- ***Тип регистров***:
  - *HOLDING*: Регистры хранения.
  - *INPUT*: Регистры ввода.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для получения данных в приложении реального времени.

Выходы блока

- ***Данные***: Принятые данные. Вектор из 16-разрядных беззнаковых целых (тип данных *uint16*).
- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции.

Если операция выполнена успешно, будет возвращено количество прочитанных регистров.

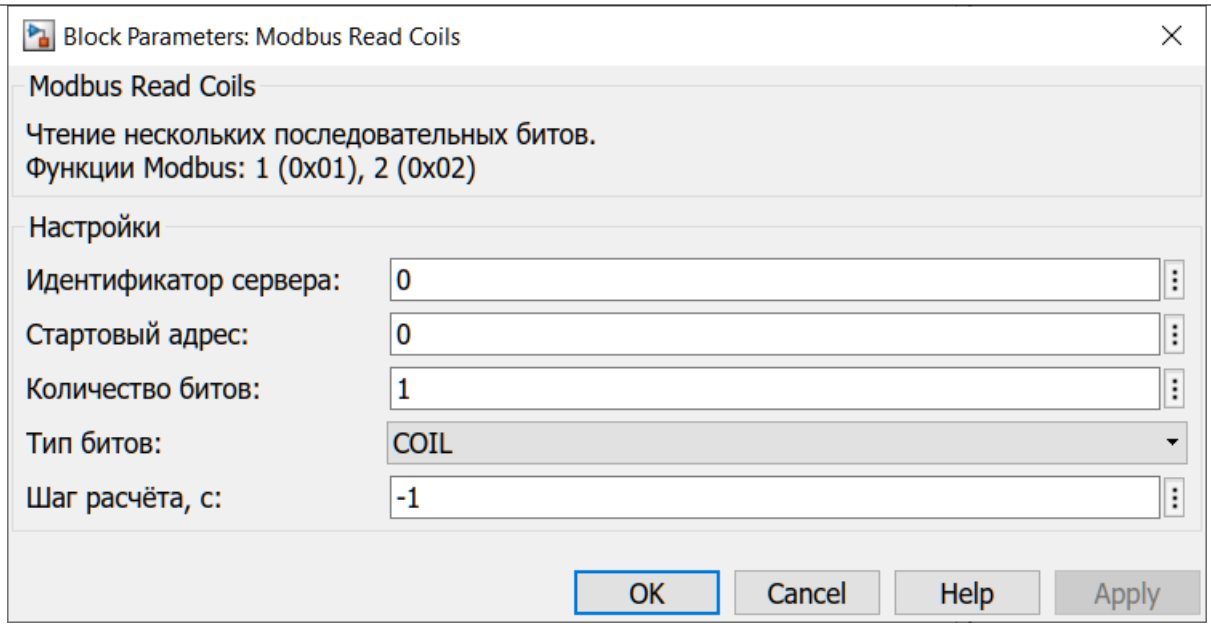
Блок Modbus Read Coils

Описание

Чтение нескольких последовательных битов.

Функции Modbus: 1 (0x01), 2 (0x02).

Маска блока



Параметры блока

- ***Идентификатор сервера***: Идентификатор слейв-устройства.
- ***Стартовый адрес***: Начальный адрес для чтения, *без учета смещения*.
- ***Количество битов***: Количество битов для чтения.
- ***Тип битов***:
  - **COIL**: Регистры флагов.
  - **DI**: Дискретные входы.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для получения данных в приложении реального времени.

Выходы блока

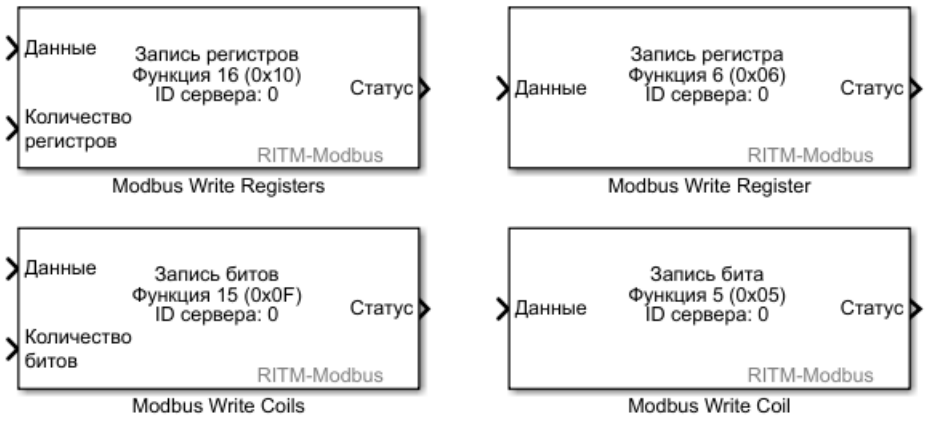
- ***Данные***: Принятые данные. Вектор из 8-разрядных беззнаковых целых (тип данных *uint8*).
- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции.

Если операция выполнена успешно, будет возвращено количество прочитанных битов.

Блоки отправки данных

Содержание

- Блок Modbus Write Registers
  - Описание
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выходы блока
- Блок Modbus Write Coils
  - Описание
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выходы блока
- Блок Modbus Write Register
  - Описание
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выходы блока
- Блок Modbus Write Coil
  - Описание
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выходы блока



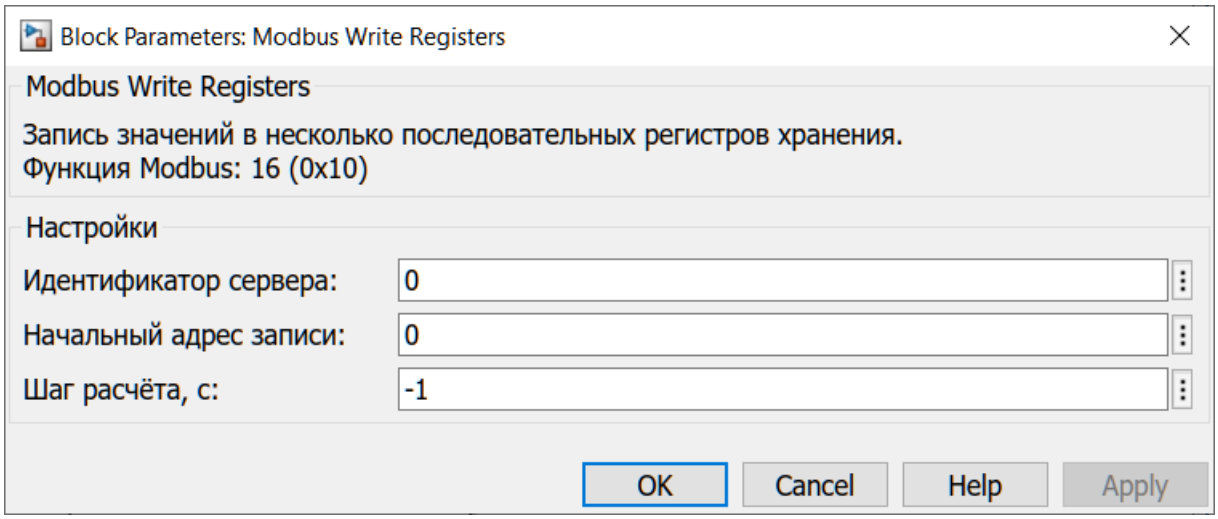
Блок Modbus Write Registers

Описание

Запись значений в несколько последовательных регистров хранения.

Функция Modbus: 16 (0x10).

Маска блока



Параметры блока

- ***Идентификатор***: Идентификатор слейв-устройства.
- ***Начальный адрес записи***: Начальный адрес для записи, *без учета смещения*.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для отправки данных из приложения реального времени.

Входы блока

- ***Данные***: Скаляр или вектор 16-разрядных беззнаковых целых (тип данных *uint16*) для записи. Длина вектора не должна превышать *123*.

Выходы блока

- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции.

Если операция выполнена успешно, будет возвращено количество записанных регистров.

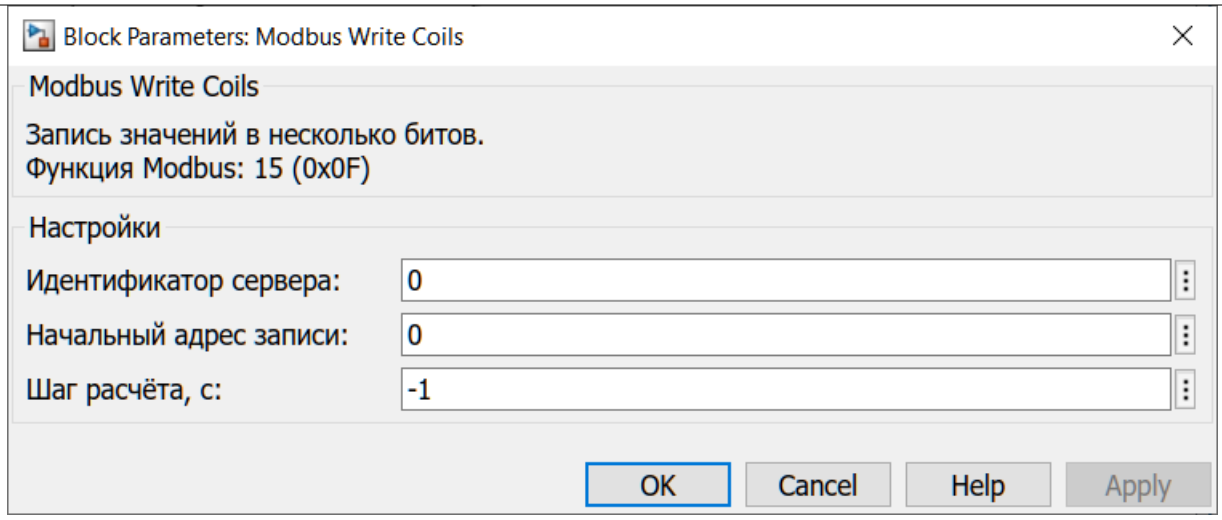
Блок Modbus Write Coils

Описание

Запись значений в несколько битов.

Функция Modbus: 15 (0x0F).

Маска блока



Параметры блока

- ***Идентификатор сервера***: Идентификатор слейв-устройства.
- ***Начальный адрес записи***: Начальный адрес для записи, *без учета смещения*.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для отправки данных из приложения реального времени.

Входы блока

- ***Данные***: Скаляр или вектор 8-разрядных беззнаковых целых (тип данных *uint8*) для записи. Длина вектора не должна превышать *1968*.

Выходы блока

- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции.

Если операция выполнена успешно, будет возвращено количество записанных битов.

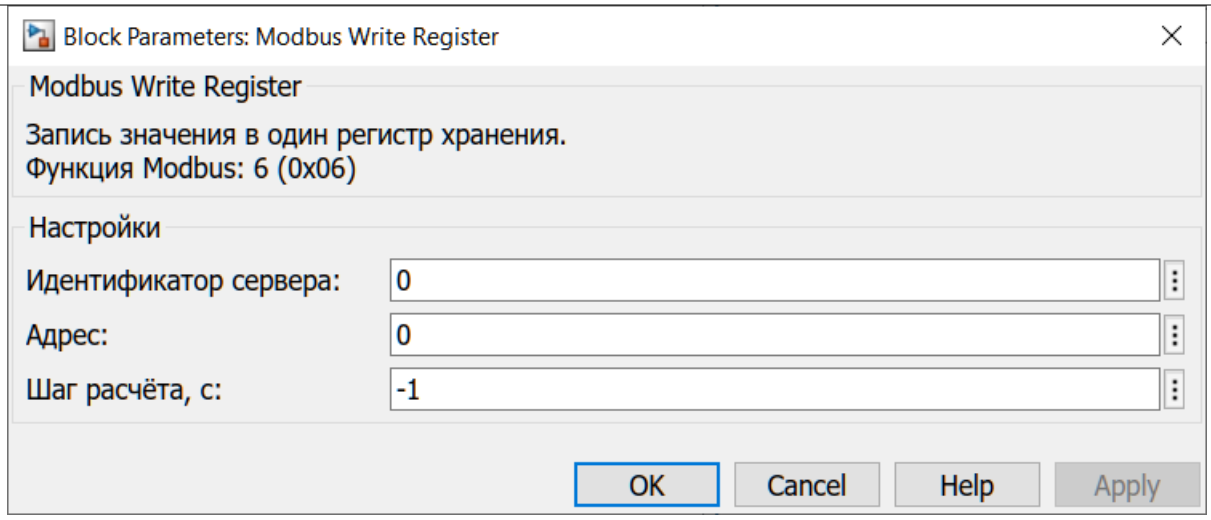
Блок Modbus Write Register

Описание

Запись значения в один регистр хранения.

Функция Modbus: 6 (0x06).

Маска блока



Параметры блока

- ***Идентификатор сервера***: Идентификатор слейв-устройства.
- ***Адрес***: Адрес для записи, *без учета смещения*.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для отправки данных из приложения реального времени.

Входы блока

- ***Данные***: 16-разрядное беззнаковое целое (тип данных *uint16*) для записи.

Выходы блока

- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции

Если операция выполнена успешно, будет возвращена 1.

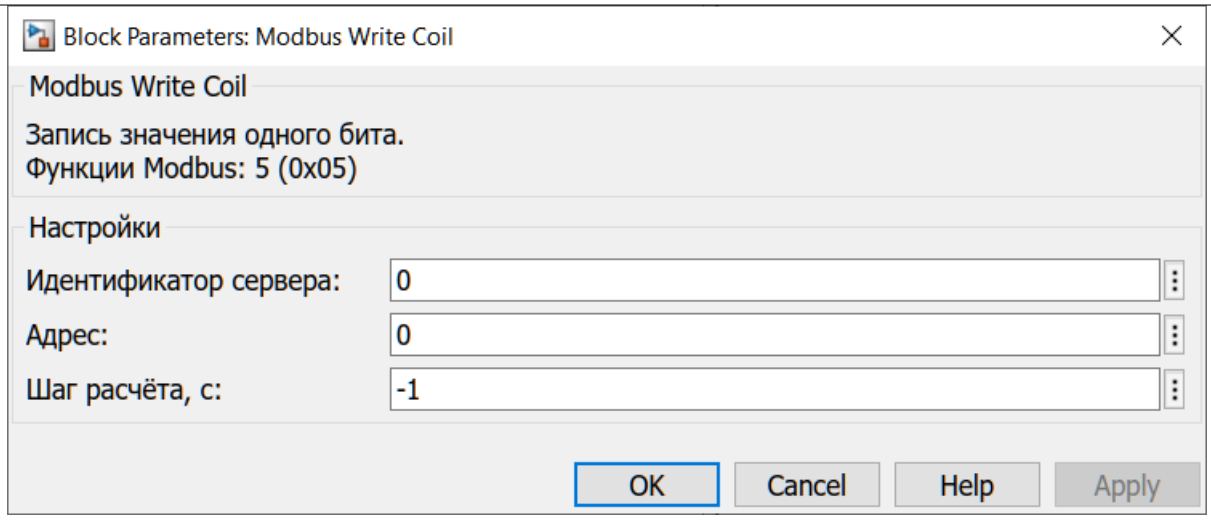
Блок Modbus Write Coil

Описание

Запись одного флага. Блок реализует следующие функции протокола:

- 5 (0x05) - Force Single Coil.

Маска блока



Параметры блока

- ***Идентификатор сервера***: Идентификатор слейв-устройства.
- ***Адрес***: Адрес для записи, *без учета смещения*.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для отправки данных из приложения реального времени.

Входы блока

- ***Данные***: 8-разрядное беззнаковое целое (тип данных *uint8*) для записи.

Выходы блока

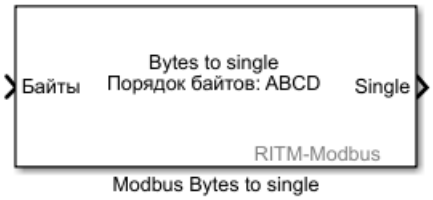
- ***Статус***: Статус операции. Предназначен для контроля ошибок и их обработки в модели. Коды ошибок:
  - *-1*: Ошибка операции.

Если операция выполнена успешно, будет возвращена 1.

Блоки для обработки данных

Содержание

- Блок Modbus Bytes to single
  - Описание
  - Маска блока
  - Параметры блока
- Блок Modbus Single to bytes
  - Описание
  - Маска блока
  - Параметры блока



Modbus Bytes to single

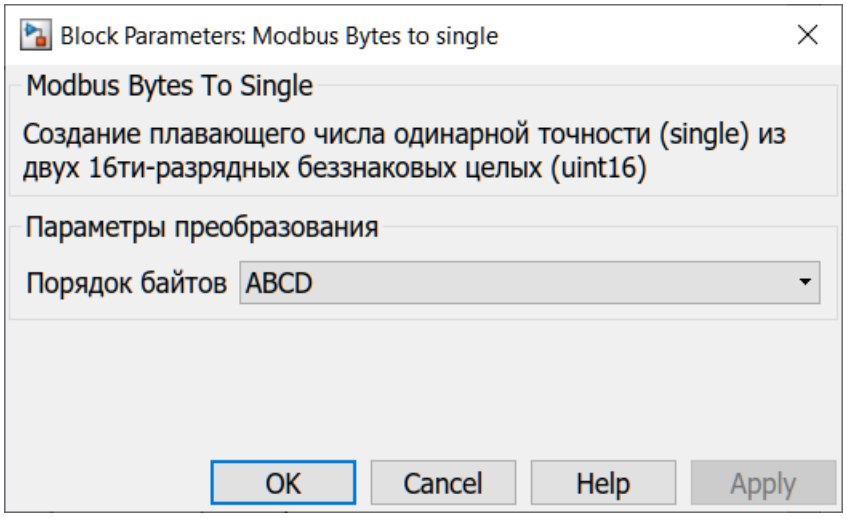
Modbus Single to bytes

Блок Modbus Bytes to single

Описание

Создание числа с плавающей точкой одинарной точности (тип данных *single*) из двух 16 разрядных целых (тип данных *uint16*).
В ходе преобразования возможно задать порядок байтов в итоговом значении.

Маска блока



Параметры блока

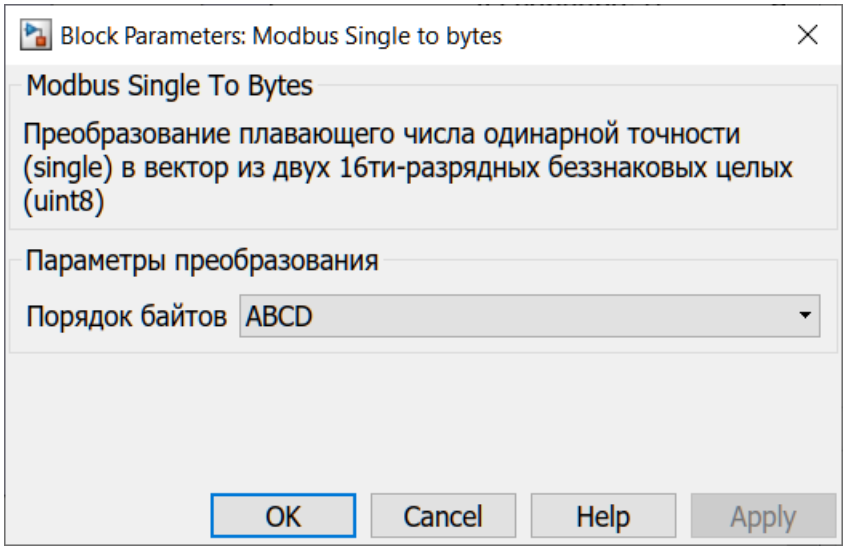
- ***Порядок байтов***: Порядок следования байтов.

Блок Modbus Single to bytes

Описание

Создание вектора, состоящего из двух 16 разрядных целых (тип данных *uint16*), из числа с плавающей точкой одинарной точности (тип данных *single*).
В ходе преобразования возможно задать порядок байтов в итоговом векторе.

Маска блока



Параметры блока

- ***Порядок байтов***: Порядок следования байтов.

Работа с библиотекой блоков Modbus TCP

Содержание

- Рекомендуемое ПО для ознакомления с библиотекой
- Особенности работы с симулятором ModRSsim2
- Особенности работы с симулятором ModbusPal
- Особенности работы библиотеки
  - Карта смещений
- Начало работы
  - Минимальное необходимое ПО
  - Простой обмен данными
  - Рекомендуемые приемы моделирования

Рекомендуемое ПО для ознакомления с библиотекой

Для ознакомления с основными возможностями библиотеки рекомендуется использовать симуляторы слейв-устройств, такие как:

- ModRSsim2
- Modbus Universal MasterOPC
- pyModSlave
- ModbusPal

Особенности работы с симулятором ModRSsim2

Строго рекомендуется пересоздавать сокеты после сеанса работы с симулятором. Причина заключается в том, что после закрытия соединения сокет остается активным некоторое время, что приводит к невозможности подключения.

Особенности работы с симулятором ModbusPal

Симулятор поддерживает работу только с регистрами хранения (Holding Registers) и регистрами флагов (Coils)

Особенности работы библиотеки

Карта смещений

Библиотека блоков использует следующую карту смещений:

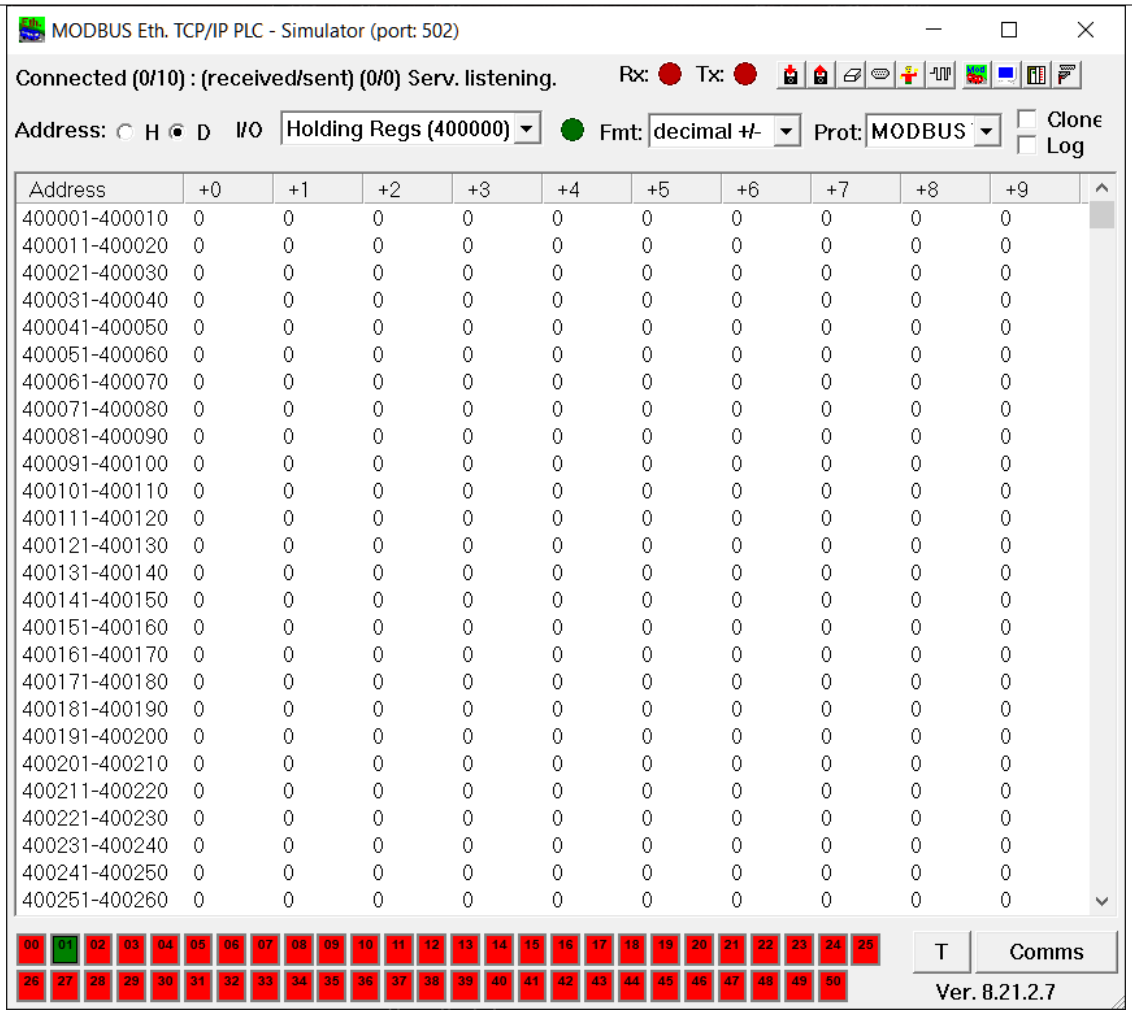
- Coil Outputs - 00000
- Digital Inputs - 10000
- Analog Inputs - 30000
- Holding Registers - 40000

Начало работы

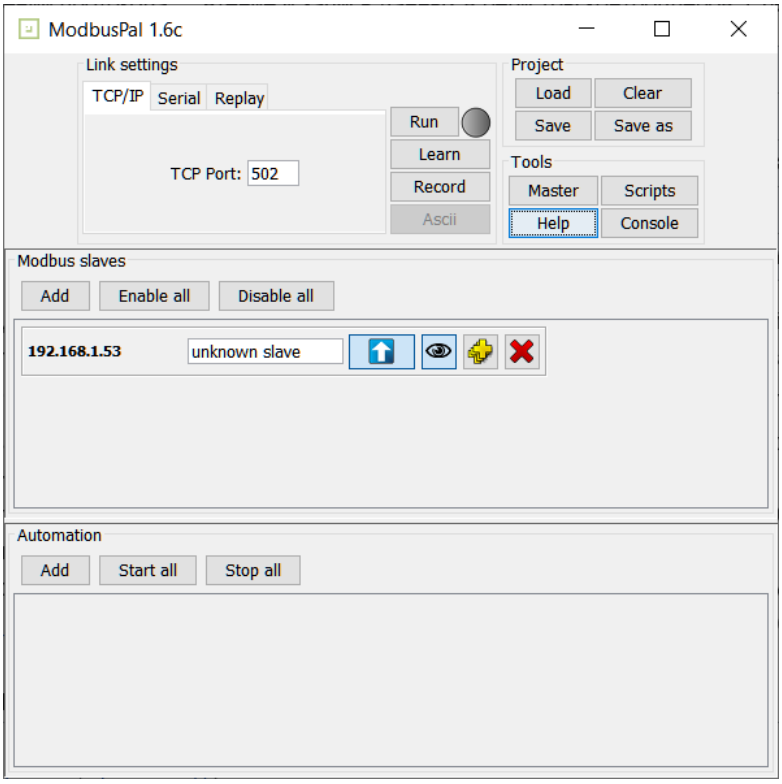
Минимальное необходимое ПО

- ModRSsim2 (Для ОС Windows)

ВНИМАНИЕ: для корректной работы демонстрации требуется запустить ModRSSim2 и включить станцию 1



- ModbusPal (кроссплатформенный симулятор, требует Java)

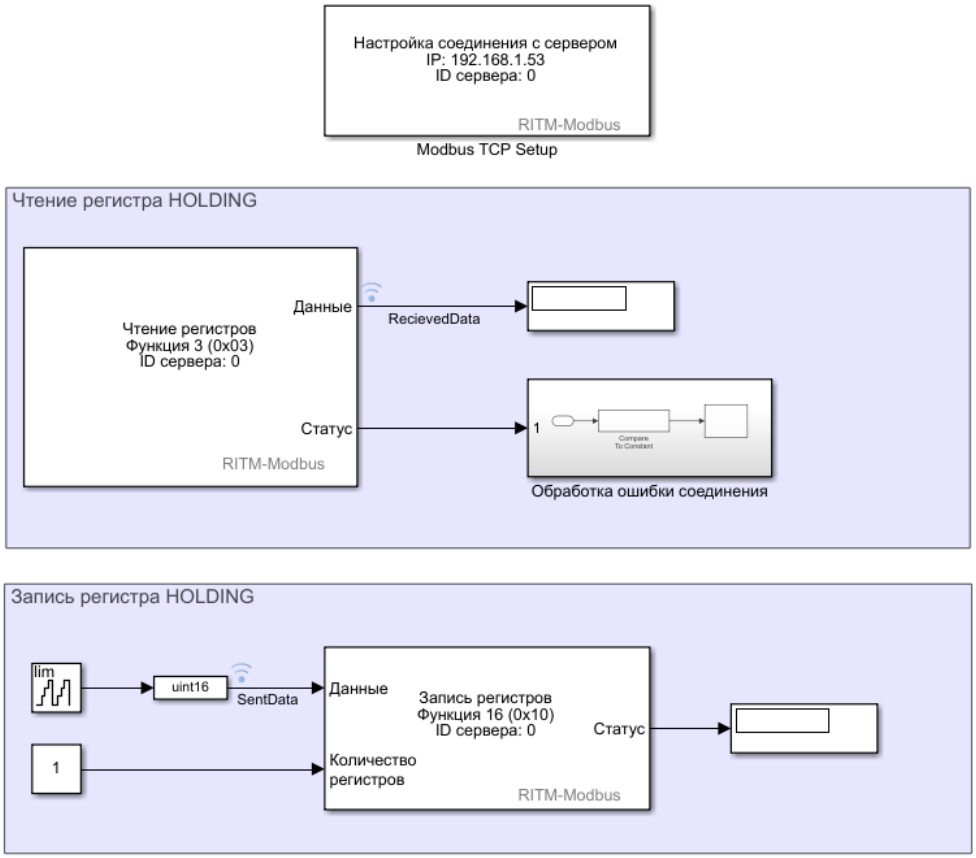


ВНИМАНИЕ: для быстрого старта рекомендуется войти в обучающий режим перед нажатием кнопки Run. Для этого нажмите кнопку Learn

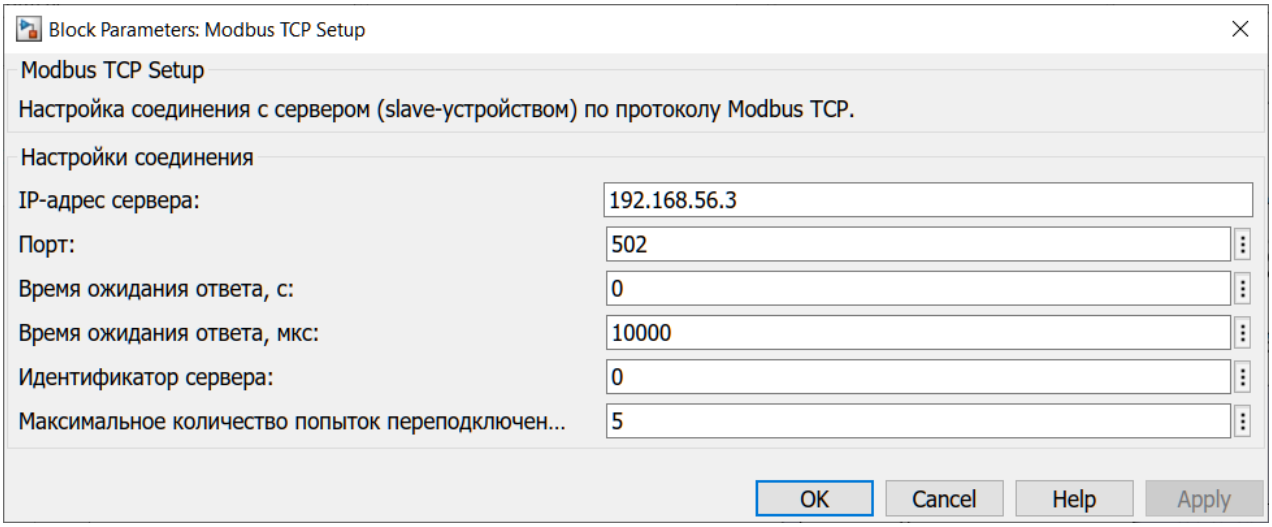
Простой обмен данными

Рассмотрим модель:

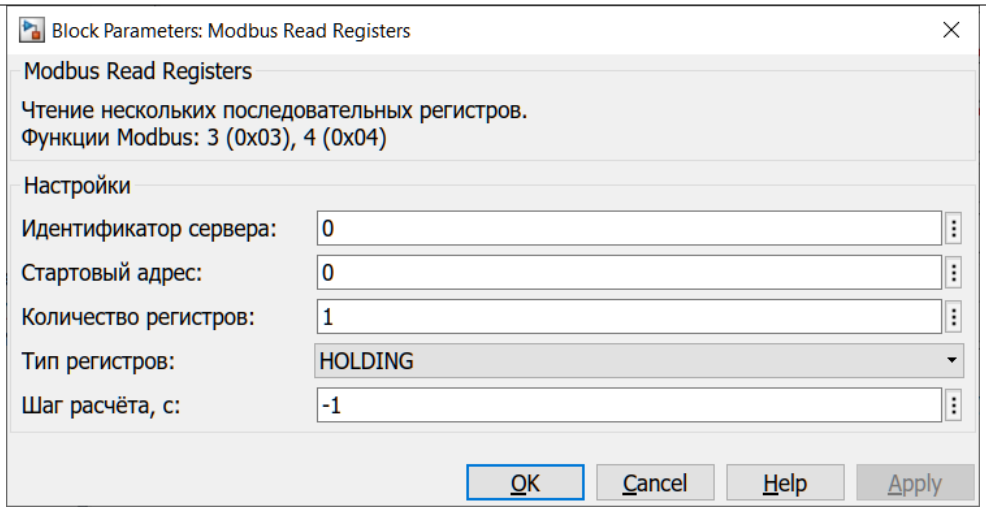
```
open_system('ritmdemo_block_modbus_ops')
```



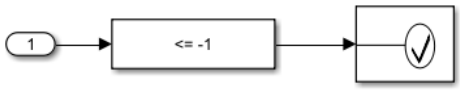
Соединение с сервером Modbus устанавливается при помощи блока Modbus TCP Setup. Его параметры следующие:



Из настроек видно, что устанавливается связь с сервером 192.168.56.3:502, время ожидания ответа - 0,01 секунд, а максимальное количество попыток переподключения к серверу - 5. Данному соединению присваивается внутренний идентификатор 0. В дальнейшем, данный идентификатор будет использоваться в блоках чтения и записи, например, в блоке **Modbus Read Registers**:



Рассмотрим область Чтение регистра HOLDING. В этой области, выход “Статус” блока чтения регистров подается на подсистему “Обработка ошибки соединения”.



Назначение этой подсистемы - отслеживать статус операции. Если статус операции равен **-1**, то блок Assertion выдаст предупреждение. Применяя такую конструкцию моделирования, можно останавливать исполнение модели при длительном обрыве соединения.

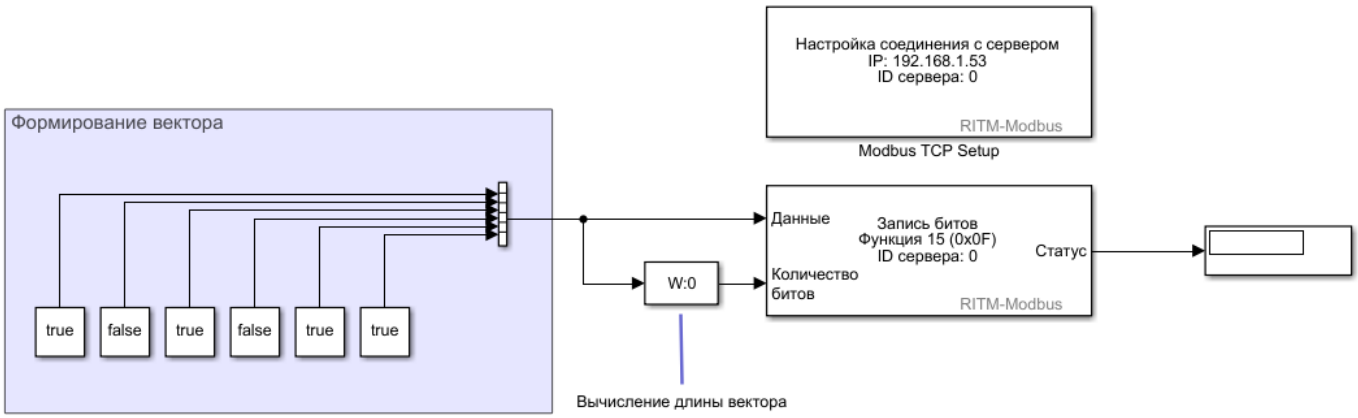
Рекомендуемые приемы моделирования

Запись множественных значений

В том случае, если требуется записать несколько последовательных регистров или битов, рекомендуется формировать вектор значений с помощью блока Vector Concatenate. Блоки для записи последовательности регистров или битов требуют явного задания количества записываемых данных. Количество данных вычисляется с помощью блока Probe Width.

В качестве примера обратитесь к модели:

```
open_system('ritmdemo_block_modbus_vectors')
```



Работа с библиотекой блоков Modbus RTU

Содержание

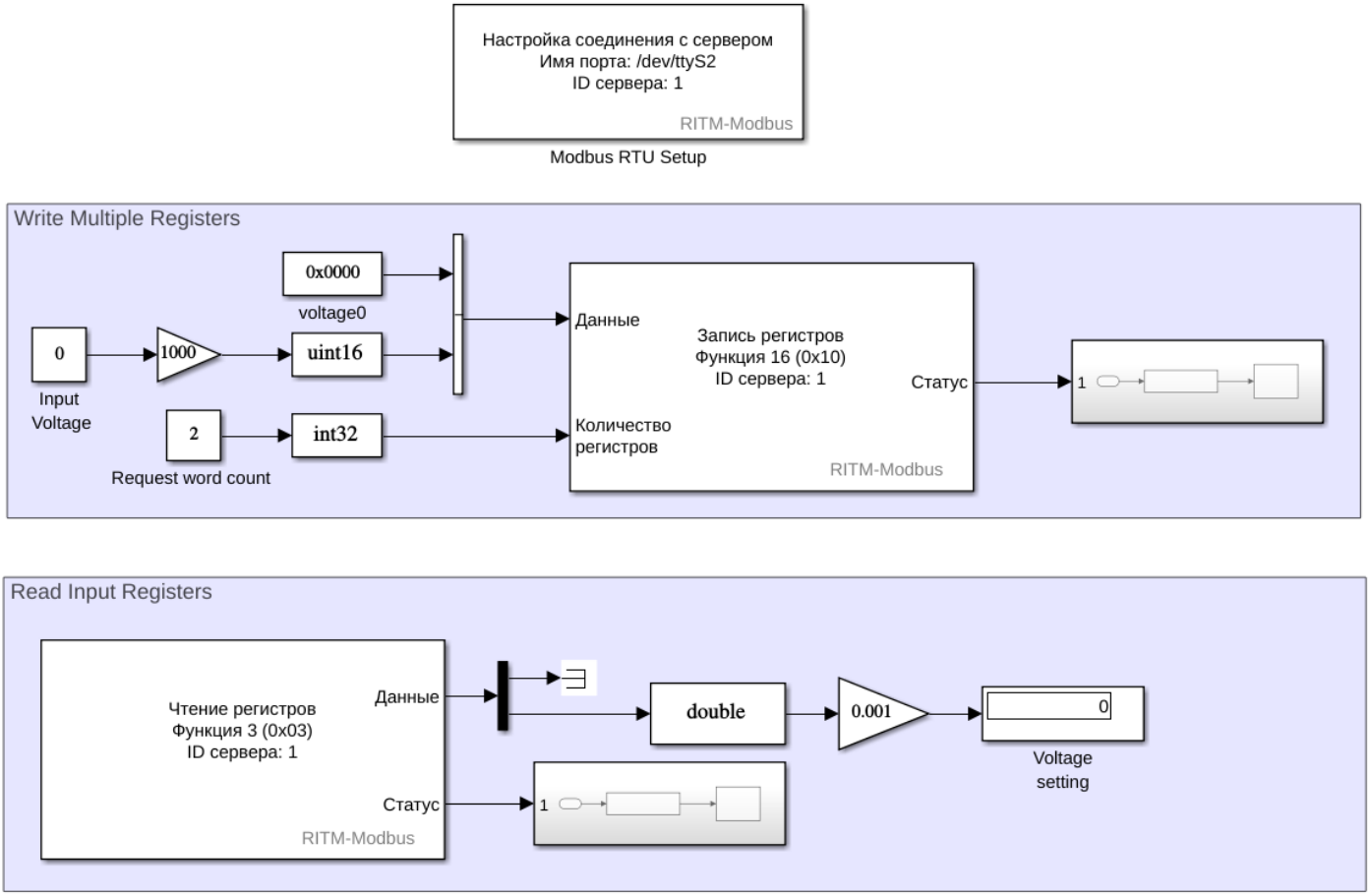
- Модель
- Моделирование

Для ознакомления с основными возможностями библиотеки в качестве слейв-устройства используется источник питания MAISHENG WSD-6050.

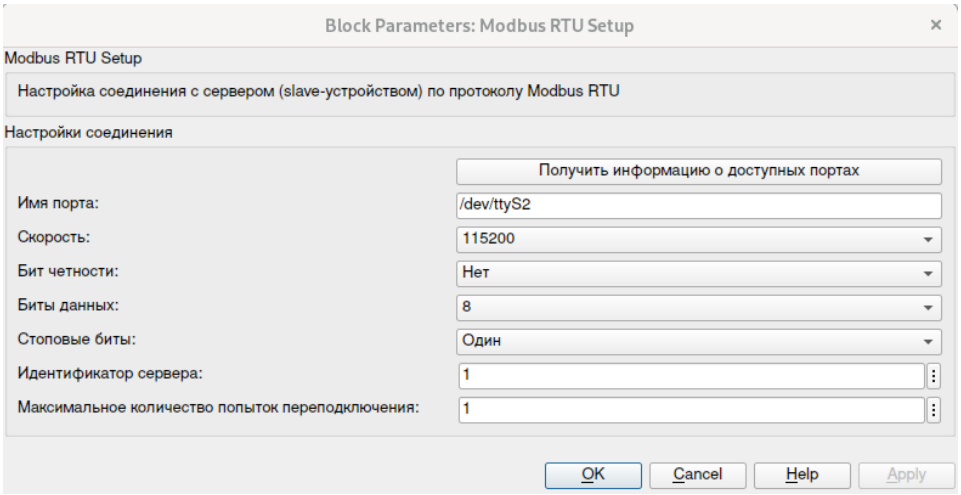
Модель

Рассмотрим модель:

```
open_system('ritmdemo_block_modbusRTU')
```

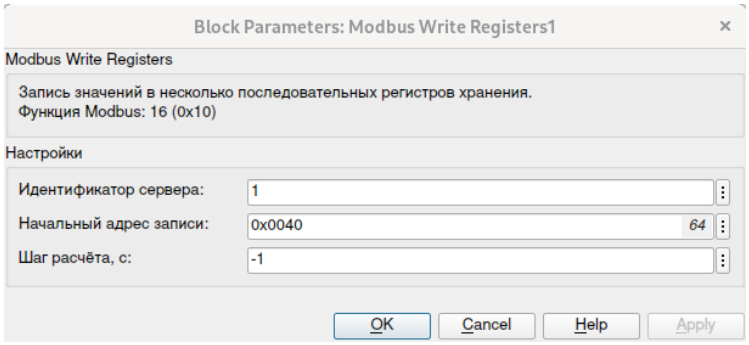


Соединение с сервером Modbus устанавливается при помощи блока Modbus RTU Setup. Его параметры следующие:

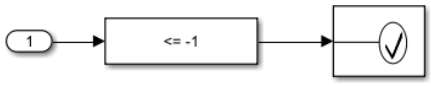


Из настроек видно, что устанавливается связь по порту ttyS2, задается скорость приемопередачи равная 115200 бод.

Бит четности, биты данных и стоповые биты задаются в соответствии с требованием слейв-устройства. Данному соединению присваивается внутренний идентификатор 1. В дальнейшем, данный идентификатор будет использоваться в блоках чтения и записи, например, в блоке **Modbus Write Registers**:



К примеру рассмотрим область Write Multiple Registers. В этой области, выход “Статус” блока чтения регистров подается на подсистему “Обработка ошибки соединения”.

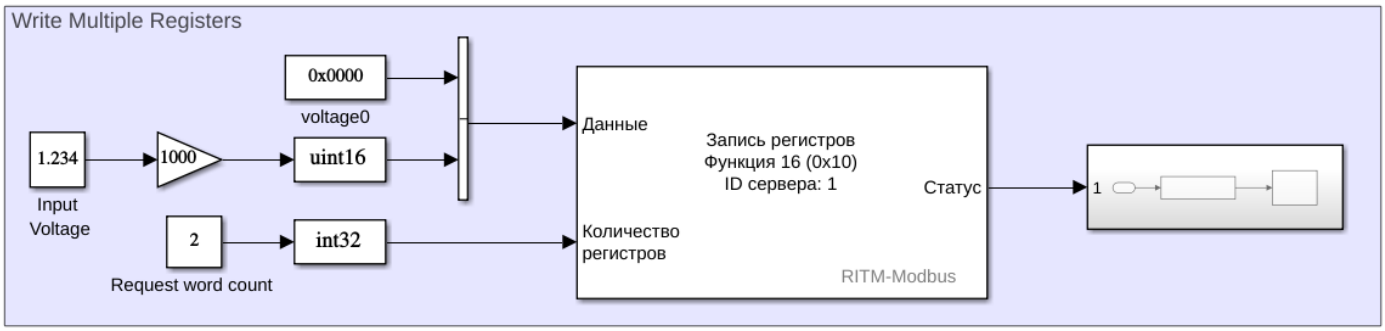


Назначение этой подсистемы - отслеживать статус операции. Если статус операции равен -1, то блок Assertion выдаст предупреждение. Применяя такую конструкцию моделирования, можно останавливать исполнение модели при длительном обрыве соединения.

Моделирование

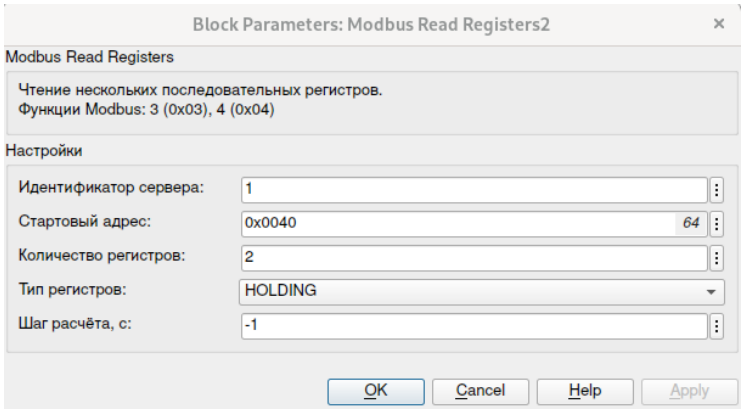
До начала записи каких-либо данных на слейв-устройство можно заметить, что его регистры, предназначенные для записи, не содержат никаких значений:

Для демонстрации записи данных на слейв-устройство установим значение желаемого выходного напряжения на блоке **Input Voltage**, которое будет передано в блок **Modbus Write Registers** в качестве данных для записи по адресу 0x0040:

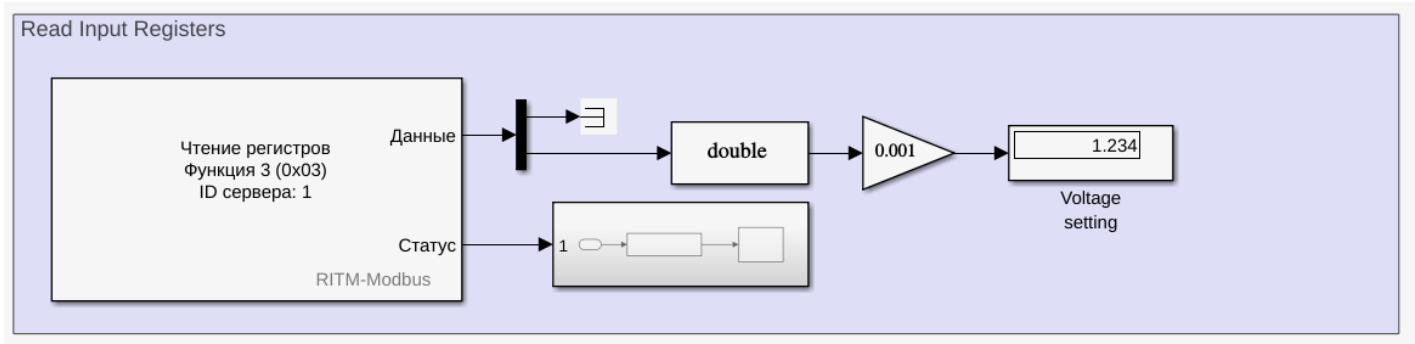


На экране слейв-устройства наблюдаем записанное значение напряжения:

Чтение данных со слейв-устройства происходит при помощи блока **Modbus Read Registers**. В параметрах блока указываем адрес, в котором находятся интересующие нас данные:



В модели на блоке **Voltage setting** наблюдаем считанное значение напряжения со слейв устройства:



Библиотека источников питания

Содержание

- Общие сведения
- Особенности работы
- Блоки для работы с источниками питания
- Руководство по использованию библиотеки

Общие сведения

Библиотека содержит блоки, предназначенные для дистанционного управления источниками питания.
Поддерживаемые источники питания: - MAISHENG WSD-6050 - OWON OWP3006H

Особенности работы

Для связи между источниками питания и КПМ РИТМ используется протокол Modbus RTU.

Блоки для работы с источниками питания

- Блок управления источником питания MAISHENG WSD-6050
- Блок управления источником питания OWON OWP3006H

Руководство по использованию библиотеки

Для начала работы с библиотекой рекомендуется ознакомиться с разделом документации для используемого источника питания.

- Работа с источником питания MAISHENG WSD-6050
- Работа с источником питания OWON OWP3006H

Блок управления источником питания MAISHENG WSD-6050

Содержание

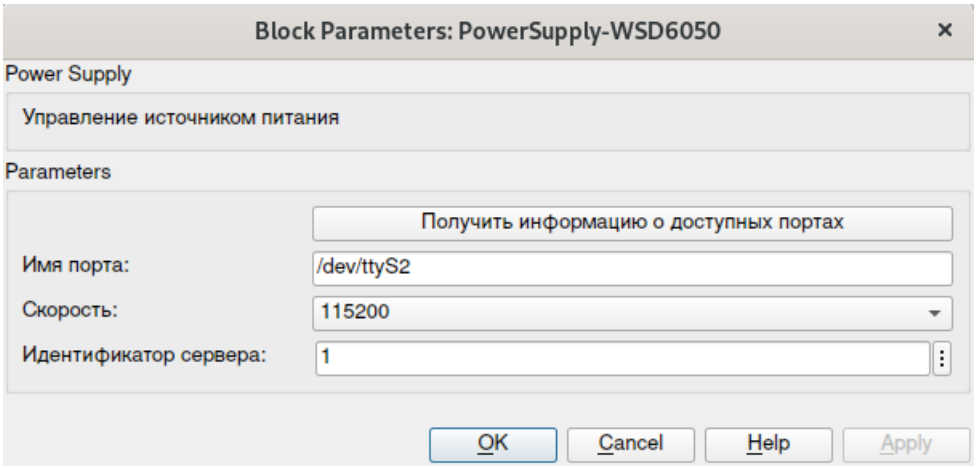
- Описание
- Маска блока
- Параметры блока
- Входы блока
- Выходы блока

Блок управления источником питания MAISHENG WSD6050.

Описание

Блок PowerSupply-WSD6050 предназначен для дистанционного управления источником питания MAISHENG WSD6050.

Маска блока



Параметры блока

- ***Имя порта***: Задает имя порта для настройки. Имена доступных портов могут быть получены во всплывающем окне при нажатии на кнопку “Получить информацию о доступных портах” в маске блока.
- ***Скорость***: Задает скорость приемопередачи. Возможные варианты: *9600*, *19200*, *38400*, *57600*, *115200* бод.
- ***Идентификатор сервера***: Идентификатор соединения. Данный идентификатор используется в блоках приема и передачи. Допустимые значения: от *0* до *15*.

Входы блока

- ***Вкл/Выкл***: Для подачи установленных значений тока и напряжения на выходные клеммы источника питания необходимо на вход блока подать *1*, а для прекращения подачи подать *0*.
- ***Напряжение***: На вход блока подается значение типа *double*, которое интерпретируется как значение напряжения с шагом *0.001 В*.
- ***Ток***: На вход блока подается значение типа *double*, которое интерпретируется как значение тока с шагом *0.01 А*.

Выходы блока

- ***Напряжение***: Текущее значение напряжения типа *double* на выходных клеммах источника питания.
- ***Ток***: Текущее значение тока типа *double* на выходных клеммах источника питания.

Работа с блоком управления источника питания MAISHENG WSD-6050

Содержание

- Включение дистанционного управления
- Демонстрация работы
- Моделирование

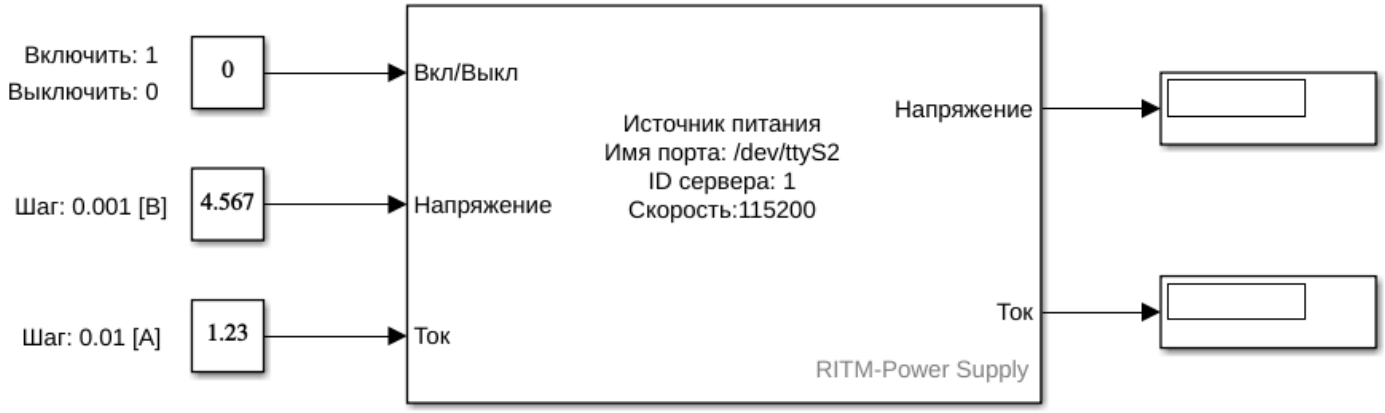
Включение дистанционного управления

По умолчанию источник питания работает в локальном режиме. Для перевода управления в дистанционный режим необходимо на источнике питания нажать кнопку “L/R” и убедиться, что режим управления сменился с Lo на Re.

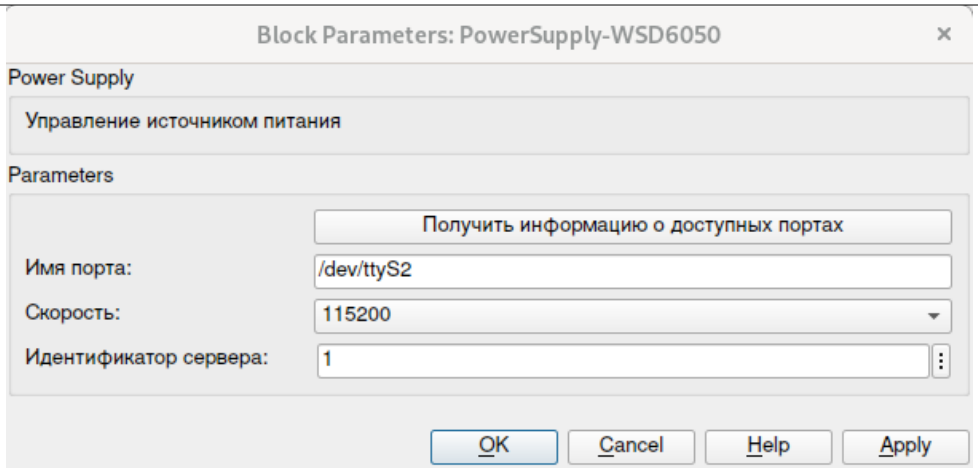
Демонстрация работы

Рассмотрим модель:

```
open_system('ritmdemo_block_powersupply')
```



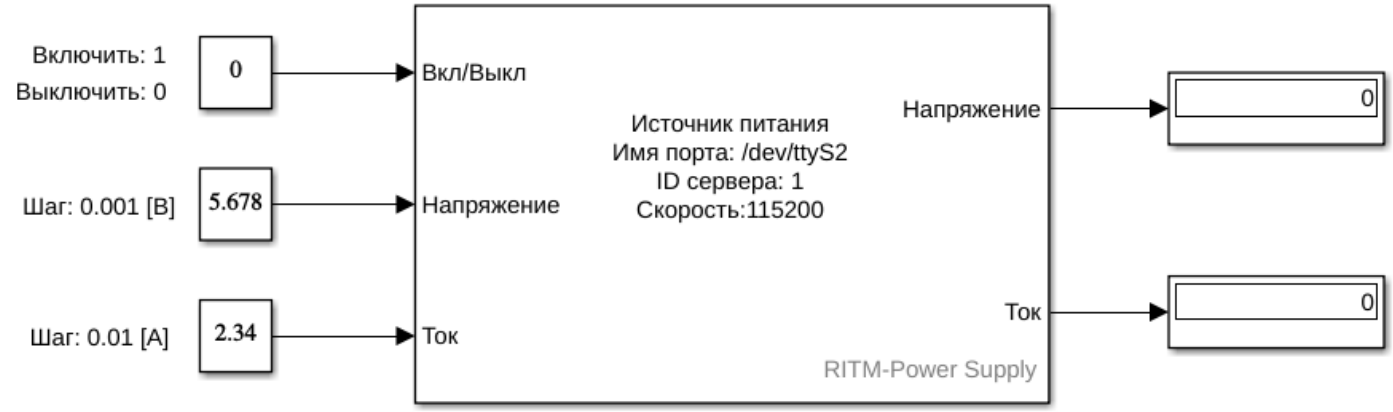
Параметры соединения с источником питания задаются из маски блока. Его параметры следующие:



Из настроек видно, что устанавливается связь по порту ttyS2, задается скорость приемопередачи равная 115200 бод. Данному соединению присваивается внутренний идентификатор 1. Идентификатор должен быть уникальным для каждого блока истоничка питания и блоков настройки соединения, использующих протокол Modbus.

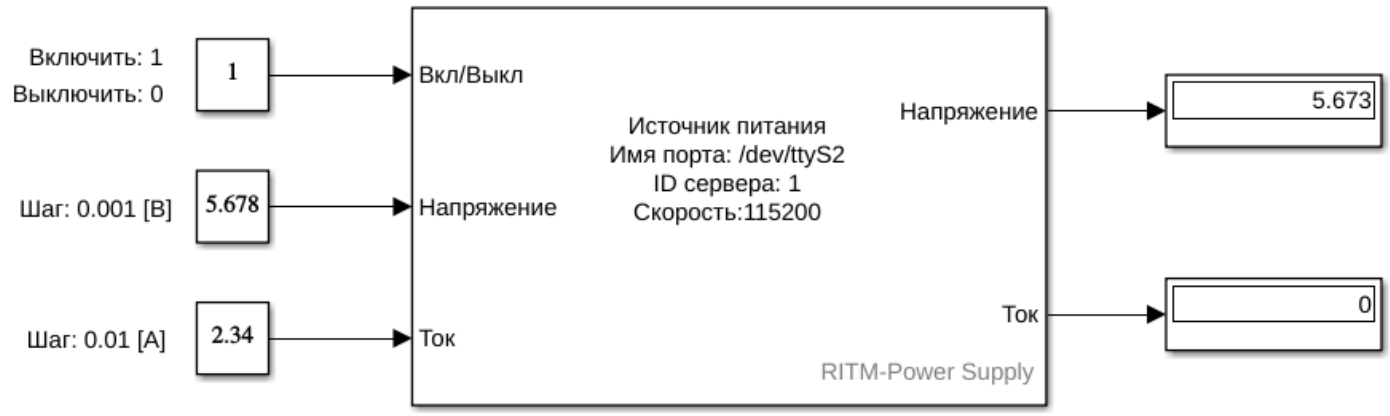
Моделирование

Для демонстрации работы установим значения напряжения и тока на входах Напряжение и Ток, которые хотим получить на выходных клеммах источника питания:



На экране источника питания наблюдаем установленные значения напряжения и тока.

Для подачи напряжения на выходные клеммы источника питания, необходимо установить на входе Вкл/Выкл блока значение равное 1:



На экране источника питания и на выходах Напряжение и Ток блока наблюдаем текущие значения напряжения и тока на выходных клеммах источника питания.

Блок управления источником питания OWON OWP3006H

Содержание

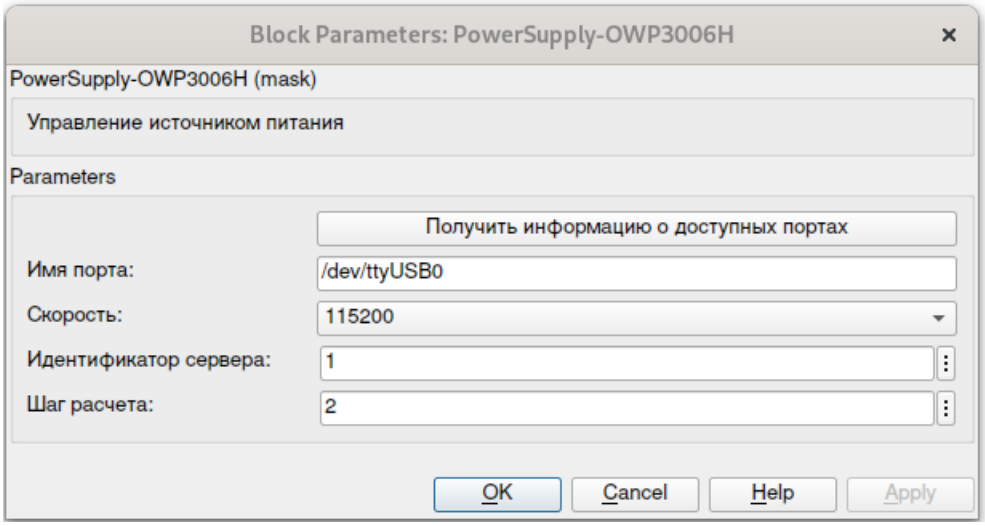
- Описание
- Маска блока
- Параметры блока
- Входы блока
- Выходы блока

Блок управления источником питания OWON OWP3006H.

Описание

Блок PowerSupply-OWP3006H предназначен для дистанционного управления источником питания OWON OWP3006H.

Маска блока



Параметры блока

- ***Имя порта***: Задает имя порта для настройки. Имена доступных портов могут быть получены во всплывающем окне при нажатии на кнопку “Получить информацию о доступных портах” в маске блока.
- ***Скорость***: Задает скорость передачи. Возможные варианты: *9600*, *19200*, *38400*, *57600*, *115200* бод.
- ***Идентификатор сервера***: Идентификатор сервера протокола Modbus. Данный идентификатор должен быть уникальным для каждого блока источника питания и блоков настройки соединения, использующих протокол Modbus. Допустимые значения: от *0* до *15*.
- ***Шаг расчёта, с***: Позволяет задать шаг расчёта в секундах для получения данных в приложении реального времени. Допустимы значения от *1* секунды.

Входы блока

- ***Вкл/Выкл***: Для подачи установленных значений тока и напряжения на выходные клеммы источника питания необходимо на вход блока подать *1*, а для прекращения подачи подать *0*.
- ***Напряжение***: На вход блока подается значение типа *double*, которое интерпретируется как значение напряжения с шагом *0.01 В*.
- ***Ток***: На вход блока подается значение типа *double*, которое интерпретируется как значение тока с шагом *0.01 А*.

Выходы блока

- ***Напряжение***: Текущее значение напряжения типа *double* на выходных клеммах источника питания.

Работа с блоком управления источника питания OWON OWP3006H

Содержание

- Включение дистанционного управления
- Демонстрация работы
- Моделирование

Включение дистанционного управления

По умолчанию источник питания работает в дистанционном режиме со скоростью передачи 115200 бод. Режим работы источника питания отображается в правом нижнем углу на экране устройства.

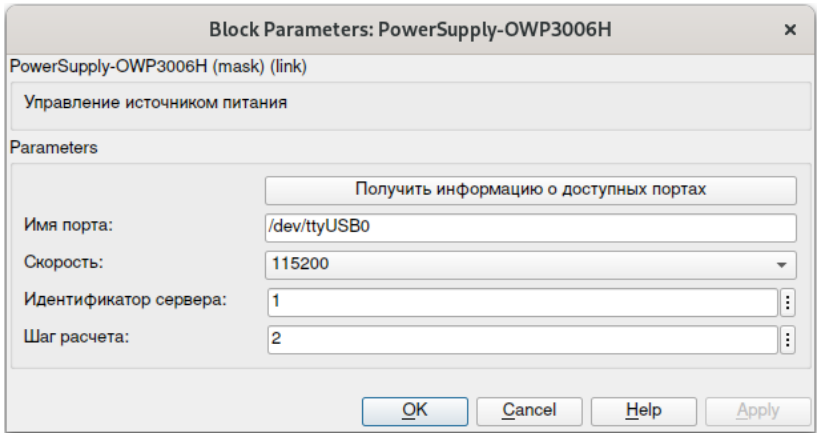
Надпись **Remo** означает, что блок работает в режиме дистанционного управления.

Демонстрация работы

Рассмотрим модель:

```
open_system('ritmdemo_block_powersupply_owon')
```

Параметры соединения с источником питания задаются из маски блока.

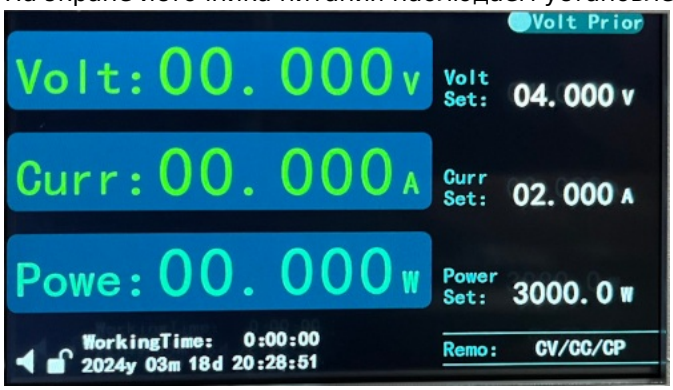


Из настроек видно, что устанавливается связь по порту ttyUSB0, задается скорость передачи равная 115200 бод. Данному соединению присваивается внутренний идентификатор 1. Идентификатор должен быть уникальным для каждого блока источника питания и блоков настройки соединения, использующих протокол Modbus.

Моделирование

Для демонстрации работы установим значения напряжения и тока на входах Напряжение и Ток, которые хотим получить на выходных клеммах источника питания:

На экране источника питания наблюдаем установленные значения напряжения и тока.



Для подачи напряжения на выходные клеммы источника питания, необходимо установить на входе Вкл/Выкл блока значение равное 1:

На экране источника питания и на выходе Напряжение блока наблюдаем текущее значение напряжения, которое равно напряжению на выходных клеммах источника питания.

Volt Prior
Volt: 04.001 V
Curr: 01.921 A
Powe: 07.685 W
WorkingTime: 0:01:07
2024y 03m 18d 20:27:28
Remo: CV

Библиотека контроллеров управления

Содержание

- Блок для работы с рулевым управлением Logitech G29
- Блоки для работы джойстиком и блоком РУД Thrustmaster Hotas Warthog

Блок для работы с рулевым управлением Logitech G29

Блок предназначени для получения состояния рулевого управления от контроллера Logitech G29.

Блоки для работы джойстиком и блоком РУД Thrustmaster Hotas Warthog

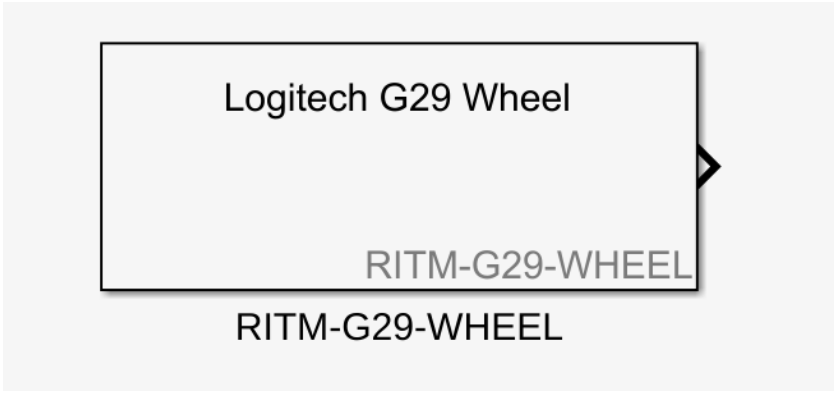
Блок предназначени для получения состояния РУД и джойстика.

Блок для работы с рулевым управлением Logitech G29

Содержание

- Библиотека блоков
- Блок RITM G29 WHEEL
  - Маска блока
  - Выход блока
  - Параметры блока
  - Пример работы блока

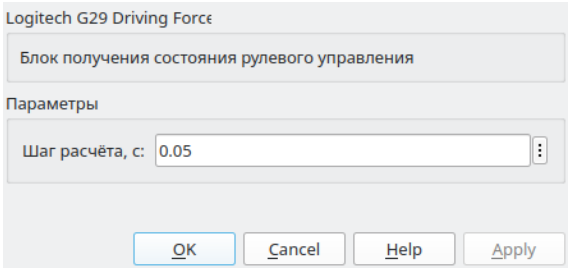
Библиотека блоков



Блок RITM G29 WHEEL

Блок предназначени для получения состояния рулевого управления от контроллера Logitech G29.

Маска блока



Выход блока

Выход блока **RITM G29 WHEEL** является шиной типа **RITM_LOG_G29_STATE**:

Описание шины RITM_LOG_G29_STATE:
Шина содержит в себе набор сигналов описывающих текущее состояние контроллера рулевого управления:

- **stearing** - тип данных **double**, диапазон - от -450 до +450. Угол поворота рулевого колеса в градусах.
- **pedal_accelerator** - тип данных **double**, диапазон - от 0 до 100. Процент нажатия на педаль газа.
- **pedal_brake** - тип данных **double**, диапазон - от 0 до 100. Процент нажатия на педаль тормоза.
- **pedal_clutch** - тип данных **double**, диапазон - от 0 до 100. Процент нажатия на педаль сцепления.

Сигналы состояния кнопок: **false** - кнопка не нажата, **true** - кнопка нажата.

- **button_cross** - тип данных **boolean**. Состояние кнопки **X**.
- **button_square** - тип данных **boolean**. Состояние кнопки ■.
- **button_triangle** - тип данных **boolean**. Состояние кнопки △.
- **button_circle** - тип данных **boolean**. Состояние кнопки ◯.
- **button_up** - тип данных **boolean**. Состояние кнопки ↑.
- **button_left** - тип данных **boolean**. Состояние кнопки ←.

- **button_right** - тип данных **boolean**. Состояние кнопки →.
- **button_down** - тип данных **boolean**. Состояние кнопки ↓.
- **button_plus** - тип данных **boolean**. Состояние кнопки **+**.
- **button_minus** - тип данных **boolean**. Состояние кнопки **-**.
- **button_l2** - тип данных **boolean**. Состояние кнопки **L2**.
- **button_l3** - тип данных **boolean**. Состояние кнопки **L3**.
- **button_r2** - тип данных **boolean**. Состояние кнопки **R2**.
- **button_r3** - тип данных **boolean**. Состояние кнопки **R3**.
- **shifter_left** - тип данных **boolean**. Состояние левого подрулевого лепестка.
- **shifter_right** - тип данных **boolean**. Состояние правого подрулевого лепестка.
- **button_enter** - тип данных **boolean**. Состояние кнопки ↵.

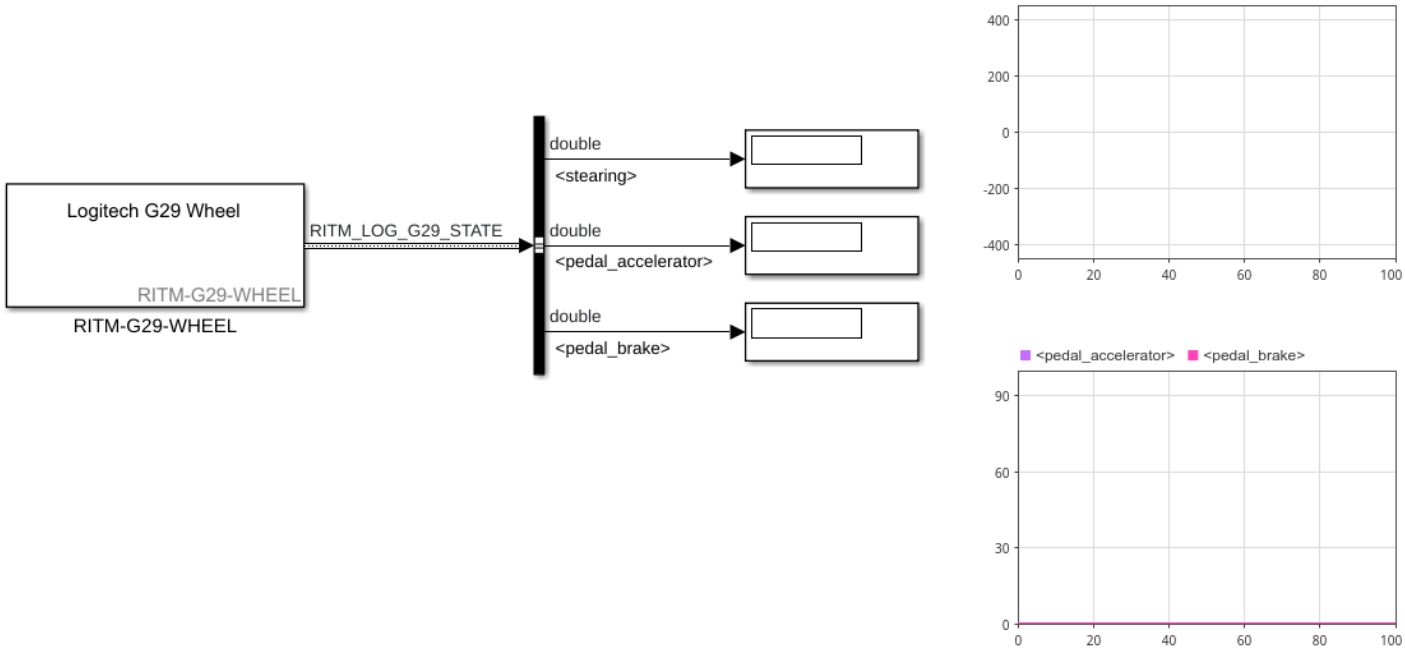
Параметры блока

- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах, с которым будет получаться значение текущего состояния рулевого управления.

Пример работы блока

Пример использования блока RITM G29 WHEEL.

```
open_system('ritmdemo_block_g29_wheel.slx')
```

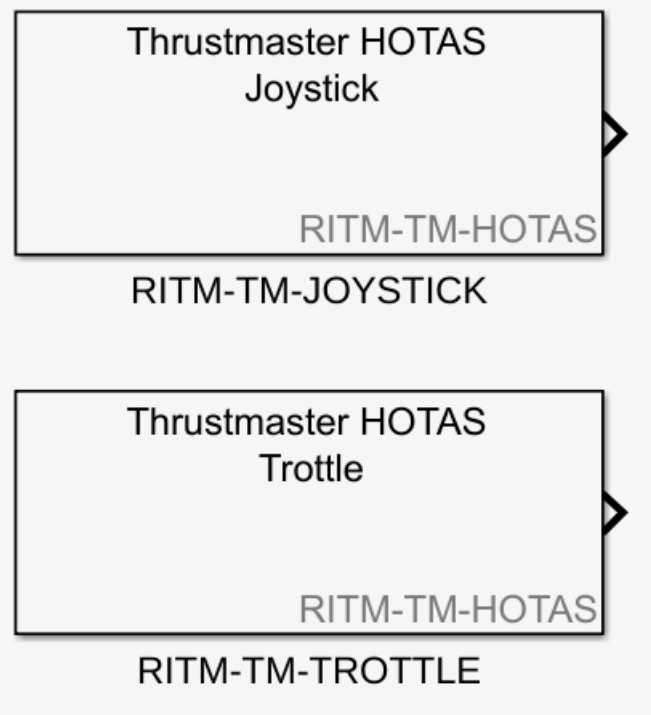


Блоки для работы джойстиком и блоком РУД Thrustmaster Hotas Warthog

Содержание

- Библиотека блоков
- Блок RITM TM JOYSTICK
  - Маска блока
  - Выход блока
  - Параметры блока
- Блок RITM TM TROTTLE
  - Маска блока
  - Выход блока
  - Параметры блока
  - Пример работы блока

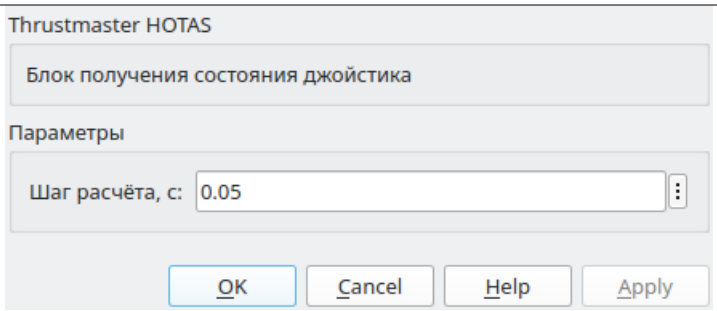
Библиотека блоков



Блок RITM TM JOYSTICK

Блок предназначени для получения состояния джойстика.

Маска блока



Выход блока

Выход блока **RITM TM JOYSTICK** является шиной типа **RITM_TM_JOYSTICK_STATE**:

Описание шины RITM_TM_JOYSTICK_STATE:
Шина содержит в себе набор сигналов описывающих текущее состояние джойстика:

- **pitch** - тип данных **double**, диапазон - от -1 до +1. Ось тангажа.
- **roll** - тип данных **double**, диапазон - от 0 до 100. Ось крена.

Сигналы состояния кнопок: **false** - кнопка не нажата, **true** - кнопка нажата.

- **WeaponsRelease** - тип данных **boolean**.
- **GunTrigger** - тип данных **boolean**.

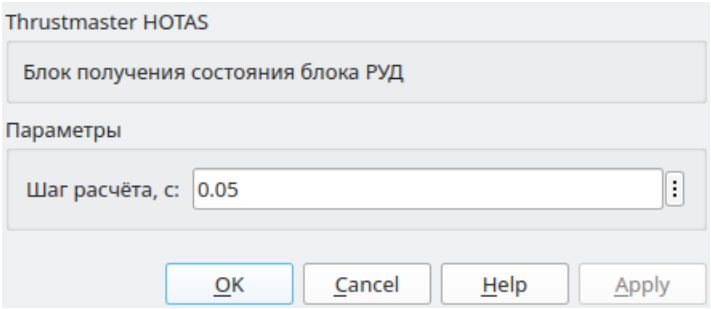
Параметры блока

- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах, с которым будет получаться значение текущего состояния рулевого управления.

Блок RITM TM TROTTLE

Блок предназначени для получения состояния блока РУД.

Маска блока



Выход блока

Выход блока **RITM TM TROTTLE** является шиной типа **RITM_TM_TROTTLE_STATE**:

Описание шины RITM_TM_TROTTLE_STATE:
Шина содержит в себе набор сигналов описывающих текущее состояние блока РУД:

- **TrottleRight** - тип данных **double**, диапазон - от 0 до +1. Положение правого РУД, 1 - полный газ, 0 - выключен.
- **TrottleLeft** - тип данных **double**, диапазон - от 0 до +1. Положение левого РУД, 1 -

полный газ, 0 - выключен

Сигналы состояния кнопок: **false** - кнопка не нажата, **true** - кнопка нажата.

- **EAC** - тип данных **boolean**.
- **RDR_ALTM** - тип данных **boolean**.
- **Autopilot_Engage** - тип данных **boolean**.

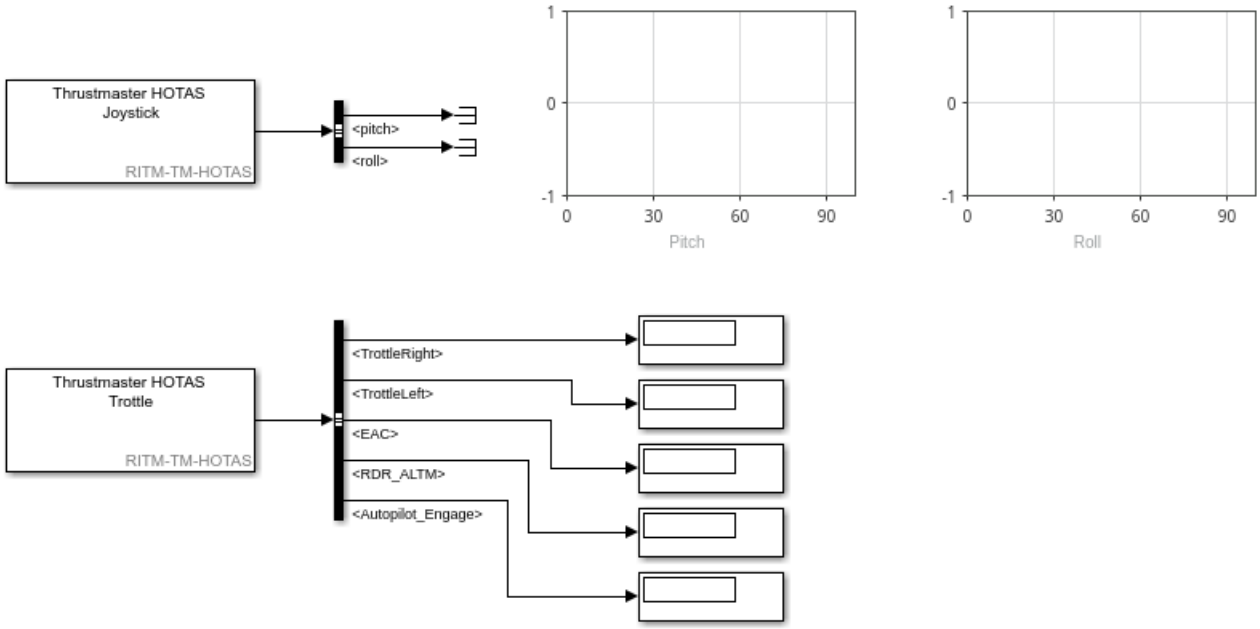
Параметры блока

- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах, с которым будет получаться значение текущего состояния рулевого управления.

Пример работы блока

Пример использования блоков.

```
open_system('ritmdemo_block_tm_joystick.slx')
```



Модули поддержки MIL-STD-1553

Содержание

- Модуль RITMeX MI-EL-45

Модуль RITMeX MI-EL-45

Модуль с четырьмя резервированными мультиплексными каналами по ГОСТ Р 52070-2003 (ГОСТ 26765.52-87, MIL-STD-1553B). Поддерживает работу в режимах контроллера канала, оконечного устройства и монитора канала.

Модули RITMeX MI-EL-X5

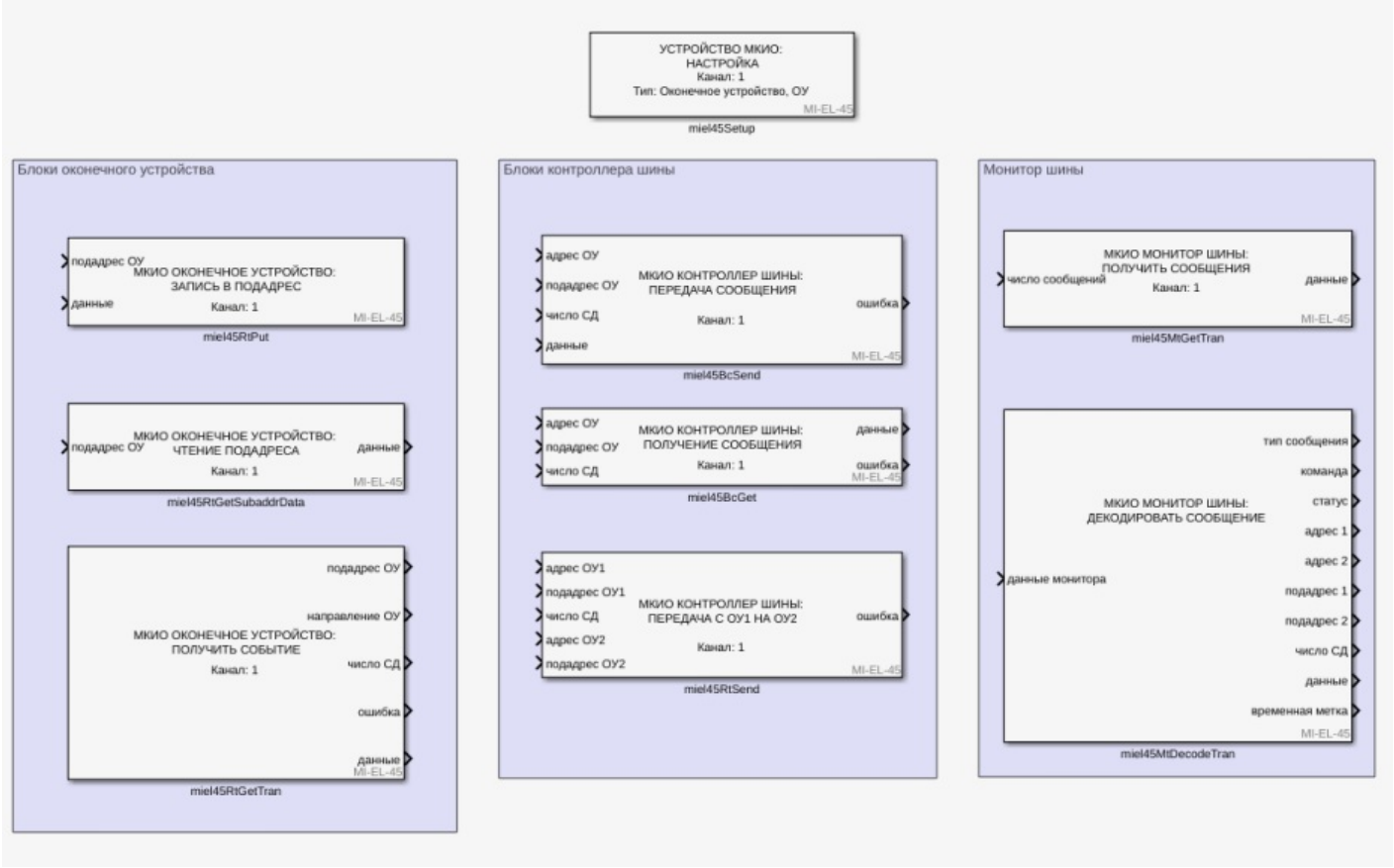
Содержание

- Описание
- Библиотека блоков
- Настройка канала
  - Блок “Устройство МКИО: настройка” в Simulink
- Контроллер шины
  - Блок “МКИО контроллер шины: передача сообщения” в Simulink
  - Блок “МКИО контроллер шины: получение сообщения” в Simulink
  - Блок “МКИО контроллер шины: передача сообщения с ОУ1 на ОУ2” в Simulink
  - Расшифровка статуса ошибки в блоках контроллера шины
- Оконечное устройство
  - Блок “МКИО оконечное устройство: запись в подадрес” в Simulink
  - Блок “МКИО оконечное устройство: чтение из подадреса” в Simulink
  - Блок “МКИО оконечное устройство: получить событие” в Simulink
- Монитор шины
  - Блок “МКИО монитор шины: получить сообщения” в Simulink
  - Блок “МКИО монитор шины: декодировать сообщения” в Simulink

Описание

Модули MI-EL-45 и MI-EL-55 с четырьмя резервированными мультиплексными каналами по ГОСТ Р 52070-2003 (ГОСТ 26765.52-87, MIL-STD-1553B). Поддерживает работу в режимах контроллера шины, оконечного устройства и монитора шины в КПМ “РИТМ”.

Библиотека блоков

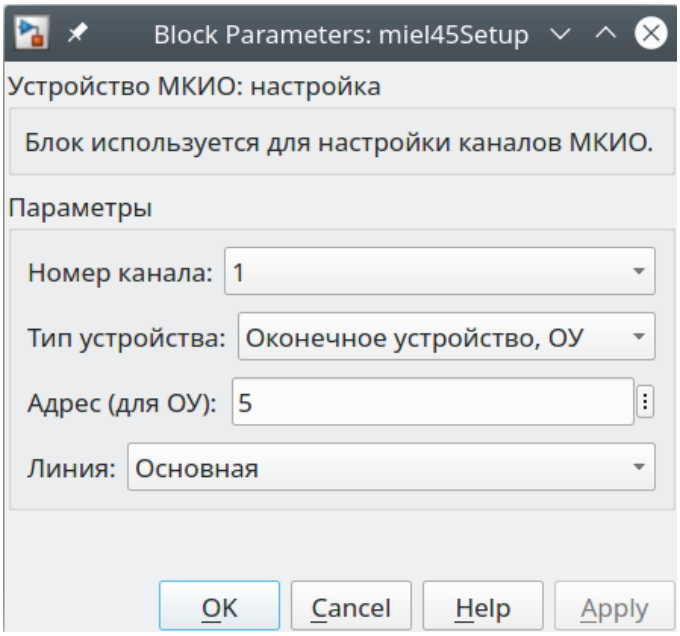


Настройка канала

Блок “Устройство МКИО: настройка” в Simulink

Блок используется для настройки каналов МКИО.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Тип устройства**: Тип устройства на шине: **Контроллер шины (КШ)**, **Оконечное**

устройство (ОУ) или **Монитор шины (МШ)**.

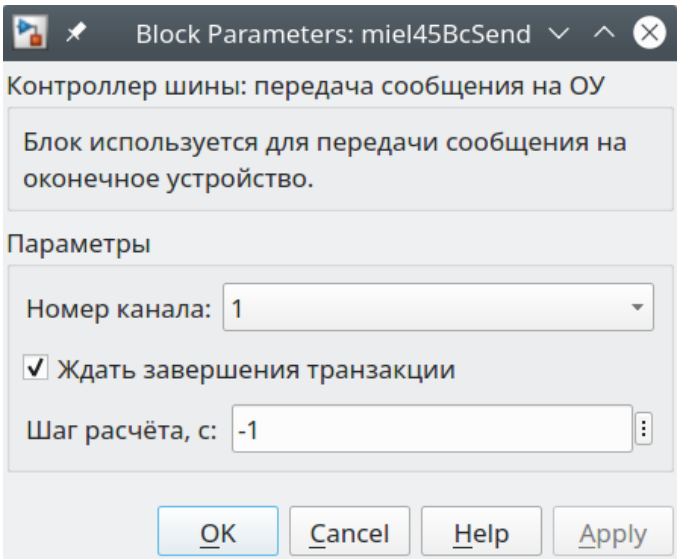
- **Адрес (для ОУ)** : Адрес оконечного устройства. Параметр становится доступным, если выбран параметр **Тип устройства** как **Оконечное устройство (ОУ)**.
- **Линия**: Линия передачи данных: **Основная** или **Резервная**.

Контроллер шины

Блок “МКИО контроллер шины: передача сообщения” в Simulink

Блок используется для передачи сообщения на оконечное устройство.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Ждать завершения транзакции**: С этой опцией контроллер шины ожидает завершения обмена в линии. Внимание: это может привести к Переполнению шага расчета. При отключенной опции разработчик модели должен сам контролировать возможные конфликты транзакций в линии и, если требуется, увеличить шаг расчёта блока.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Входы блока

- **Адрес ОУ**: Адрес оконечного устройства, на которое будет передано сообщение. Адрес 31 - широковещательный.
- **Подадрес ОУ**: Подадрес оконечного устройства.
- **Число СД**: Число слов данных для передачи.
- **Данные**: Массив из 32 элементов типа **uint16** - информационные слова для передачи. В линию передаются только **Число СД** элементов этого массива.

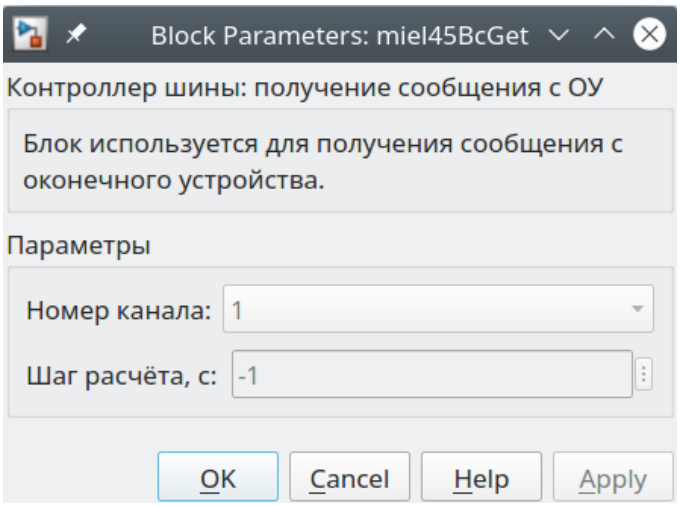
Выходы блока

- **Ошибка**: Флаг ошибки при обмене в канале. В случае, если контроллер шины зарегистрировал ошибку при обмене, будет установлено ненулевое значение.

Блок “МКИО контроллер шины: получение сообщения” в Simulink

Блок используется для получения сообщения с оконечного устройства.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Входы блока

- **Адрес ОУ**: Адрес оконечного устройства, с которого будет запрошено сообщение.
- **Подадрес ОУ**: Подадрес оконечного устройства.
- **Число СД**: Число слов данных для приёма.

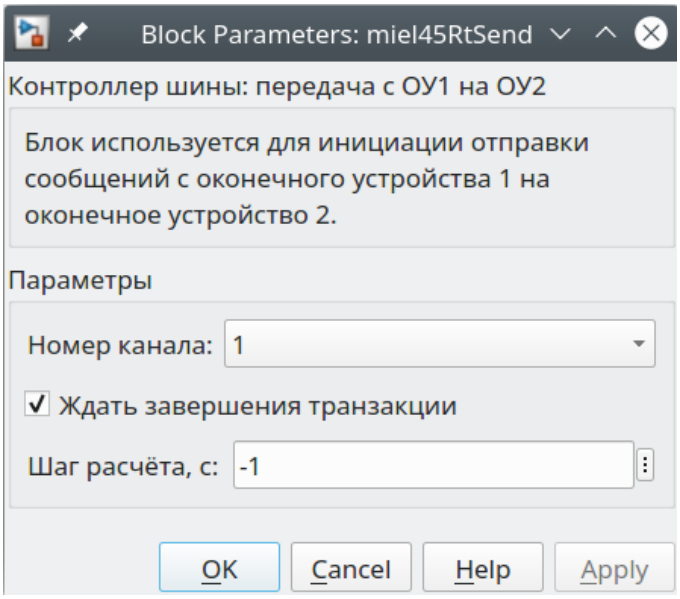
Выходы блока

- **Ошибка**: Флаг ошибки при обмене в канале. В случае, если контроллер шины зарегистрировал ошибку при обмене, будет установлено ненулевое значение.
- **Данные**: Массив из 32 элементов типа **uint16** - принятые информационные слова.

Блок “МКИО контроллер шины: передача сообщения с ОУ1 на ОУ2” в Simulink

Блок используется для инициации отправки сообщений с оконечного устройства 1 на оконечное устройство 2.

Маска блока



Параметры блока

- **Номер канала**: Номер канала контроллера шины, инициирующего транзакцию. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Ждать завершения транзакции**: С этой опцией контроллер шины ожидает завершения обмена в линии. Внимание: это может привести к Переполнению шага расчета. При отключенной опции разработчик модели должен сам контролировать возможные конфликты транзакций в линии и, если требуется, увеличить шаг расчёта блока.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Входы блока

- **Адрес ОУ1**: Адрес оконечного устройства, с которого будет производиться отправка сообщения.
- **Подадрес ОУ1**: Подадрес оконечного устройства, с которого будет производиться отправка сообщения.
- **Число СД**: Число слов данных для передачи.
- **Адрес ОУ2**: Адрес оконечного устройства, на которое будет производиться отправка сообщения. Адрес 31 - широковещательный.
- **Подадрес ОУ2**: Подадрес оконечного устройства, на которое будет производиться отправка сообщения.

Выходы блока

- **Ошибка**: Флаг ошибки при обмене в канале. В случае, если контроллер шины зарегистрировал ошибку при обмене, будет установлено ненулевое значение.

Расшифровка статуса ошибки в блоках контроллера шины

Выход **Ошибка** имеет тип данных **uint32**. В зависимости от ситуации, задетектированной на шине, в нем выставляются следующие биты:

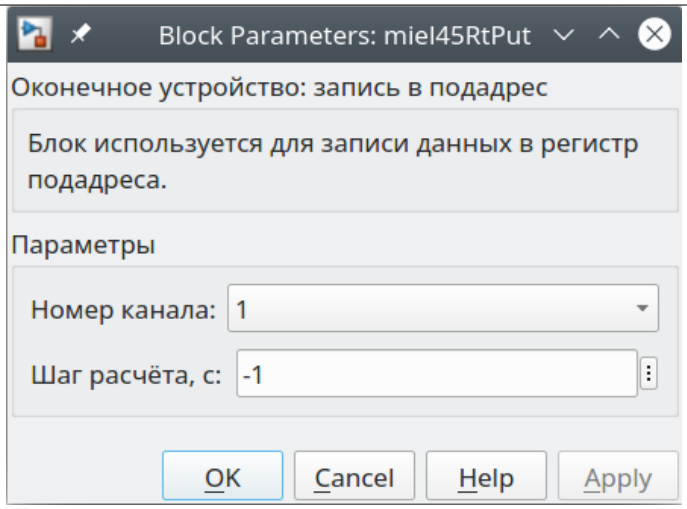
- **ERAO_ERROR**: Ошибка адреса ОУ в ОС. Маска **0x1**.
- **MEO_ERROR**: Ошибка кода “Манчестер-2”. Маска **0x2**.
- **TO_ERROR**: Таймаут при приёме. Маска **0x4**.
- **EBC_ERROR**: Ошибка самоконтроля при передаче или составная ошибка при приёме. Маска **0x8**.
- **RTFL_ERROR**: Бит ОС “Неисправность ОУ”. Маска **0x10**.
- **DNBA**: Бит ОС “Принято управление интерфейсом”. Маска **0x20**.
- **SSFL_ERROR**: Бит ОС “Неисправность подсистемы”. Маска **0x40**.
- **BUSY**: Бит ОС “Подсистема занята”. Маска **0x80**.
- **RT_ERROR**: Таймаут при приёме. Маска **0x100**.
- **HW_ERROR**: Локальная аппаратная ошибка. Маска **0x200**.

Оконечное устройство

Блок “МКИО оконечное устройство: запись в подадрес” в Simulink

Блок используется для записи данных в регистр подадреса.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

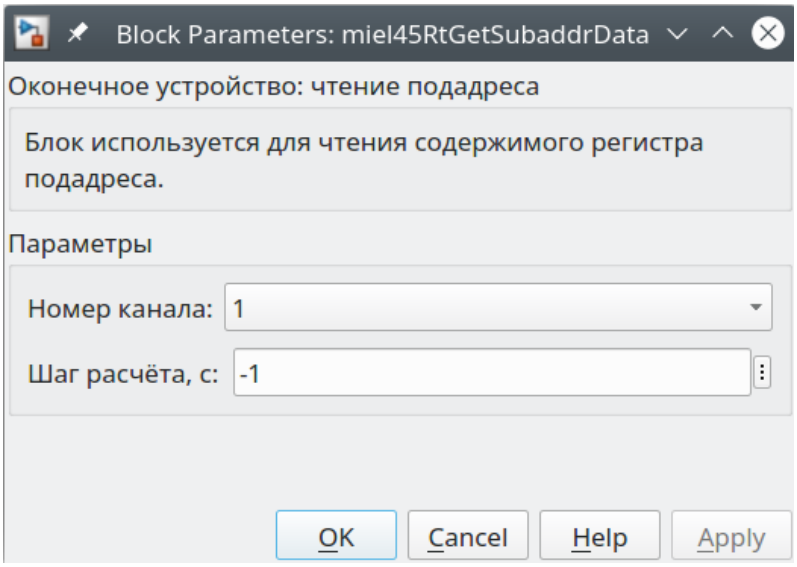
Входы блока

- **Подадрес ОУ**: Подадрес оконечного устройства для записи данных.
- **Данные**: Массив из 32 элементов типа **uint16** для записи в подадрес оконечного устройства.

Блок “МКИО оконечное устройство: чтение из подадреса” в Simulink

Блок используется для чтения содержимого регистра подадреса.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Входы блока

- **Подадрес ОУ**: Подадрес оконечного устройства, из которого будут считываться данные.

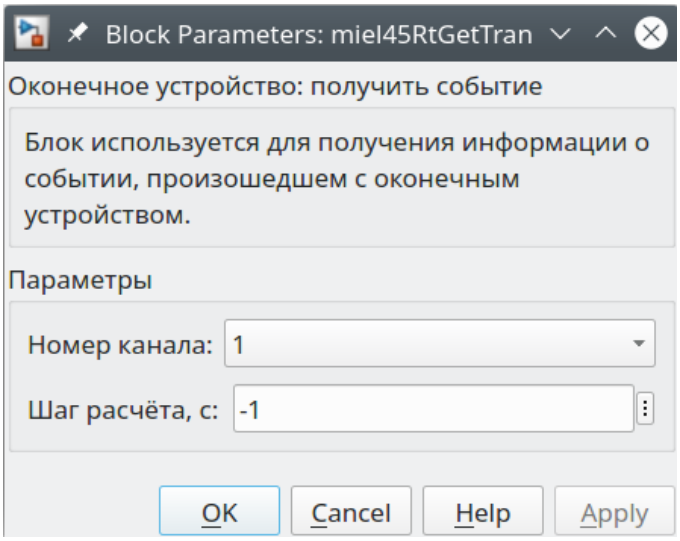
Выходы блока

- **Данные**: Массив из 32 элементов типа **uint16**, считанных из подадреса оконечного устройства.

Блок “МКИО оконечное устройство: получить событие” в Simulink

Блок используется для получения информации о событии, произошедшем с оконечным устройством. Модуль имеет встроенный **FIFO** буфер на 512 событий, которые могут быть получены в модели. Если неполученных событий нет, все выходы блока вернут нулевые значения.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Выходы блока

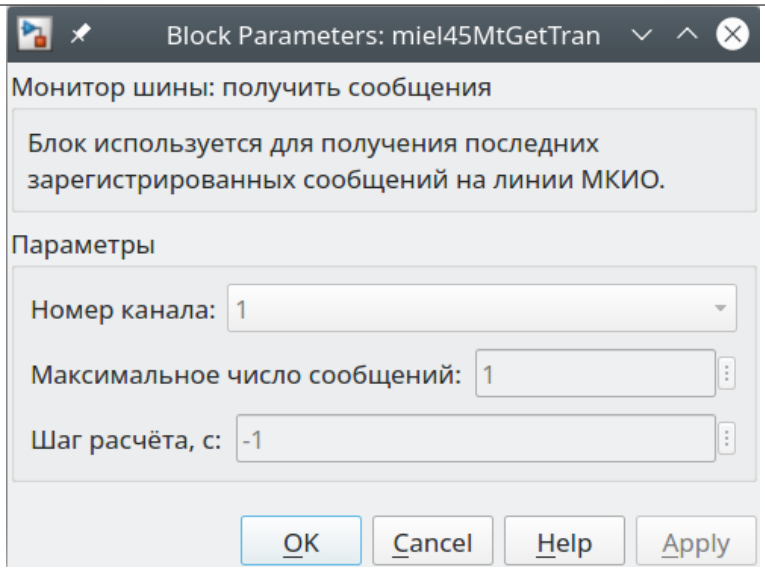
- **Подадрес ОУ**: Подадрес оконечного устройства, который участвовал в транзакции.
- **Направление**: Направление передачи данных. Чтение с оконечного устройства - **1**, запись в оконечное устройство - **2**.
- **Число СД**: Количество прочитанных или записанных слов данных.
- **Ошибка**: Флаг ошибки при совершении транзакции. Если оконечное устройство зарегистрировало ошибку, то будет выставлено ненулевое значение.
- **Данные**: Записанные на оконечное устройство информационные слова во время последней операции записи.

Монитор шины

Блок “МКИО монитор шины: получить сообщения” в Simulink

Блок используется для получения последних зарегистрированных сообщений на линии МКИО.

Маска блока



Параметры блока

- **Номер канала**: Номер канала МКИО. Если в машине установлено больше одного модуля, то применяется сквозная нумерация каналов.
- **Максимальное число сообщений**: Наибольшее число последних сообщений, которые могут быть считаны за раз (не более 512).
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах.

Входы блока

Число сообщений: Число сообщений, которые надо считать из памяти монитора шины на этом шаге расчёта.

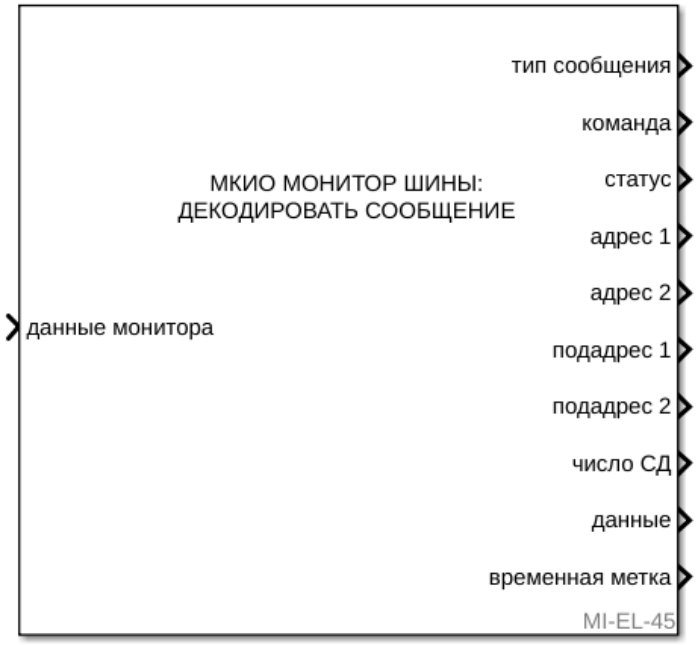
Выходы блока

Данные: Матрица из элементов типа **uint16** размерностью **[39 × Максимальное число сообщений]**. Количество столбцов матрицы соответствует максимальному числу сообщений. Каждая транзакция содержит 39 элементов и транзакции расположены по столбцам матрицы. Первые 36 элементов столбца содержат зарегистрированные на линии информационные слова и слова данных, относящиеся к этому сообщению. Элементы 37 - 39 содержат служебную информацию: слово состояния и временную метку. Столбцы матрицы за пределами **Числа сообщений** содержат неопределенные данные.

Блок “МКИО монитор шины: декодировать сообщения” в Simulink

Блок используется для декодирования сырых данных, полученных от блока **“МКИО монитор шины: получить сообщения”**.

Вид блока



Входы блока

Данные: Массив из 39 элементов. Один столбец матрицы, полученной от блока **“МКИО монитор шины: получить сообщения”**.

Выходы блока

- **Тип сообщения**: Шина из 10 сигналов. Значение сигнала **1** в шине означает, что сообщение имеет соответствующий тип.
  - **DATA_BC_RT**: передача данных КК-ОУ (формат 1).
  - **DATA_BC_RT_BRCST**: передача данных КК-ОУ в групповом режиме (формат 7).
  - **DATA_RT_BC**: передача данных ОУ-КК (формат 2).
  - **DATA_RT_RT**: передача данных ОУ-ОУ (формат 3).
  - **DATA_RT_RT_BRCST**: передача данных ОУ-ОУ в групповом режиме (формат 8).
  - **CTRL_C_A**: команда управления формата КС-ОС (формат 4).
  - **CTRL_C_BRCST**: команда управления формата КС в групповом режиме (формат 9).
  - **CTRL_CD_A**: команда управления формата КС+ИС-ОС (формат 6).
  - **CTRL_CD_BRCST**: команда управления формата КС+ИС в групповом режиме (формат 10).
  - **CTRL_C_AD**: команда управления формата КС-ОС+ИС (формат 5).
- **Команда**: Шина из 17 сигналов. Значение сигнала **1** в шине означает, что сообщение содержит соответствующую команду.
  - **DYNAMIC_BUS_CONTROL**: динамическое управление.
  - **SYNCHRONIZE**: синхронизация.
  - **TRANSMIT_STATUS_WORD**: передать ответное слово.
  - **INITIATE_SELF_TEST**: начать самоконтроль.
  - **TRANSMITTER_SHUTDOWN**: блокировать передатчик.
  - **OVERRIDE_TRANSMITTER_SHUTDOWN**: разблокировать передатчик.
  - **INHIBIT_TF_BIT**: подавить бит флага терминала.
  - **OVERRIDE_INHIBIT_TF_BIT**: отменить подавление бита флага терминала.

    - **RESET_REMOTE_TERMINAL**: установить исходное состояние ОУ.
    - **TRANSMIT_VECTOR_WORD**: передать векторное слово.
    - **SYNCHRONIZE_WITH_DATA_WORD**: синхронизация с ИС.
    - **TRANSMIT_LAST_COMMAND_WORD**: передать последнее командное слово.
    - **TRANSMIT_BUILINTEST_WORD**: передать слово встроенного контроля.
    - **SELECTED_TRANSMITTER_SHUTDOWN**: блокировать выбранный передатчик.
    - **OVERRIDE_SELECTED_TRANSMITTER_SHUTDOWN**: разблокировать выбранный передатчик.
    - **ILLEGAL_COMMAND**: недопустимая команда.
    - **NO_COMMAND**: нет команды.

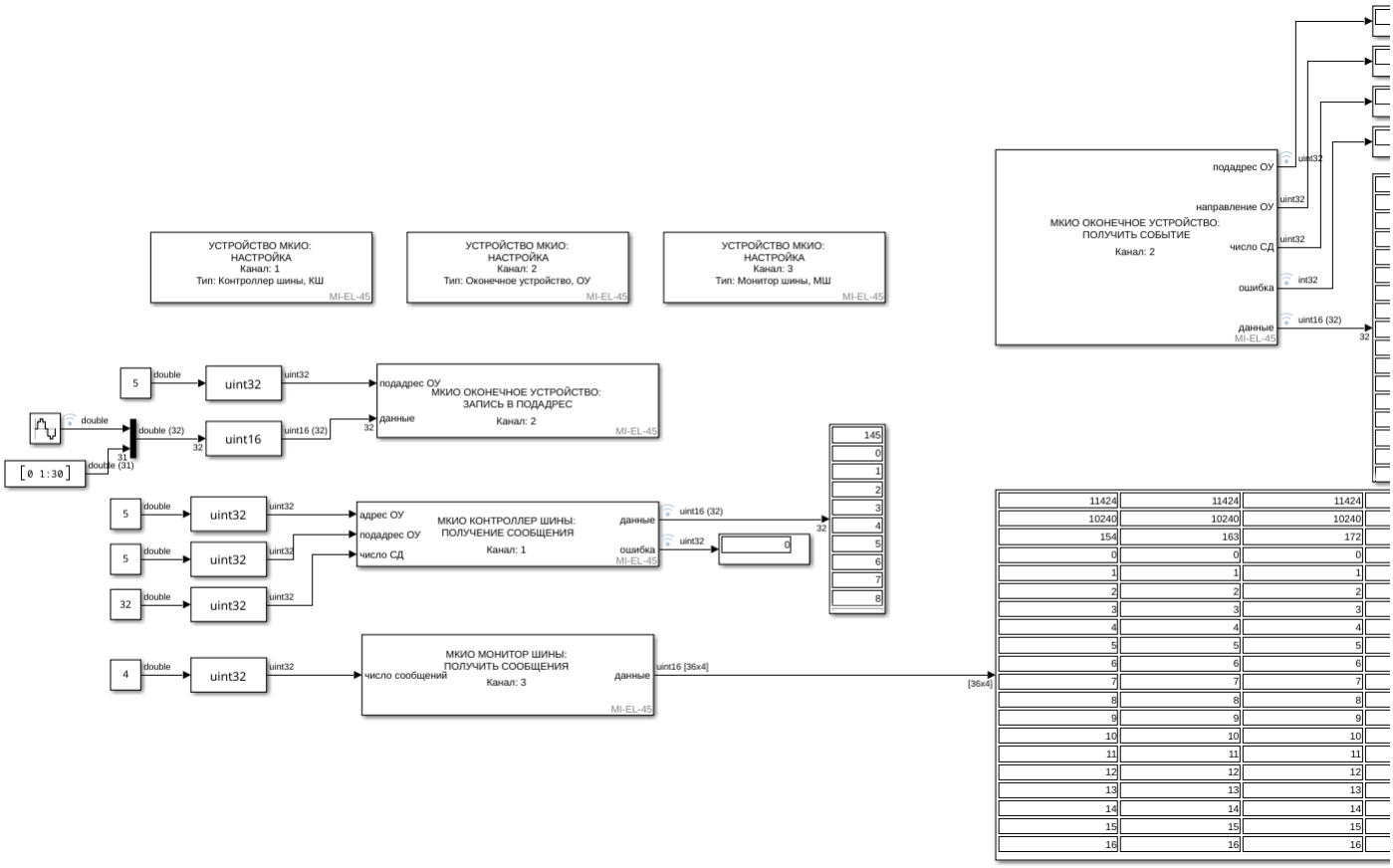
- **Статус**: Шина из 16 сигналов. Значение сигнала **1** в шине означает, что передача сообщения завершилась с соответствующим статусом.
    - **SUCCESS**: передача завершилась успешно.
    - **FORMAT_ERROR**: ошибка формата.
    - **MESSAGE_INCONSISTENCY**: неопределенность в сообщении.
    - **CW1_ERROR**: ошибка в первом КС.
    - **CW2_ERROR**: ошибка во втором КС.
    - **BUSA_JAM**: обнаружение генерации в основной ЛП.
    - **BUSB_JAM**: обнаружение генерации в резервной ЛП.
    - **DIFF_WC_IN_CW1_AND_CW2**: разное число информационных слов в первом и втором КС.
    - **STATUS_BIT_SET**: установлен бит в ОС.
    - **MANCHESTER_ENCODING_ERROR**: ошибка кода ‘Манчестер-2’.
    - **STATUS_TIMEOUT_ERROR**: нет ответного слова (Ошибка паузы перед ОС).
    - **DATA_TIMEOUT_ERROR**: нет информационного слова (слов) (Ошибка паузы перед ИС).
    - **EXTRA_DATA**: число ИС больше заданного.
    - **WRONG_RT_ADDRESS_IN_STATUS**: неверный адрес ОУ в ОС.
    - **SYNC_TYPE_ERROR**: неверный тип синхронизации.
    - **BUS_SWITCHING**: ошибка типа синхроимпульса.

- **Адрес 1**: Адрес оконечного устройства, которое участвовало в обмене.
- **Адрес 2**: Адрес оконечного устройства приёмника, для форматов 3 и 8. Адрес 31 - широковещательный.
- **Подадрес 1**: Подадрес оконечного устройства, который участвовал в обмене.
- **Подадрес 2**: Подадрес оконечного устройства приёмника, для форматов 3 и 8.
- **Число СД**: Количество прочитанных или записанных слов данных.
- **Данные**: Информационные слова.
- **Временная метка**: Временная метка регистрации сообщения. Содержит значение счётчика аппаратного таймера на момент регистрации сообщения. Счётчик 32-битный, имеет разрешение 1 мкс. Таймер запускается на этапе инициализации модели. После переполнения счётчика таймер продолжает считать с 0.

Примеры и демонстрации RITMeX MI-EL-45/MI-EL-55

Пример использования блоков MI-EL-45/MI-EL-55.

```
open_system('ritmdemo_block_miel45.slx')
```



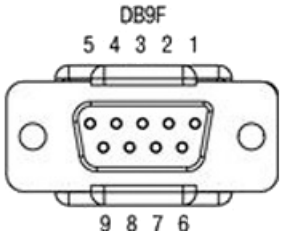
Назначение пинов MI-EL-45

Содержание

- Внешние разъемы
- Назначение контактов
- Тестовый жгут проводов
- Тип подключения
- Внешние разъемы
- Тестовый жгут проводов

Внешние разъемы

На модуле установлено два внешних разъема *DB9F*. Контакты нумеруются следующим образом:



Верхний разъем *DB9F* в таблице и на схеме обозначен как *X2*, а нижний разъем *DB9F* обозначен как *X3*.

Назначение контактов

Разъем	Контакт	Имя цепи	Разъем	Контакт	Имя цепи
X2	1	K1A_0	X3	1	K2A_0
X2	2	K1A_1	X3	2	K2A_1
X2	3	K1B_0	X3	3	K2B_0
X2	4	K1B_1	X3	4	K2B_1
X2	5	GND	X3	5	GND
X2	6	K3A_0	X3	6	K4A_0
X2	7	K3A_1	X3	7	K4A_1
X2	8	K3B_0	X3	8	K4B_0
X2	9	K3B_1	X3	9	K4B_1

Обозначения в имени цепи:
K(номер канала: от 1 до 4)(номер линии в канале: A или B)_(полярность линии: 0 или 1)

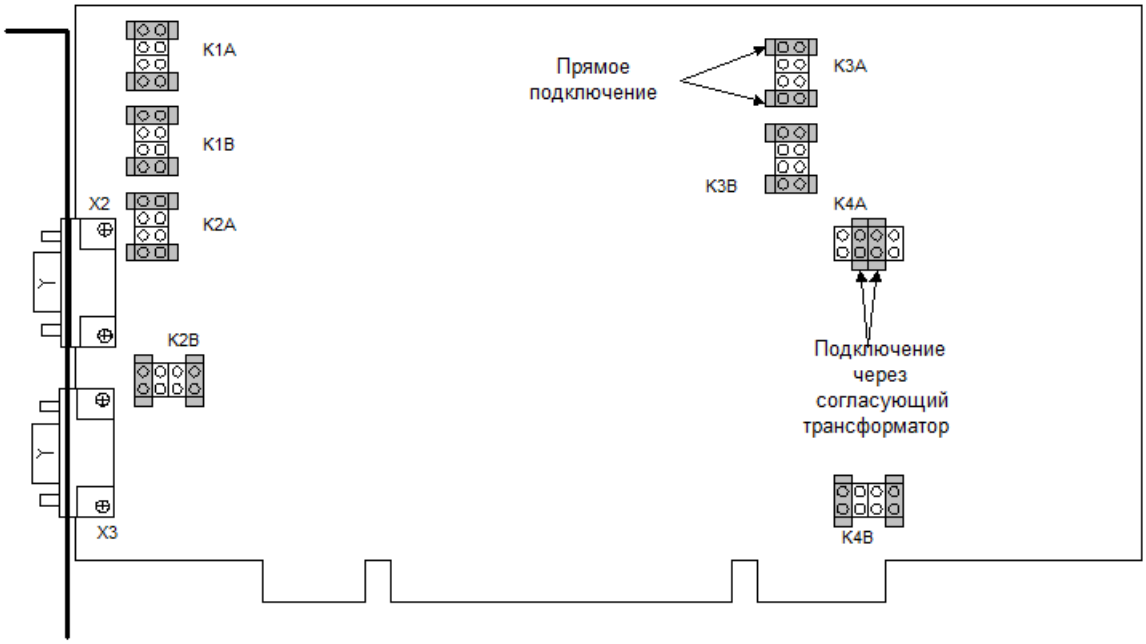
Тестовый жгут проводов

Поставляемый тестовый жгут проводов обеспечивает **прямое подключение** всех четырех каналов к линиям A и B.

При этом каналы 1 и 3 подключаются через один разъем DB9M, а каналы 2 и 4 подключаются через другой разъем DB9M.
Также предусмотрен вывод с тестового жгута на осциллограф.

Тип подключения

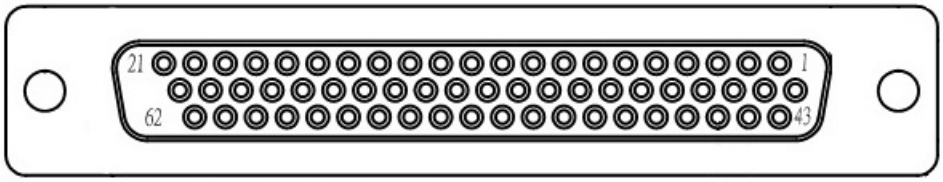
По умолчанию, джамперы платы выставлены в положение **прямого подключения**. Для подключения через **согласующий трансформатор** требуется переставить джамперы платы.



Замкнуты контакты	**Подключение к разъему**
KXX 1-2	Прямое подключение. Положительный контакт.
KXX 3-4	С согласующим трансформатором. Положительный контакт.
KXX 5-6	С согласующим трансформатором. Отрицательный контакт.
KXX 7-8	Прямое подключение. Отрицательный контакт.

Назначение пинов MI-EL-55

На модуле установлен разъем DHR-44F. Контакты нумеруются следующим образом:



Внешние разъемы

Номер контакта	**Назначение выхода**

Номер контакта	Назначение выхода
3	K1AD1
19	K1AT1
18	K1AT0
2	K1AD0
1	K1BD1
17	K1BT1
31	K1BT0
16	K1BD0
7	K2AD1
23	K2AT1
22	K2AT0
6	K2AD0
5	K2BD1
21	K2BT1
20	K2BT0
4	K2BD0
11	K3AD1
27	K3AT1
26	K3AT0
10	K3AD0
9	K3BD1
25	K3BT1
24	K3BT0
8	K3BD0
15	K4AD1
44	K4AT1
30	K4AT0
14	K4AD0
13	K4BD1

Номер контакта	Назначение выхода
29	K4BT1
28	K4BT0
12	K4BD0

Расшифровка обозначений
K
Номер канала 1,2,3,4
Линия: A — основная B — резервная
Тип подключения: D — прямое T — подключение через согласующий трансформатор
Полярность линии: 1 — положительный контакт 0 — отрицательный контакт

Тестовый жгут проводов

Поставляемый тестовый жгут проводов обеспечивает **прямое подключение** всех четырех каналов к линиям A и B.
Предусмотрен вывод с тестового жгута на осциллограф.

Модули поддержки CAN

Содержание

- Библиотека блоков Simulink для работы с CAN
- Примеры работы с библиотекой блоков CAN
- Модуль RITMeX CA-MA-45
- Модуль RITMeX CA-MA-55
- Модули RITMeX CA-ID-45 и RITMeX CA-ID-55

Библиотека блоков Simulink для работы с CAN

Примеры работы с библиотекой блоков CAN

Модуль RITMeX CA-MA-45

Двухканальный модуль для сопряжения с системами и устройствами по интерфейсу CAN 2.0 A/B. Интерфейс PCI.

Модуль RITMeX CA-MA-55

Двухканальный модуль для сопряжения с системами и устройствами по интерфейсу CAN 2.0 A/B. Интерфейс PCIe.

Модули RITMeX CA-ID-45 и RITMeX CA-ID-55

Двухканальный модуль для сопряжения с системами и устройствами по интерфейсу CAN 2.0 A/B.

Библиотека блоков RITM-CAN

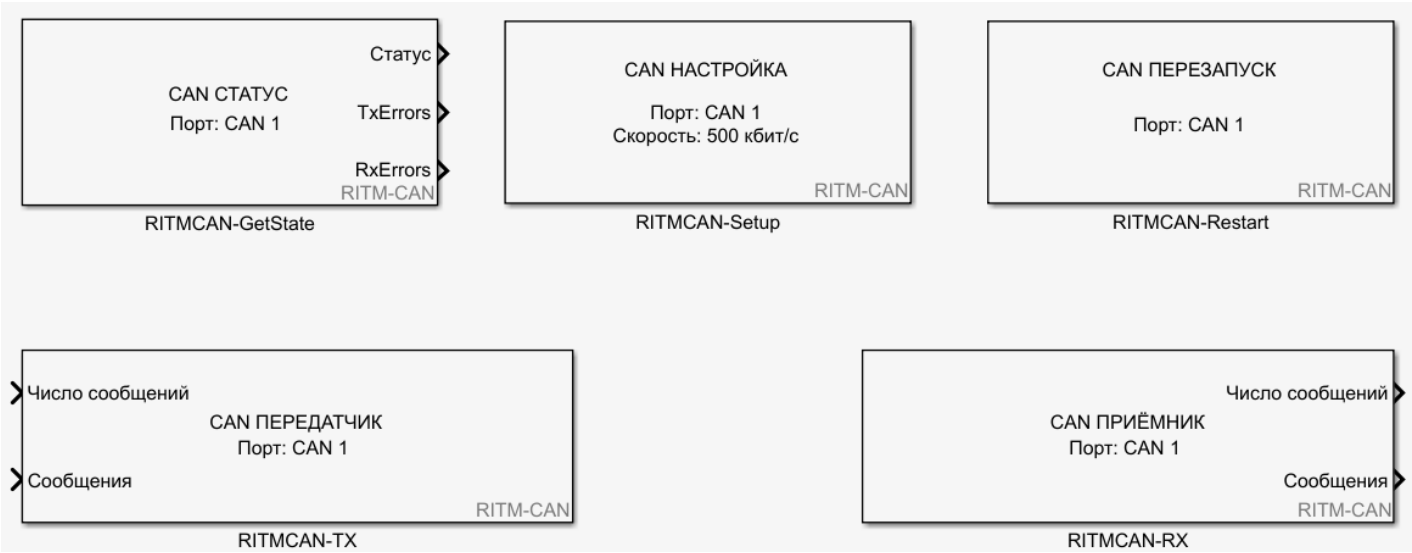
Содержание

- Описание
- Библиотека блоков
  - Блок CAN Настройка
  - Блок CAN Статус
  - Блок CAN Передатчик
  - Блок CAN Приёмник
  - Фильтр сообщений в блоке CAN Приёмник
  - Блок CAN Перезапуск

Описание

Библиотека блоков для настройки CAN интерфейсов КПМ “РИТМ”, отправки и приёма CAN сообщений.

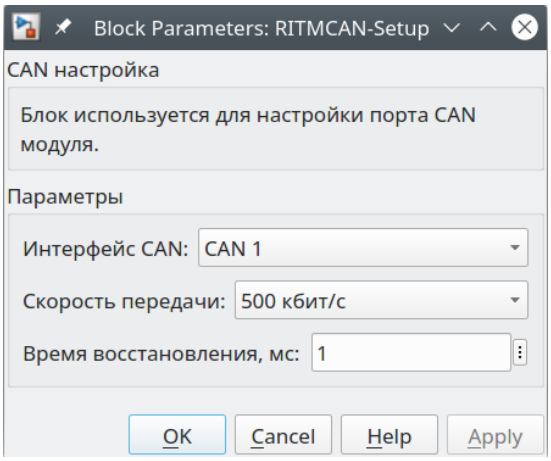
Библиотека блоков



Блок CAN Настройка

Блок используется для настройки порта CAN модуля.

Маска блока



Параметры блока

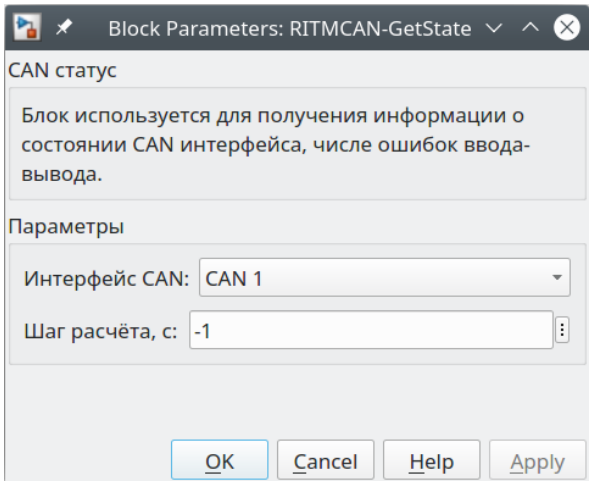
- **Интерфейс CAN**: Порт CAN, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ “РИТМ”.
- **Скорость передачи**: Позволяет задать скорость передачи в сети (от 10 Кбит/с до 1 Мбит/с). Все узлы в сети должны работать с одной скоростью.
- **Время восстановления, мс**: Время восстановления активного состояния интерфейса после попадания в

состояние Bus-off (в миллисекундах). Число **0** означает - не восстанавливаться после Bus-off.

Блок CAN Статус

Блок используется для получения информации о состоянии CAN интерфейса, числе ошибок ввода-вывода.

Маска блока



Параметры блока

- **Интерфейс CAN**: Порт CAN, состояние которого опрашивается.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для опроса состояния из приложения реального времени.

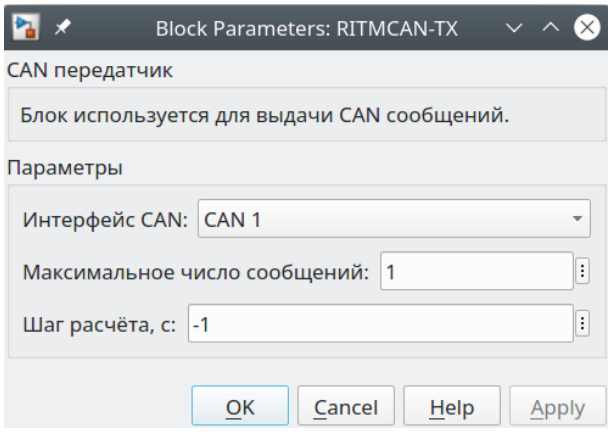
Входы блока

- **Статус**: Текущее состояние порта - шина из нескольких сигналов. Значение сигнала **1** в шине означает, что соответствующее состояние активно.
  - **ACTIVE**: приёмопередатчик исправен, число ошибок меньше 96.
  - **WARNING**: число ошибок меньше 128.
  - **PASSIVE**: приёмопередатчик прекращает активное исправление ошибочного трафика на линии, число ошибок меньше 256.
  - **BUS_OFF**: приёмопередатчик прекращает участие в обмене, число ошибок больше 256.
  - **STOPPED**: приёмопередатчик остановлен.
  - **SLEEPING**: приёмопередатчик в спящем режиме.
- **TxErrors**: Число ошибок при передаче.
- **RxErrors**: Число ошибок при приёме.

Блок CAN Передатчик

Блок используется для выдачи CAN сообщений.

Маска блока



Входы блока

- **Размер**: Число сообщений для передачи из массива **Данные**, тип данных *uint32*.

- **Данные**: Массив сообщений типа *CAN_MESSAGE_BUS* для передачи. Для создания сообщения CAN в типе данных *CAN_MESSAGE_BUS* используется блок CAN Pack.

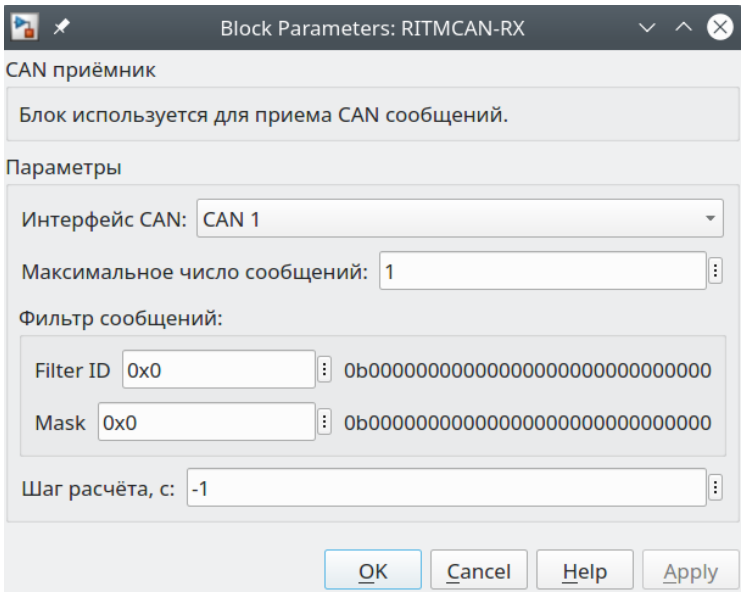
Параметры блока

- **Интерфейс CAN**: Порт CAN, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ “РИТМ”.
- **Максимальный размер сообщения**: Максимальное возможное значение входного параметра **Размер** и длины вектора **Данные**.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для отправки сообщений CAN из приложения реального времени.

Блок CAN Приёмник

Блок используется для приема CAN сообщений.

Маска блока



Выходы блока

- **Размер**: Число принятых сообщений, тип данных *uint32*. Не более значения параметра **“Максимальное число сообщений”**. Если в буфере операционной системы находится больше сообщений, они будут переданы в модель при следующем шаге расчёта.
- **Данные**: Массив принятых сообщений типа *CAN_MESSAGE_BUS*. Для дальнейшего использования сообщения CAN в модели (например, получения значения сигнала или метки времени) используется блок CAN Unpack.

Параметры блока

- **Интерфейс CAN**: Порт CAN, который будет использоваться для получения сообщений. Номера портов указаны на корпусе КПМ “РИТМ”.
- **Максимальное число сообщений**: Максимальное число сообщений передаваемое из буфера операционной системы в модель за 1 шаг расчёта.
- **Шаг расчёта**: Позволяет задать шаг расчёта в секундах для приема сообщений CAN в приложение реального времени.
- **Фильтр сообщений**: Фильтр принимаемых сообщений на уровне ОС. Для отключения фильтра и приёма всех сообщений задайте *Filter ID* и *Mask* как **0**. Подробное описание работы фильтра приводится в разделе Фильтр сообщений в блоке CAN Приёмник.

Фильтр сообщений в блоке CAN Приёмник

Длина идентификатора принимаемого сообщения CAN может быть 29 бит (для расширенного формата кадра) или 11 бит (для стандартного формата кадра). Сообщение CAN проходит через фильтр и принимается, только если все биты идентификатора прошли через фильтр. Если хотя бы один бит идентификатора сообщения не прошел через фильтр, то сообщение отбрасывается. Настройки **Filter ID** и **Mask** позволяют задать фильтр с учетом отдельных битов идентификатора принимаемого сообщения CAN.

Сообщение проходит через фильтр, если выполняется следующее условие:

CAN ID & Mask == Filter ID & Mask,
где CAN ID - идентификатор принимаемого сообщения CAN.

Другими словами, работу фильтра можно представить в виде следующих правил:

1. Если значение **Mask** для отдельного бита равно **нулю**, то этот бит не фильтруется. То есть, в фильтре этот бит не проверяется и **проходит** через фильтр.
2. Если значение **Mask** для отдельного бита равно **единице**, то осуществляется фильтрация по значению этого бита:
   а) Если бит из **CAN ID** и бит из **Filter ID совпадают**, то этот бит **проходит** через фильтр.
   б) Если бит из **CAN ID** и бит из **Filter ID отличаются**, то этот бит **не проходит** через фильтр.

Пример работы фильтра

Пусть задан следующий фильтр сообщений CAN:

```
FilterID = 0b0100;
Mask     = 0b0110;
```

Сообщения с CAN идентификаторами 0x4, 0x5 и 0xC **проходят** через фильтр и принимаются:

```
CAN_ID = [0b0100 0b0101 0b1100];
can_msg_passed = bitand(CAN_ID, Mask) == bitand(FilterID, Mask)
can_msg_passed =

 1×3 logical array

  1   1   1
```

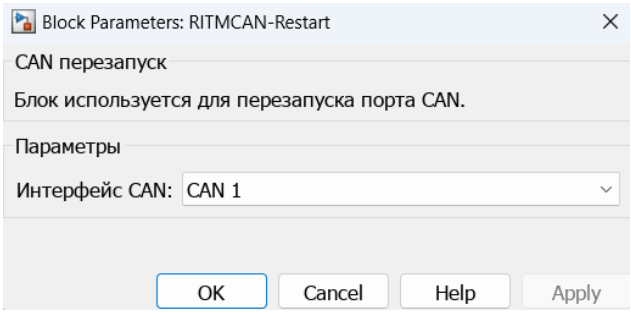
Сообщения с CAN идентификаторами 0x1, 0x2 и 0x6 **не проходят** через фильтр и отбрасываются:

```
CAN_ID = [0b0001 0b0010 0b0110];
can_msg_passed = bitand(CAN_ID, Mask) == bitand(FilterID, Mask)
can_msg_passed =

 1×3 logical array

  0   0   0
```

Блок CAN Перезапуск

Блок используется для перезапуска порта CAN.

Маска блока



Параметры блока

- **Интерфейс CAN**: Порт CAN, который будет перезапущен.

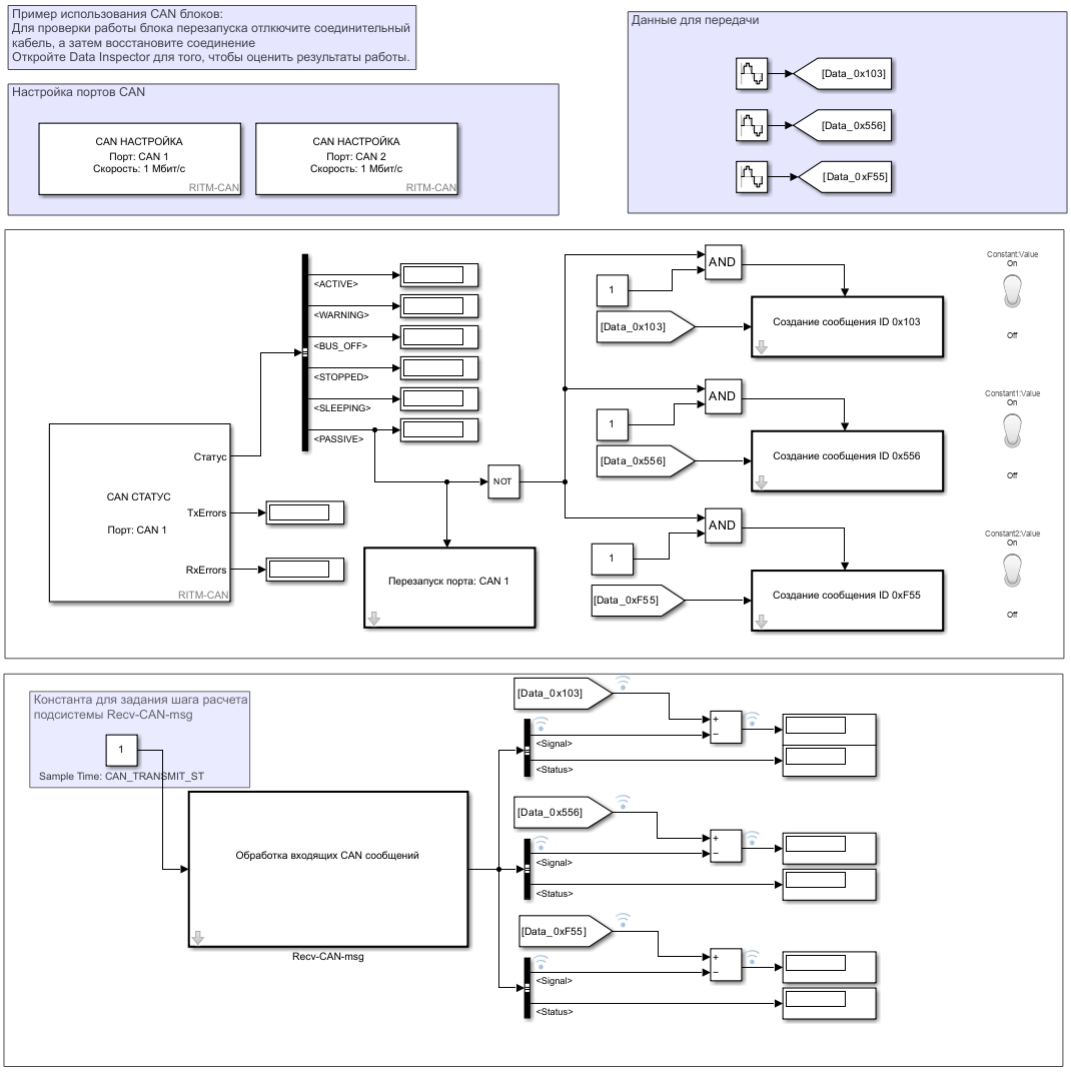
Шаг расчёта

В блоке нет возможности указать шаг расчёта. Предполагается, что блок будет помещен в тригируемую или включаемую подсистему для вызова по внешнему условию, формируемому пользователем.

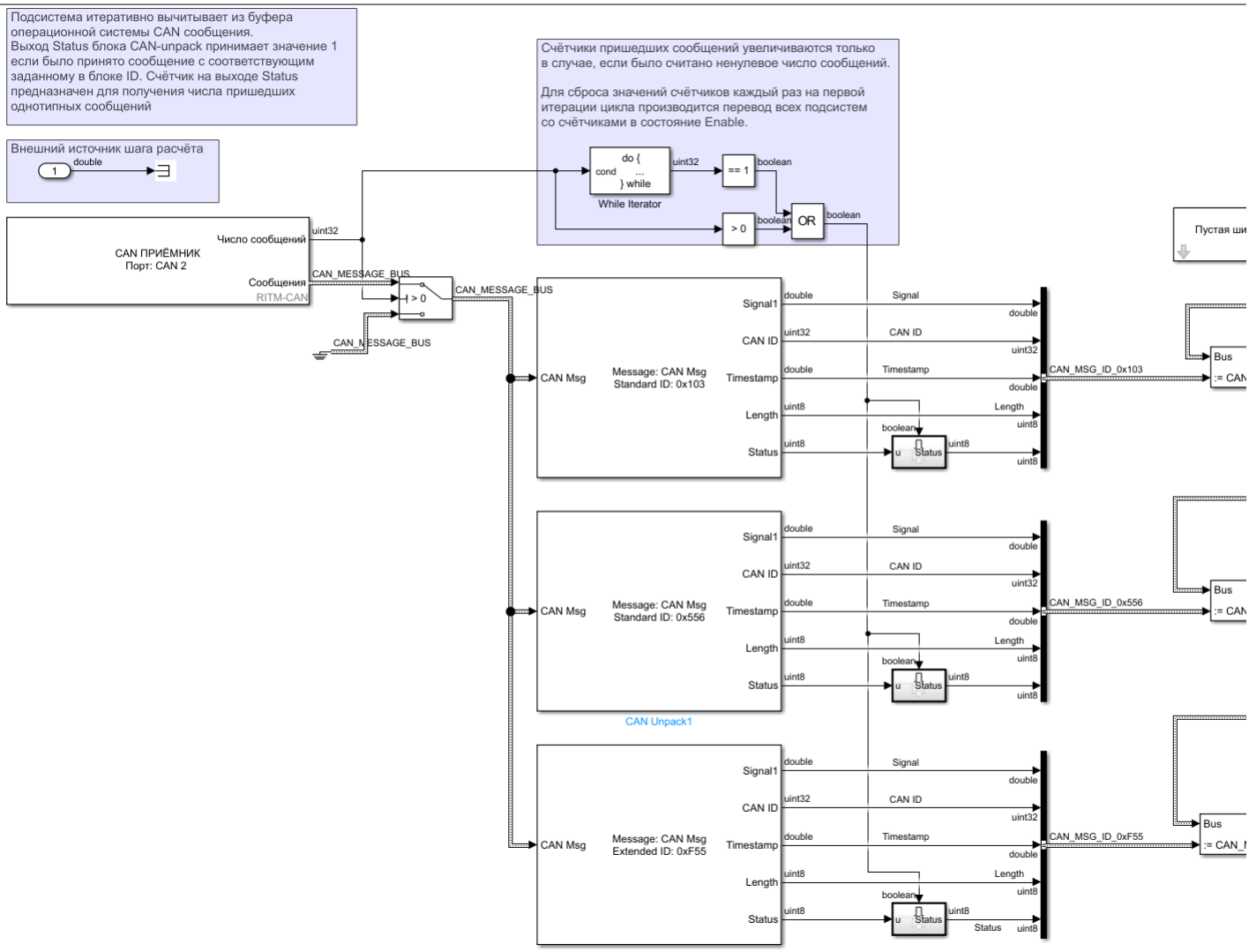
Примеры и демонстрации работы с CAN блоками КПМ “РИТМ”.

Пример использования блоков CAN.

```
open_system('ritmdemo_block_can.slx')
```



Содержание подсистемы “Обработка входящих CAN сообщений”

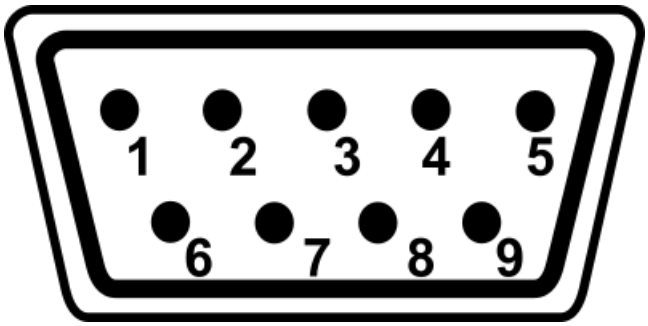

Подсистема итеративно вычитывает из буфера операционной системы CAN сообщения.
Выход Status блока CAN-unpack принимает значение 1 если было принято сообщение с соответствующим заданному в блоке ID. Счётчик на выходе Status предназначен для получения числа пришедших однотипных сообщений
Счётчики пришедших сообщений увеличиваются только в случае, если было считано ненулевое число сообщений.
Для сброса значений счётчиков каждый раз на первой итерации цикла производится перевод всех подсистем со счётчиками в состояние Enable.
Внешний источник шага расчёта
double
1
do {
cond
...
} while
While Iterator
uint32
== 1
boolean
> 0
boolean
OR
boolean
CAN ПРИЁМНИК
Порт: CAN 2
Число сообщений
uint32
Сообщения
RITM-CAN
CAN_MESSAGE_BUS
! > 0
CAN_MESSAGE_BUS
CAN_MESSAGE_BUS
CAN Msg
Message: CAN Msg
Standard ID: 0x103
Signal1
CAN ID
Timestamp
Length
Status
double
uint32
double
uint8
uint8
Signal
CAN ID
Timestamp
Length
Status
boolean
u
Status
uint8
CAN_MSG_ID_0x103
Bus
CAN Msg
Message: CAN Msg
Standard ID: 0x556
CAN_MSG_ID_0x556
CAN Unpack1
CAN Msg
Message: CAN Msg
Extended ID: 0xF55
CAN_MSG_ID_0xF55
Пустая ши

Назначение пинов CA-MA-45

Непосредственно на плате смонтированы два разъема SUB-D9 типа вилка (male) и розетка (female), соответствующие первому каналу CAN.

При необходимости задействования второго канала CAN производителем устанавливается дополнительная планка, содержащая два разъёма SUB-D9 типа вилка (male) и розетка (female), соответствующие второму каналу CAN.

Контакты разъёмов используются следующим образом:

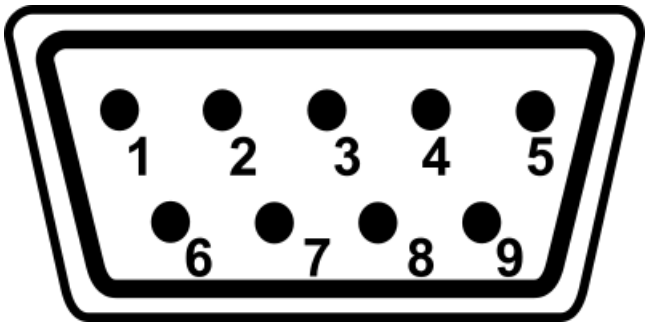


Контакт	Сигнал
1	Не используется
2	CAN_LOW
3	CAN_GND
4	Не используется
5	Не используется
6	CAN_GND
7	CAN_HIGH
8	Не используется
9	Не используется

Назначение пинов CA-MA-55

Непосредственно на плате смонтированы два разъёма SUB-D9 типа вилка (male), соответствующие первому и второму каналам CAN (подписаны CAN1 и CAN2).

Контакты разъёмов используются следующим образом:

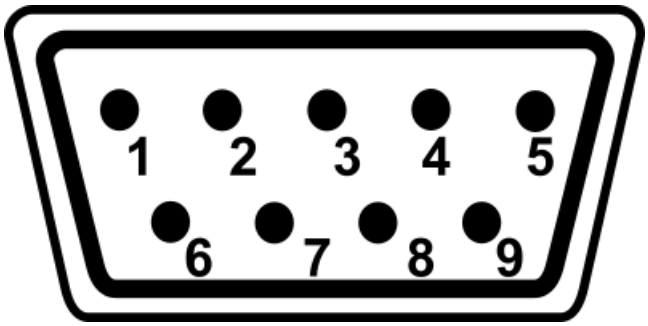


Контакт	Сигнал
1	Не используется
2	CAN_LOW
3	CAN_GND
4	Не используется
5	Не используется
6	CAN_GND
7	CAN_HIGH
8	Не используется
9	Не используется

Назначение пинов CA-ID-45 и CA-ID-55

Непосредственно на плате смонтированы два разъёма SUB-D9 типа вилка (male), соответствующие первому и второму каналам CAN.

Контакты разъёмов используются следующим образом:



Контакт	Сигнал
1	Не используется
2	CAN_LOW
3	CAN_GND
4	Не используется
5	Не используется
6	CAN_GND
7	CAN_HIGH
8	Не используется
9	Не используется

Модули поддержки ARINC 429

Содержание

- Блоки кодирования/декодирования слов ПК ARINC 429
- Модуль RITMeX AR-EL-45
- Модуль RITMeX AR-EL-55

Блоки кодирования/декодирования слов ПК ARINC 429

Блоки для создания слов последовательного кода из набора полей по ГОСТ 18977-79. Либо получения значений полей принятых слов ПК.

Модуль RITMeX AR-EL-45

PCI модуль для сопряжения с системами и устройствами по последовательному интерфейсу ARINC 429 (последовательный код - ПК) по ГОСТ 18977-79 и РТМ1495-75 и дискретным каналам (разовые команды - РК) по ГОСТ 18977-79.

Модуль RITMeX AR-EL-55

PCI Express модуль для сопряжения с системами и устройствами по последовательному интерфейсу ARINC 429 (последовательный код - ПК) по ГОСТ 18977-79 и РТМ1495-75 и дискретным каналам (разовые команды - РК) по ГОСТ 18977-79.

Блоки кодирования/декодирования слов ПК ARINC 429

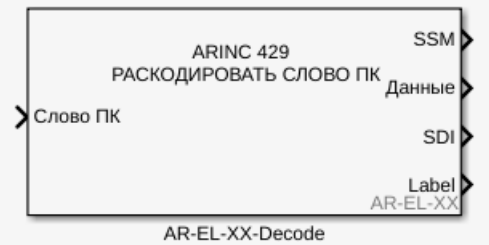
Содержание

- Блок ARINC 429 Раскодировать слово ПК
  - Маска блока
  - Входы блока
  - Выходы блока
- Блок ARINC 429 Закодировать слово ПК
  - Маска блока
  - Входы блока
  - Выходы блока

Блок ARINC 429 Раскодировать слово ПК

Блок используется для получения значений полей слова ПК ARINC 429.

Маска блока



Входы блока

- **Слово ПК**: Слово последовательного кода, тип данных *uint32*.

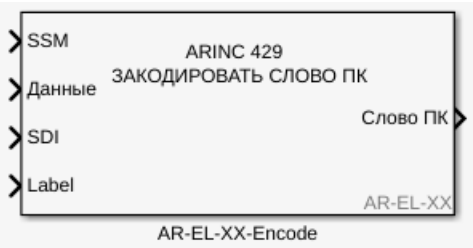
Выходы блока

- **SSM**: Матрица признака (Sign/Status Matrix), значащие биты 1-2, тип данных *uint32*.
- **Данные**: Сегмент данных слова ПК, значащие биты 1-19, тип данных *uint32*.
- **SDI**: Идентификатор источника/назначения (Source/Destination Identifiers), значащие биты 1-2, тип данных *uint32*.
- **Label**: Метка - идентификатор данных, значащие биты 1-8, тип данных *uint32*.

Блок ARINC 429 Закодировать слово ПК

Блок используется для создания слова ПК ARINC 429 из его полей.

Маска блока



Входы блока

- **SSM**: Матрица признака (Sign/Status Matrix), значащие биты 1-2, тип данных *uint32*.
- **Данные**: Сегмент данных слова ПК, значащие биты 1-19, тип данных *uint32*.
- **SDI**: Идентификатор источника/назначения (Source/Destination Identifiers), значащие биты 1-2, тип данных *uint32*.
- **Label**: Метка - идентификатор данных, значащие биты 1-8, тип данных *uint32*.

Выходы блока

- **Слово ПК**: Слово последовательного кода, тип данных *uint32*.

Модуль RITMeX AR-EL-45

Содержание

- Описание
- Библиотека блоков
  - Блок ARINC 429 Настройка
  - Блок ARINC 429 Передатчик
  - Блок ARINC 429 Приёмник

Описание

Модуль для сопряжения КПМ “РИТМ” с системами и устройствами по последовательному интерфейсу ARINC 429 (последовательный код - ПК) по ГОСТ 18977-79 и РТМ1495-75 и дискретным каналам (разовые команды - РК) по ГОСТ 18977-79.

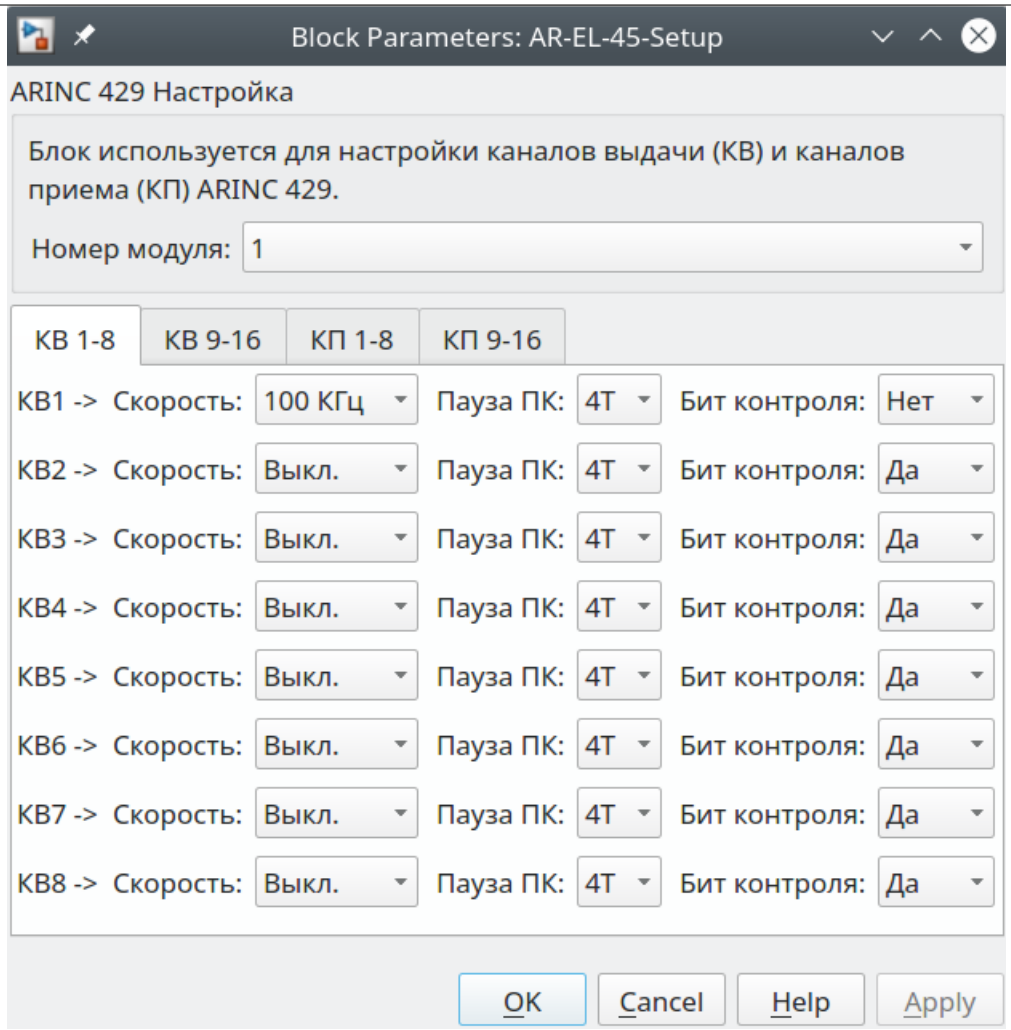
- Количество входных каналов: 16
- Количество выходных каналов: 16
- Частота принимаемых ПК: (12..14,5) КГц, 48 КГц +/-25%, 100 КГц +/-1%;
- Частота выдаваемых ПК: 12,5 КГц +/-1%, 50 КГц +/-1%, 100 КГц +/-1%;

Библиотека блоков

Блок ARINC 429 Настройка

Блок используется для настройки каналов выдачи (КВ) и каналов приема (КП) ARINC 429

Маска блока



Параметры блока

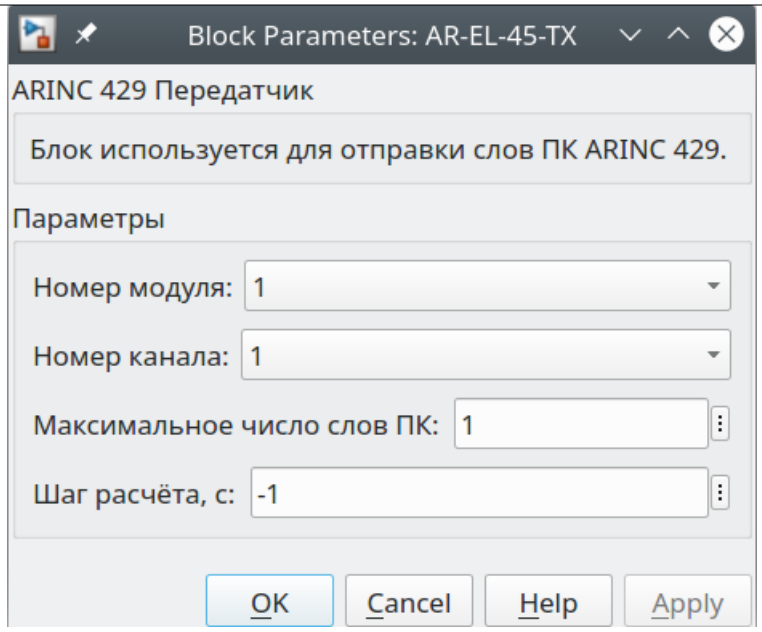
Блок позволяет индивидуально настроить каналы выдачи (с 1 по 16) и каналы приема (с 1 по 16).

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Скорость***: Частота выдаваемых или принимаемых слов (12,5 КГц, 50 КГц, 100 КГц). Значение “Выкл.” отключает канал.
- ***Пауза ПК***: Пауза между выдаваемыми словами (4Т или 8Т), где Т - период последовательного кода.
- ***Бит контроля***: Формировать или не формировать бит контроля нечетности.

Блок ARINC 429 Передатчик

Блок используется для отправки слов ПК ARINC 429.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Номер канала***: Номер канала выдачи, с которого будет производиться выдача последовательного кода.
- ***Максимальное число слов ПК***: Наибольшее число слов последовательного кода, доступное для передачи за 1 раз. Не более **256** слов.
- ***Шаг расчёта:*** Позволяет задать шаг расчета в секундах для выдачи слов ПК.

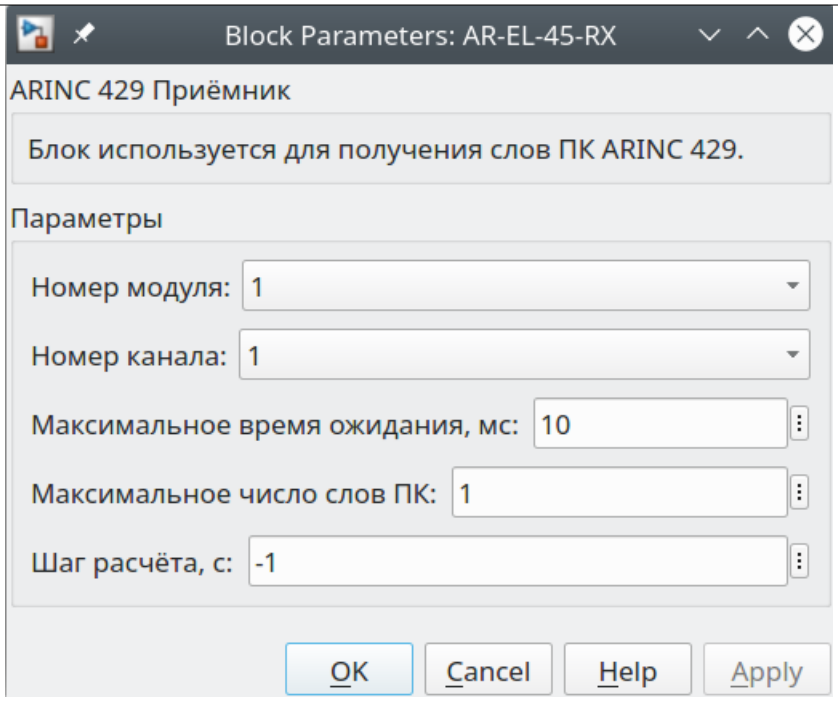
Входы блока

- **Размер**: Число слов ПК для передачи из массива **Данные**, тип данных *uint32*.
- **Данные**: Массив слов ПК для передачи, тип данных *uint32*.

Блок ARINC 429 Приёмник

Блок используется для получения слов ПК ARINC 429.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Номер канала***: Номер канала приёма, на который будет производиться приём последовательного кода.
- ***Максимальное время ожидания***: Наибольшее время, в течение которого модуль будет ожидать приёма запрошенного числа слов. Если число слов, принятое модулем за это время, меньше необходимого, слова ПК возвращены не будут.
- ***Максимальное число слов ПК***: Наибольшее число слов последовательного кода, доступное для приёма за 1 шаг расчета. Не более **256** слов.
- ***Шаг расчёта:*** Позволяет задать шаг расчета в секундах для приёма слов ПК.

Входы блока

- **Размер**: Число слов ПК для приёма, тип данных *uint32*.

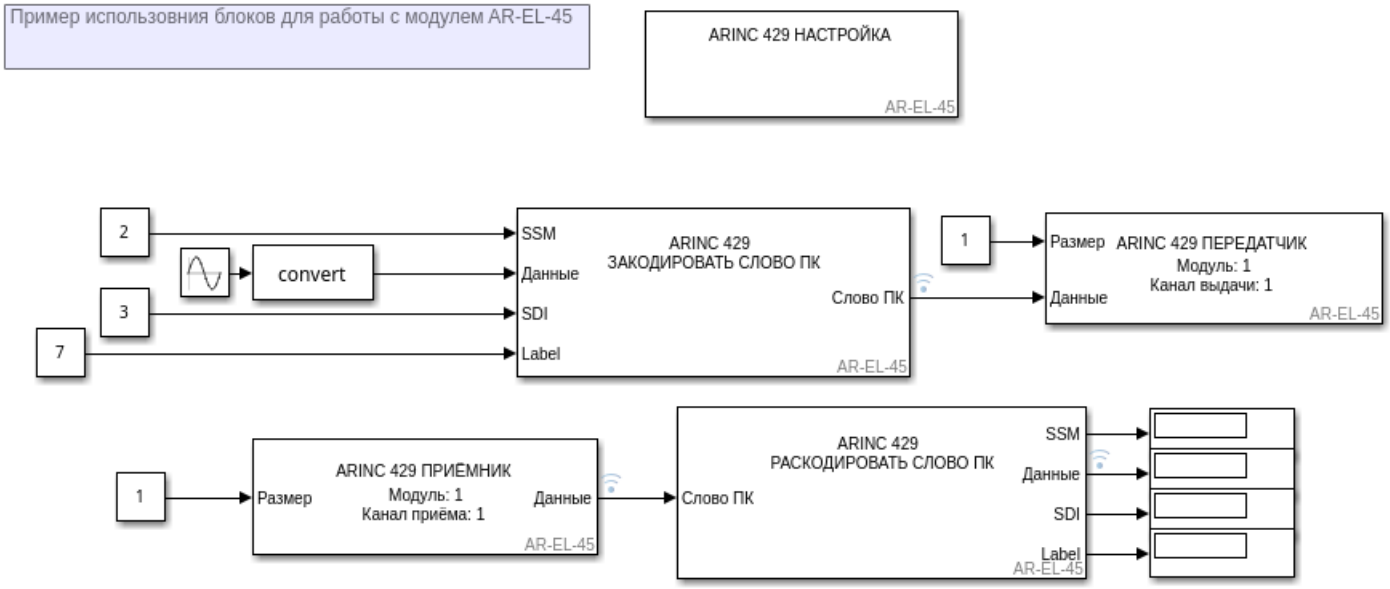
Выходы блока

- **Данные**: Массив принятых слов ПК, тип данных *uint32*.

Примеры и демонстрации AR-EL-45

Пример использования блоков AR-EL-45.

```
open_system('ritmdemo_block_arel45.slx')
```



Назначение пинов AR-EL-45

Назначение контактов внешнего разъема (розетка DHR-78F):

Контакт	Имя цепи		Контакт	Имя цепи		Контакт	Имя цепи		Контакт	Имя цепи
78	КВ10Б		59	КВ9А		39	КВ9Б		20	КВ8А
77	КВ11Б		58	КВ10А		38	КВ7А		19	КВ8Б
76	КВ12Б		57	КВ11А		37	КВ6А		18	КВ7Б
75	GND		56	КВ12А		36	КВ5А		17	КВ6Б
74	КВ13Б		55	GND		35	КВ4А		16	КВ5Б
73	КВ14Б		54	КВ13А		34	КВ3А		15	КВ4Б
72	КВ15Б		53	КВ14А		33	КВ2А		14	КВ3Б
71	КВ16Б		52	КВ15А		32	КВ1А		13	КВ2Б
70	GND		51	КВ16А		31	ДВ2 ТТЛ		12	КВ1Б
69	GND		50	ДВ4 ОК		30	ДП4		11	ДВ1 ТТЛ
68	КП12Б		49	ДВ3 ОК		29	ДП3		10	ДП2
67	КП11Б		48	КП12А		28	КП13Б		9	ДП1
66	КП10Б		47	КП11А		27	КП13А		8	КП14Б
65	КП9Б		46	КП10А		26	КП15А		7	КП14А
64	GND		45	КП9А		25	КП15Б		6	КП16А
63	КП4Б		44	GND		24	КП5Б		5	КП16Б
62	КП3Б		43	КП4А		23	КП5А		4	КП6Б
61	КП2Б		42	КП3А		22	КП7А		3	КП6А
60	КП1Б		41	КП2А		21	КП7Б		2	КП8А
			40	КП1А					1	КП8Б

Замыкание каналов в поставляемой тестовой заглушке:

КП1,..КП16 = КВ1,..КВ16.

Модуль RITMeX AR-EL-55

Содержание

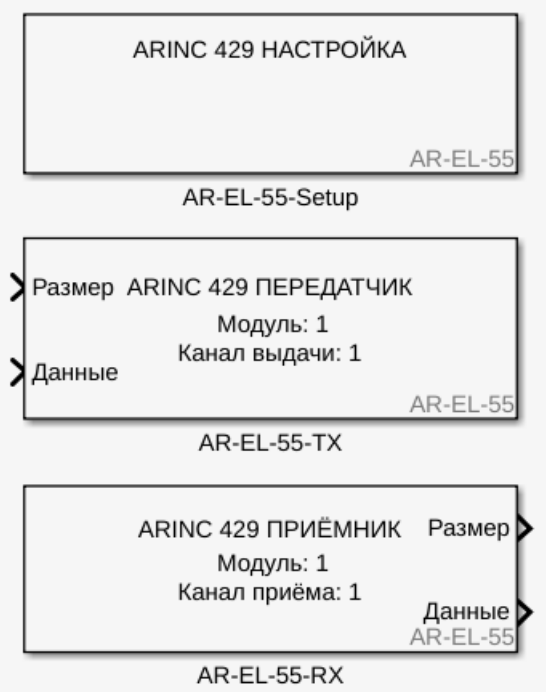
- Описание
- Библиотека блоков
  - Блок ARINC 429 Настройка
  - Блок ARINC 429 Передатчик
  - Блок ARINC 429 Приёмник

Описание

Модуль для сопряжения КПМ “РИТМ” с системами и устройствами по последовательному интерфейсу ARINC 429 (последовательный код - ПК) по ГОСТ 18977-79 и РТМ1495-75 и дискретным каналам (разовые команды - РК) по ГОСТ 18977-79.

- Количество входных каналов: 8
- Количество выходных каналов: 8
- Частота принимаемых ПК: (12..14,5) КГц, 48 КГц +/-25%, 100 КГц +/-1%;
- Частота выдаваемых ПК: 12,5 КГц +/-1%, 50 КГц +/-1%, 100 КГц +/-1%;

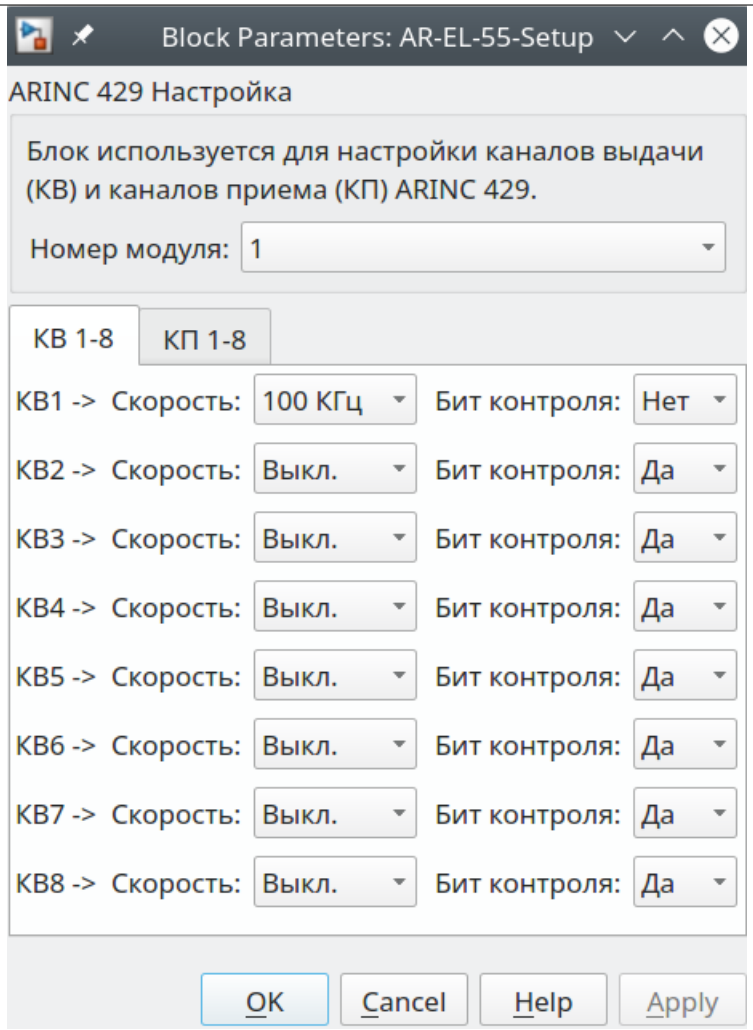
Библиотека блоков



Блок ARINC 429 Настройка

Блок используется для настройки каналов выдачи (КВ) и каналов приема (КП) ARINC 429

Маска блока



Параметры блока

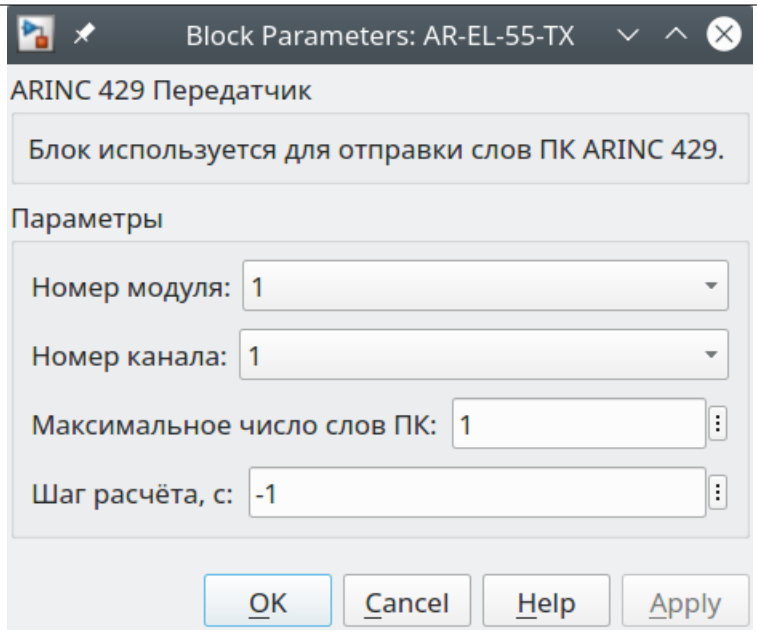
Блок позволяет индивидуально настроить каналы выдачи (с 1 по 8) и каналы приема (с 1 по 8).

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Скорость***: Частота выдаваемых или принимаемых слов (12,5 КГц, 50 КГц, 100 КГц). Значение “Выкл.” отключает канал.
- ***Бит контроля***: Формировать или не формировать бит контроля нечетности.

Блок ARINC 429 Передатчик

Блок используется для отправки слов ПК ARINC 429.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Номер канала***: Номер канала выдачи, с которого будет производиться выдача последовательного кода.
- ***Максимальное число слов ПК***: Наибольшее число слов последовательного кода, доступное для передачи за 1 раз. Не более **256** слов.
- ***Шаг расчёта:*** Позволяет задать шаг расчета в секундах для выдачи слов ПК.

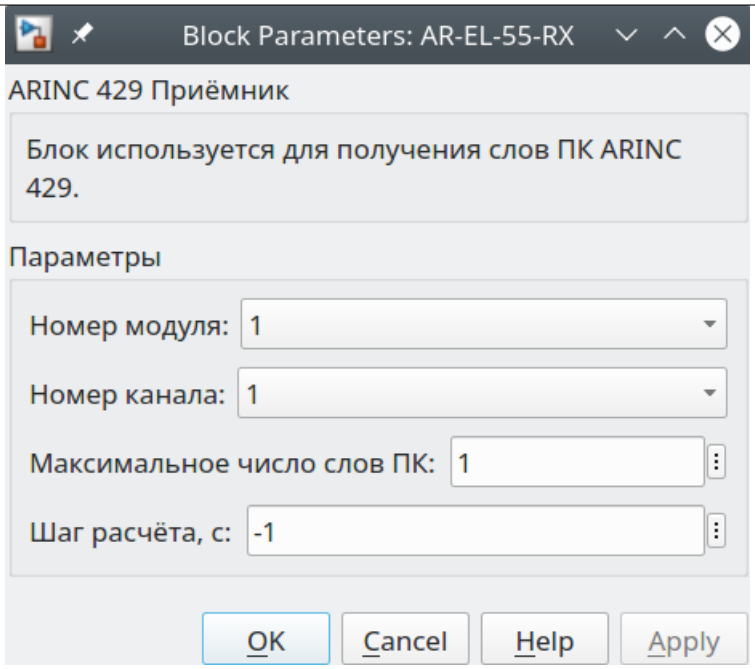
Входы блока

- **Размер**: Число слов ПК для передачи из массива **Данные**, тип данных *uint32*.
- **Данные**: Массив слов ПК для передачи, тип данных *uint32*.

Блок ARINC 429 Приёмник

Блок используется для получения слов ПК ARINC 429.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа. Номера модулей подписаны на корпусе КПМ “РИТМ”.
- ***Номер канала***: Номер канала приёма, на который будет производиться приём последовательного кода.
- ***Максимальное число слов ПК***: Наибольшее число слов последовательного кода, доступное для приёма за 1 шаг расчета. Не более **255** слов - это размер внутреннего кольцевого буфера. При его переполнении непрочитанные слова будут замещаться более новыми.
- ***Шаг расчёта:*** Позволяет задать шаг расчета в секундах для приёма слов ПК.

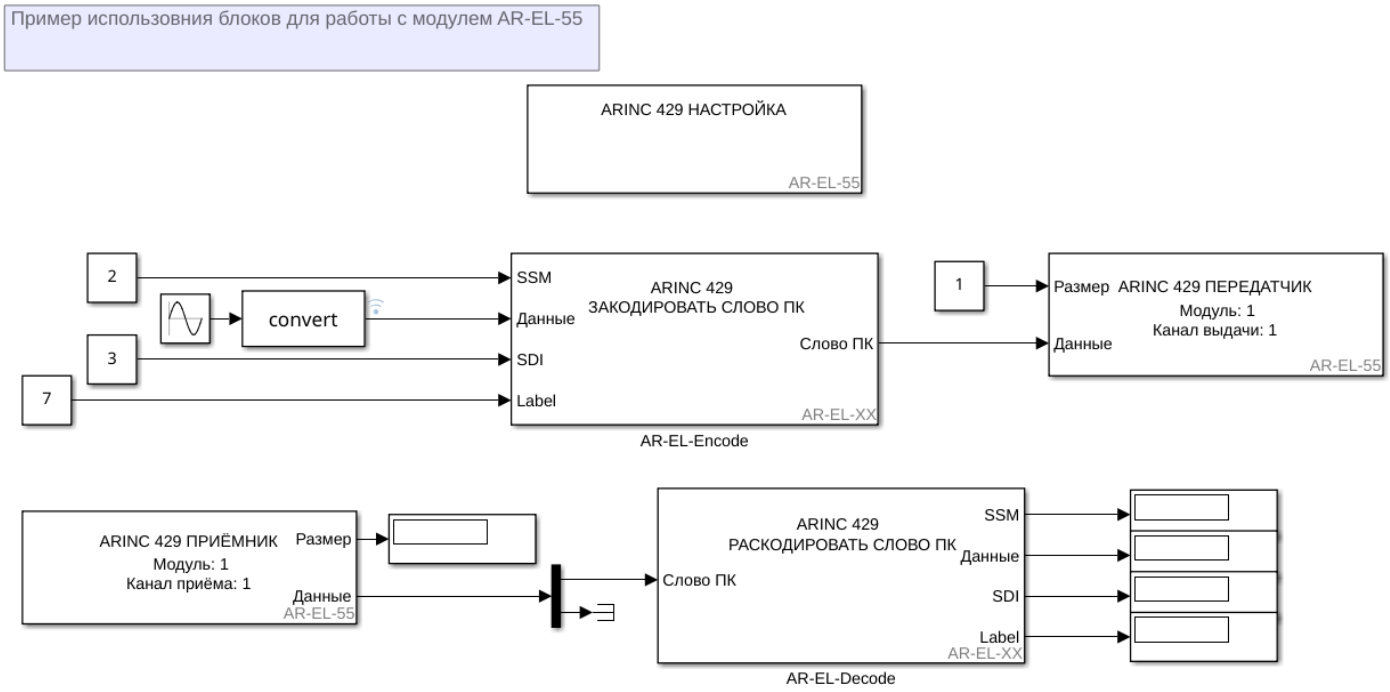
Выходы блока

- **Размер**: Число принятых слов ПК, записанных в массив **Данные**, тип данных *uint32*.
- **Данные**: Массив принятых слов ПК, тип данных *uint32*.

Примеры и демонстрации AR-EL-55

Пример использования блоков AR-EL-55.

```
open_system('ritmdemo_block_arel55.slx')
```



Назначение пинов AR-EL-55

Назначение контактов внешнего разъема (розетка DHR-44F):

Контакт	Имя цепи		Контакт	Имя цепи		Контакт	Имя цепи		Контакт	Имя цепи
1	КП4Б		12	КП5Б		23	КВ2Б		34	КВ5Б
2	КП4А		13	-		24	КВ1А		35	КВ4А
3	КП3Б		14	-		25	КП2А		36	КВ3Б
4	КП3А		15	GND		26	КП1А		37	КВ2А
5	КП8Б		16	GND		27	КП6А		38	КВ1Б
6	КП8А		17	КВ8Б		28	КП5А		39	GND
7	КП7Б		18	КВ7А		29	-		40	-
8	КП7А		19	КВ6Б		30	-		41	-
9	КП2Б		20	КВ5А		31	КВ8А		42	-
10	КП1Б		21	КВ4Б		32	КВ7Б		43	-
11	КП6Б		22	КВ3А		33	КВ6А		44	-

Замыкание каналов в поставляемой тестовой заглушке:

КП1,..КП8 = КВ1,..КВ8.

Запуск алгоритмов на ПЛИС

Содержание

- Общие сведения о запуске алгоритмов на ПЛИС
- Подключение цифровых входов-выходов к алгоритму на ПЛИС
- Подключение аналоговых входов-выходов к алгоритму на ПЛИС
- Подключение аналоговых входов-выходов RF1 к алгоритму на ПЛИС
- Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС
- Моделирование силовой электроники в реальном времени на ПЛИС

Модули ввода-вывода на базе ПЛИС и цифровых, аналоговых, оптических и видео субмодулей. Позволяют пользователю самостоятельно реализовать любой функционал на ПЛИС.

Общие сведения о запуске алгоритмов на ПЛИС

Общая информация о технологии, возможностях и ограничениях при реализации алгоритмов на ПЛИС.

Подключение цифровых входов-выходов к алгоритму на ПЛИС

В этом разделе объясняется, как подключить цифровые входы и выходы к алгоритму, работающему на ПЛИС.

Подключение аналоговых входов-выходов к алгоритму на ПЛИС

В этом разделе объясняется, как подключить аналоговые входы и выходы к алгоритму, работающему на ПЛИС.

Подключение аналоговых входов-выходов RF1 к алгоритму на ПЛИС

В этом разделе объясняется, как подключить RF1 аналоговые входы и выходы к алгоритму, работающему на ПЛИС.

Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС

В этом разделе даются базовые модели, подключения цифровых/аналоговых входов и выходов к алгоритму, работающему на ПЛИС.

Моделирование силовой электроники в реальном времени на ПЛИС

В этом разделе показан пример запуска физической модели с элементами силовой электроники на ПЛИС.

Общие сведения о запуске алгоритмов на ПЛИС

Содержание

- Общая информация
- Как это работает
  - С точки зрения аппаратуры
  - С точки зрения пользователя
  - С точки зрения программного обеспечения
- Установка
- Дополнительная информация

Общая информация о технологии, возможностях и ограничениях при реализации алгоритмов на ПЛИС

Общая информация

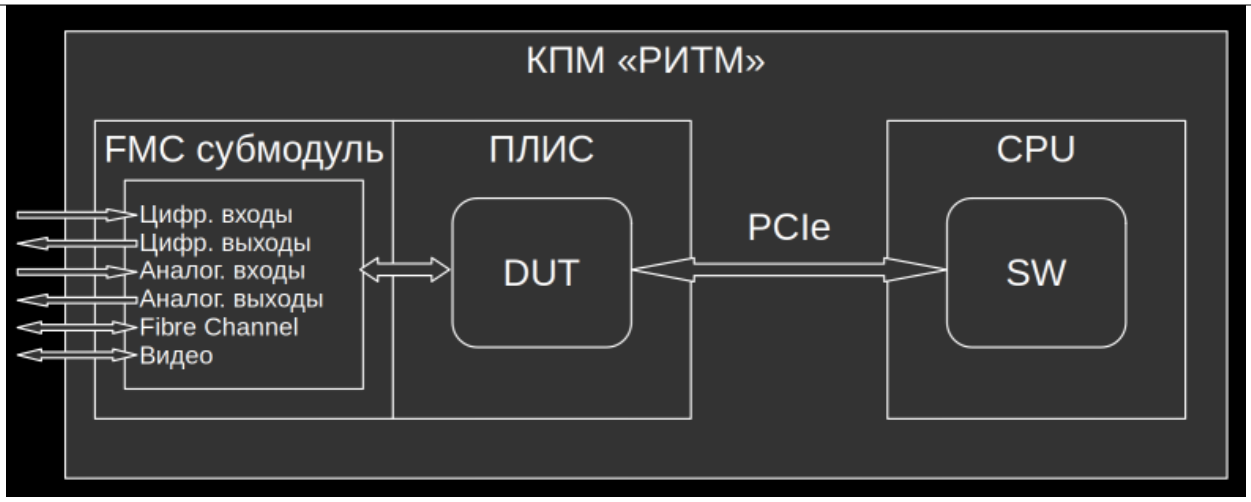
В некоторых случаях производительности центрального процессора РИТМ (будем называть его в дальнейшем CPU) не достаточно для решения конкретных задач. Например:

- Шаг расчета алгоритма должен быть меньше минимального шага расчета, который позволяет достичь ОСРВ “РИТМ.Реальное время” с данной моделью, на данном РИТМе;
- Не хватает вычислительных ресурсов для решения задачи в реальном времени на CPU (например, для алгоритма обработки видео, или потока данных с мультигигабитной оптической линии, или для симуляции высокочастотного процесса в силовой электронике);
- Требуется вынести часть расчетов за пределы CPU;
- Требуется подключить к алгоритму на ПЛИС цифровые, аналоговые или другие интерфейсы и выполнять задачу обработки гораздо быстрее, чем это возможно на CPU.

В этих случаях предпочтительно использовать модуль ввода-вывода на базе ПЛИС как сопроцессор, помогающий CPU решать общую задачу быстро и эффективно.

Как это работает

Эта технология позволяет переносить алгоритмы на ПЛИС и запускать их совместно с алгоритмами на CPU с подключением к внешним интерфейсам РИТМ.



С точки зрения аппаратуры

В комплексе полунатурного моделирования РИТМ установлен модуль ввода-вывода, содержащий на себе кристалл ПЛИС. Этот модуль ввода-вывода также содержит субмодуль (мезонинная плата с разъёмом FMC, или просто FMC в дальнейшем), предоставляющий внешние интерфейсы (цифровые, аналоговые, оптические и другие). Эти внешние интерфейсы доступны напрямую из ПЛИС. Кроме того, ПЛИС и CPU могут обмениваться друг с другом данными через шину PCI Express.

С точки зрения пользователя

Пользователь разделяет свою модель на две части. Одна часть работает на ПЛИС (т.н. DUT - Device Under Test), а другая часть работает на CPU (т.н. программная модель - SW). Пользователь назначает интерфейсы DUT - например, выбирает внешние интерфейсы субмодуля (FMC) и интерфейсы между DUT и SW (например, общение через DMA или регистры). При помощи мастера настройки HDL Workflow Advisor модель DUT реализуется в ПЛИС, а программная модель - в CPU.

С точки зрения программного обеспечения

Из модели DUT автоматически генерируется HDL код и вся необходимая обвязка для реализации на ПЛИС. Автоматически создается и синтезируется проект в среде разработки для ПЛИС. Из программной модели автоматически генерируется C код и собирается под центральный процессор CPU. Программирование ПЛИС осуществляется при загрузке программной модели в РИТМ. От пользователя **не требуется написания кода** C или HDL ни на одном из этапов.

Установка

Для корректной работы HDL Workflow Advisor настройте пути к среде разработки Xilinx Vivado, например:

Для Windows:

```
hdlsetuptoolpath('ToolName', 'Xilinx Vivado', 'ToolPath', 'C:\Xilinx\Vivado\2022.1\bin\vivado.bat');
```

Для Linux:

```
hdlsetuptoolpath('ToolName', 'Xilinx Vivado', 'ToolPath', '/tools/Xilinx/Vivado/2022.1/bin/vivado');
```

Дополнительная информация

Дополнительная информация содержится в разделе документации “Hardware-Software Co-Design” инструмента HDL Coder.

Подключение цифровых входов-выходов к алгоритму на ПЛИС

Содержание

- Требования к аппаратному обеспечению
- Требования к программному обеспечению
- Общая структура системы
- Пример использования
  - Демо модель
  - Генерация прошивки для ПЛИС
  - Обзор программной модели

В этом разделе объясняется, как подключить цифровые входы и выходы к алгоритму, работающему на ПЛИС.

Требования к аппаратному обеспечению

- Установленная базовая плата RITMeX BA-IS-45 или RITMeX BA-IS-55
- FMC расширение для базовой платы ПЛИС RITMeX GP-RT-FMC Level 1

Требования к программному обеспечению

- HDL Coder для генерации кода
- Fixed-Point Designer для работы с HDL Coder
- Xilinx Vivado для программирования ПЛИС:
  - Для программирования BA-IS-55 требуется версия Xilinx Vivado с поддержкой Ultrascale
  - Для программирования BA-IS-45 требуется версия Xilinx Vivado с поддержкой 7 Series
  - Работа верифицирована на версии Xilinx Vivado 2022.1

Общая структура системы

Для лучшего понимания взаимодействия железа и софта обратитесь к разделу Общие сведения о запуске алгоритмов на ПЛИС

Пример использования

В качестве примера рассмотрим программно-аппаратную реализацию сигнала типа “меандр”. Логика формирования меандра будет реализована на ПЛИС, а период меандра будет задаваться в программной части.

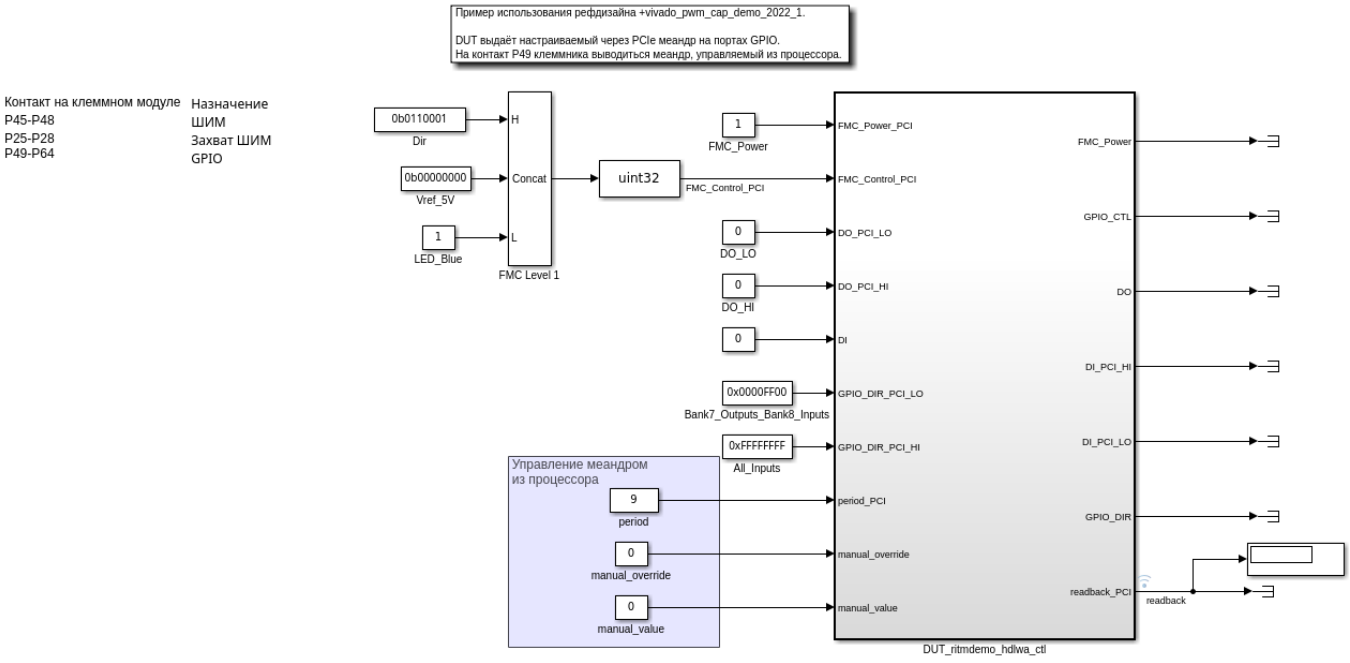
Демо модель

Рассмотрим демо модель.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_ctl.slx')
```

Для модуля BA-IS-55.

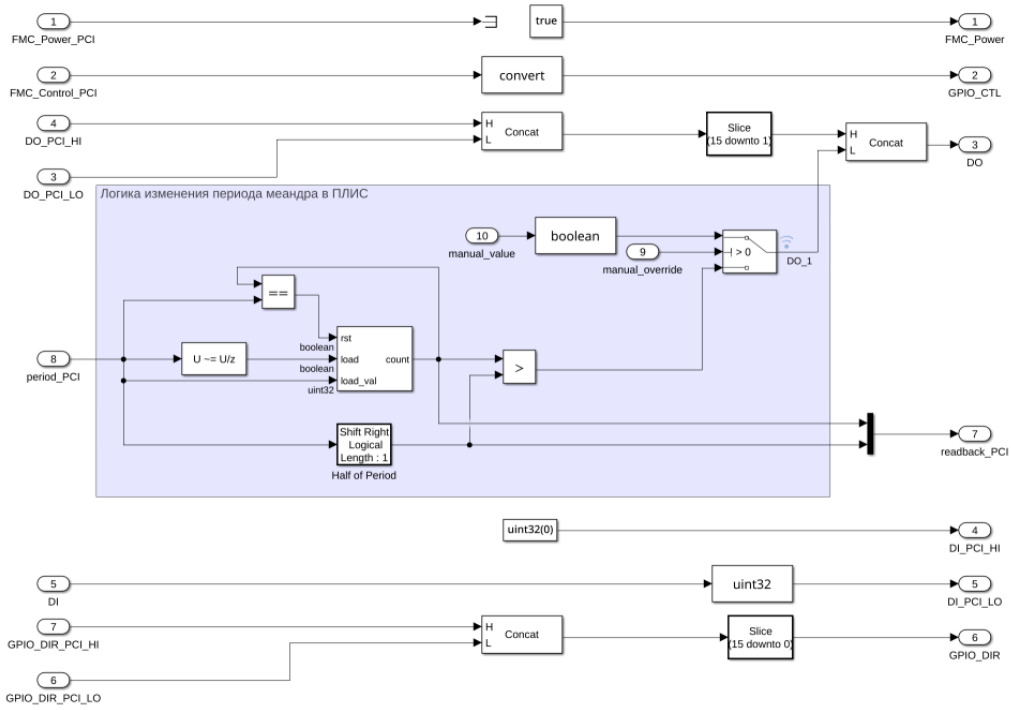
```
open_system('ritmdemo_hdlwa_ctl_BA_IS_55.slx')
```



Модель состоит из двух частей. Первая часть - это подсистема DUT_ritmdemo_hdlwa_ctl. Она содержит алгоритмы, которые будут реализованы на ПЛИС. Эта часть модели называется DUT (Device Under Test).
Цифровые входы-выходы нужно настроить путём управления микросхемой на FMC субмодуле через входной сигнал FMC_Control_PCI подсистемы DUT_ritmdemo_hdlwa_ctl. Можно управлять направлением, уровнем напряжения цифровых входов-выходов и светодиодом на клеммном модуле. Выходной сигнал GPIO_CTL подсистемы DUT будет автоматически подключен к фиксированным пинам ПЛИС в проекте.
Входной сигнал FMC_Power_PCI подсистемы DUT управляет включением или выключением питания субмодуля FMC. Выходной сигнал FMC_Power автоматически привязывается к управляющему пину ПЛИС.
Вторая часть - это всё, кроме этой подсистемы. Она содержит алгоритмы, которые будут реализованы на процессоре. Эта часть модели называется программная модель или SW.

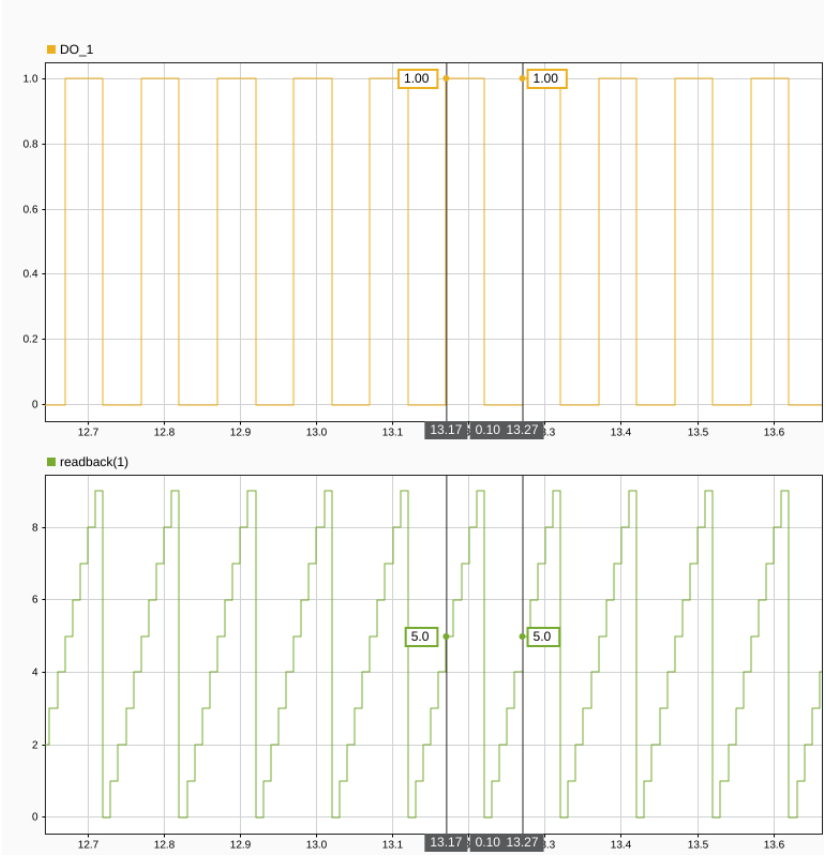
Используя блоки period, manual_override и manual_value в программной модели, мы управляем меандром из процессора.
Используя блоки Dir, Vref_5V, LED_Blue в программной модели, мы управляем направлением, уровнем напряжения цифровых входов-выходов и светодиодом на клеммном модуле.
Используя блок FMC_Power, мы управляем включением питания субмодуля FMC. По умолчанию питание включено.

Внутри DUT находится логика изменения периода меандра в ПЛИС:



Входной сигнал period_PCI будет приходить в ПЛИС со стороны процессора по PCIe, а выходной сигнал DO будет подключен к FMC субмодулю и внешним цифровым выходам.

До реализации модели в железе её требуется проверить при помощи симуляции. Можно запустить симуляцию модели, поменять значение периода меандра (заданного в тактах ПЛИС) и увидеть на графиках значения цифрового выхода, на который поступает меандр. Для отладки на графики выводится значения половины периода меандра и внутреннего счетчика.



Генерация прошивки для ПЛИС

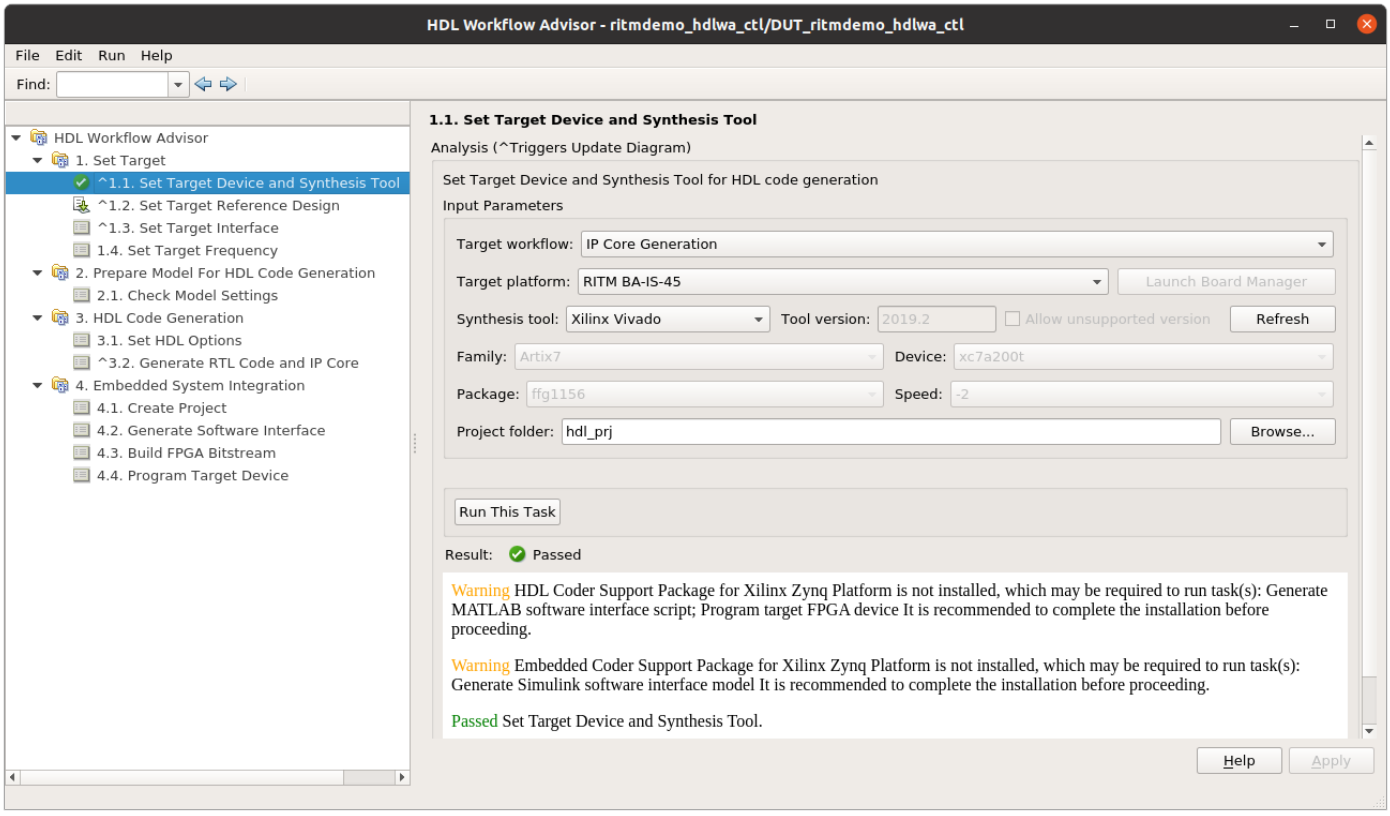
Рассмотрим автоматическую генерацию прошивки ПЛИС из алгоритма в подсистеме DUT.

Запустите HDL Workflow Advisor (HDLWA), щелкнув правой кнопкой мыши по подсистеме DUT_ritmdemo_hdlwa_ctl и выбрав HDL Code -> HDL Workflow Advisor.

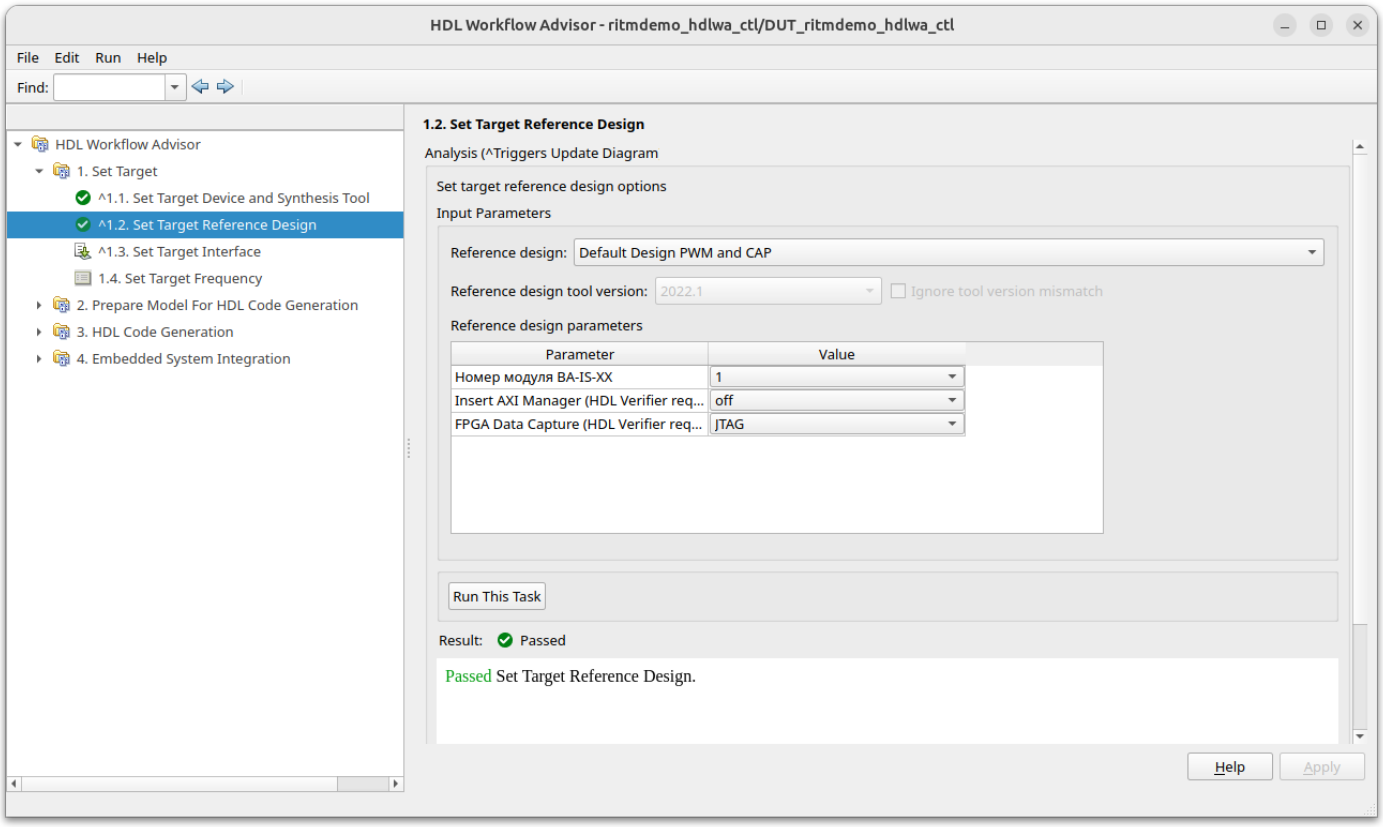
Назначение интерфейсов DUT к интерфейсам на ПЛИС

Рассмотрим привязку интерфейсов в модели DUT к реальным интерфейсам на ПЛИС.

- На шаге 1.1 выставите настройки и нажмите Run This Task:



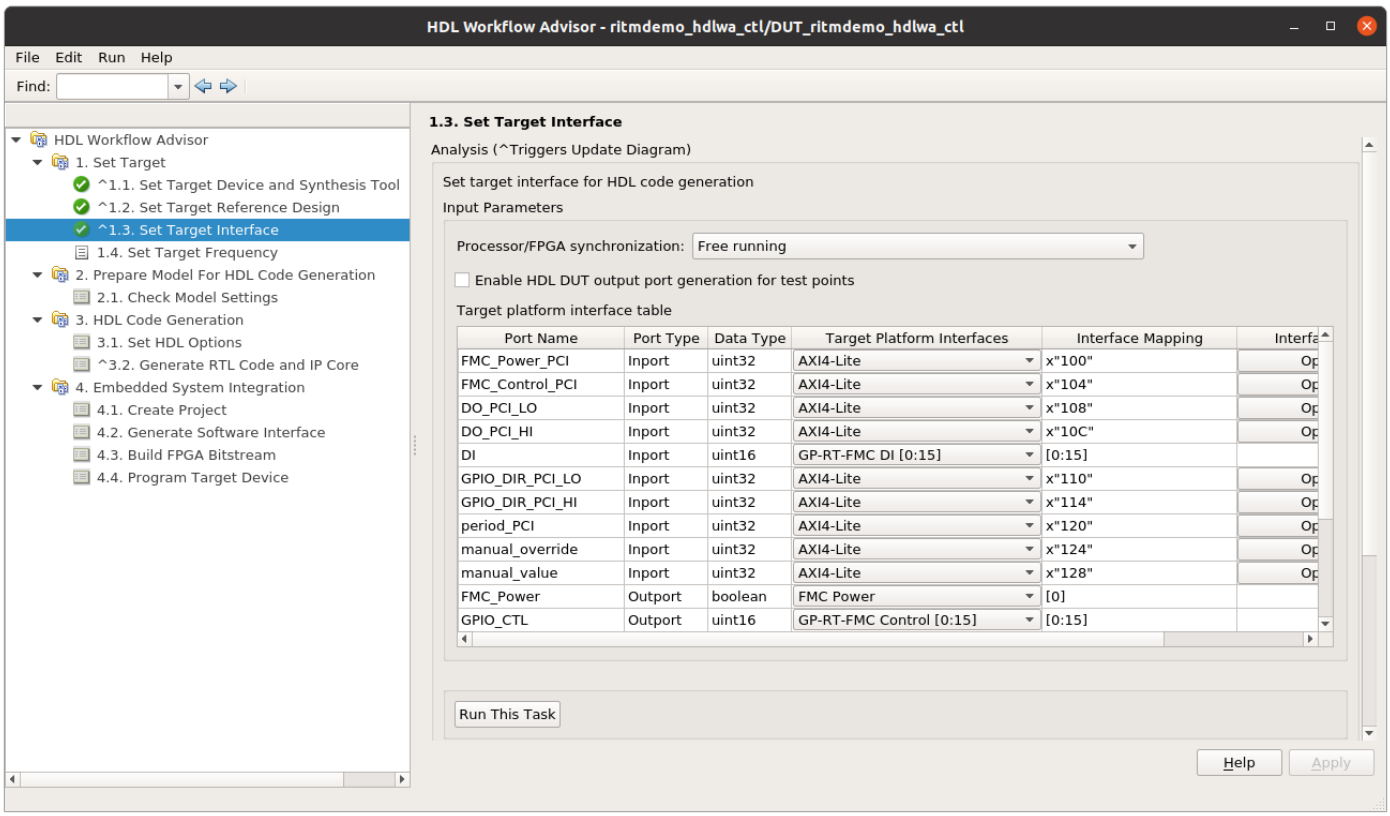
- На шаге 1.2 выставите настройки и нажмите Run This Task:

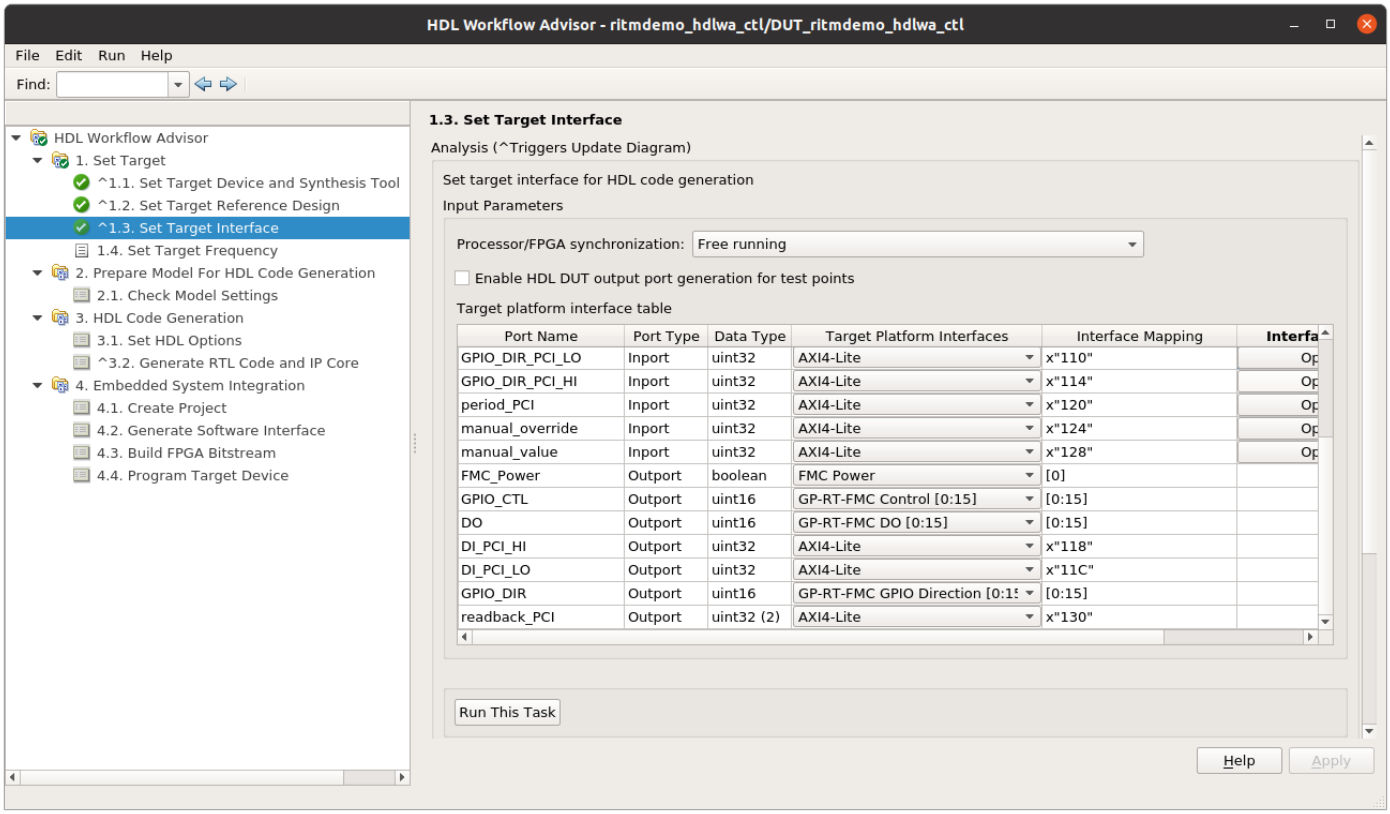


- Номер модуля BA-IS-XX - Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.

На данном шаге выбирается конфигурация внешних интерфейсов РИТМ, и какие IP ядра будут задействованы.

- На шаге 1.3 настройте привязку интерфейсов DUT к интерфейсам ПЛИС:

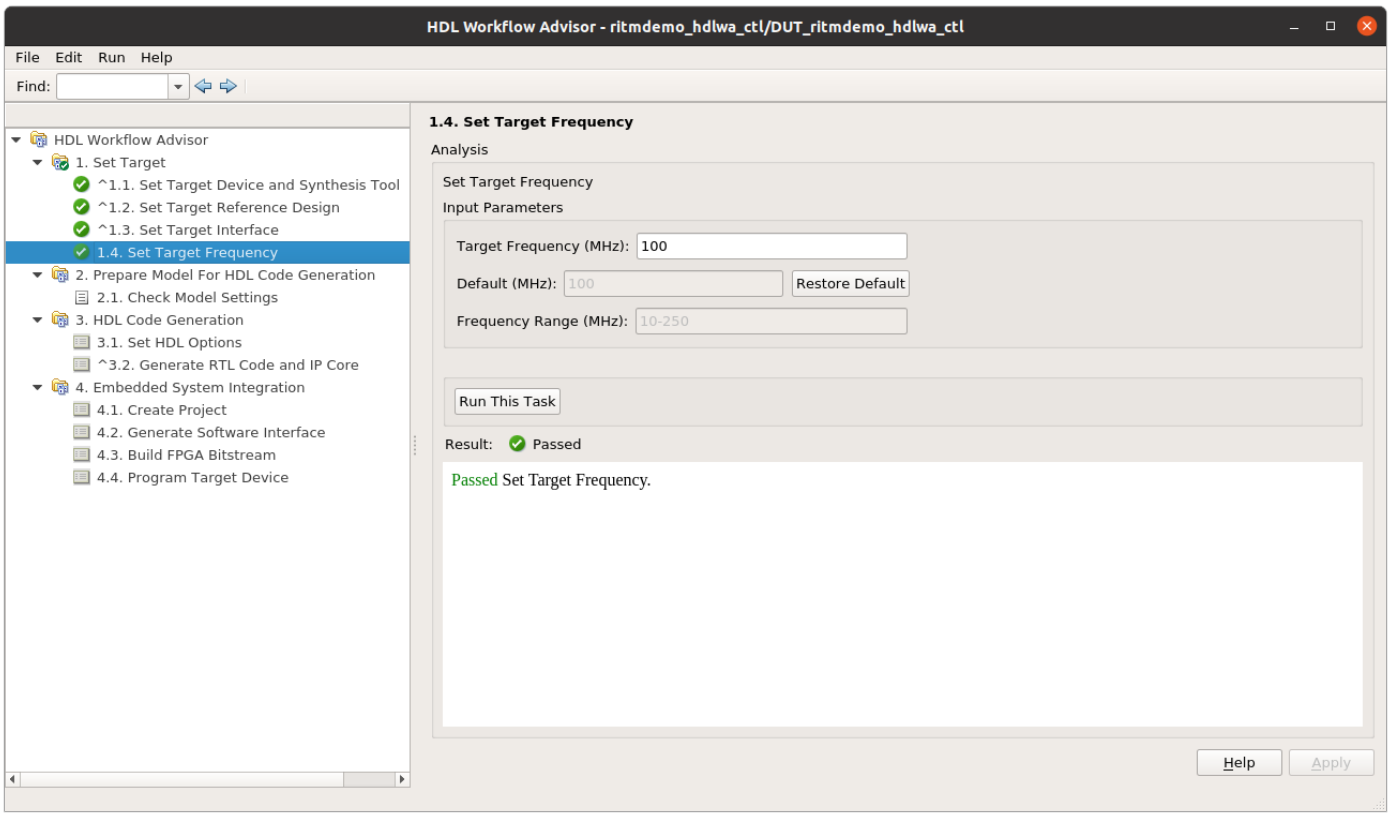




При работе с FMC модулем используются интерфейсы:
- AXI4-Lite - Интерфейс для обмена данными между процессором и ПЛИС через PCIe; - GP-RT-FMC Control - Интерфейс для управления портом специального назначения;
- GP-RT-FMC Direction - Порт задающий направление (вход/выход);
- GP-RT-FMC DO - Выходной порт общего назначения;
- GP-RT-FMC DI - Входной порт общего назначения;
- FMC-POWER - Питание модуля FMC.

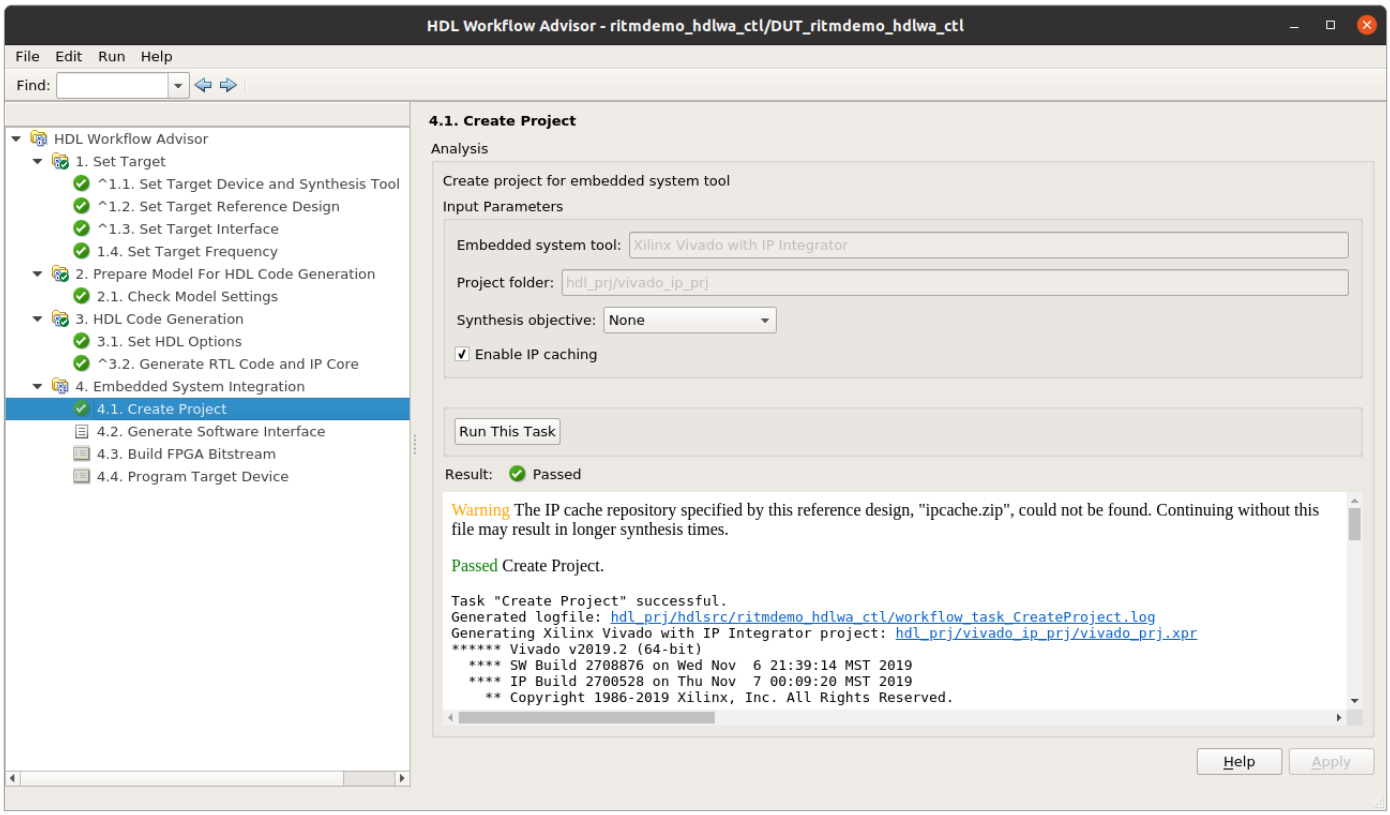
При назначении интерфейса происходит автоматическое назначение адреса порта.

- На шаге 1.4 настройте тактовую частоту DUT:

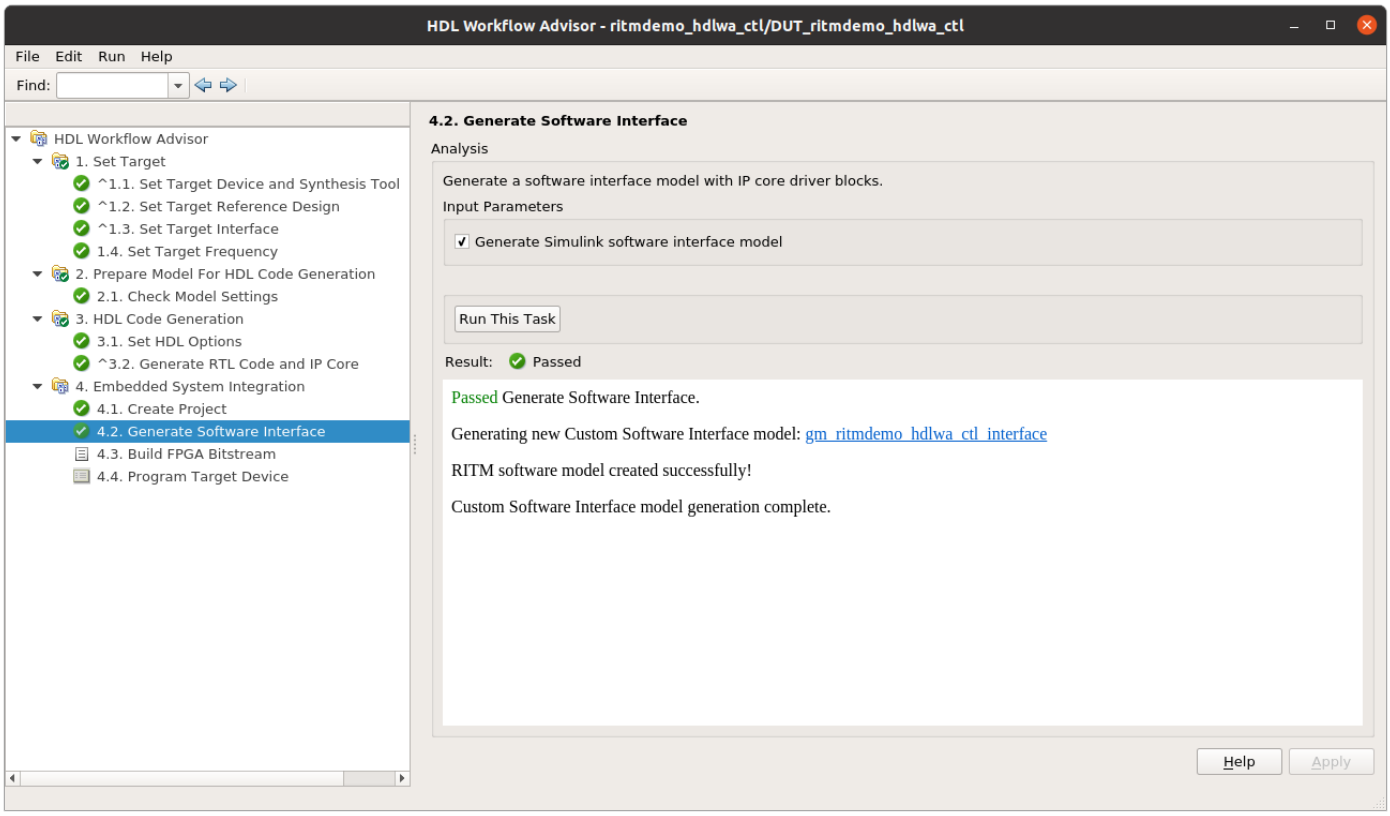


Генерация HDL кода и интеграция с проектом Xilinx Vivado

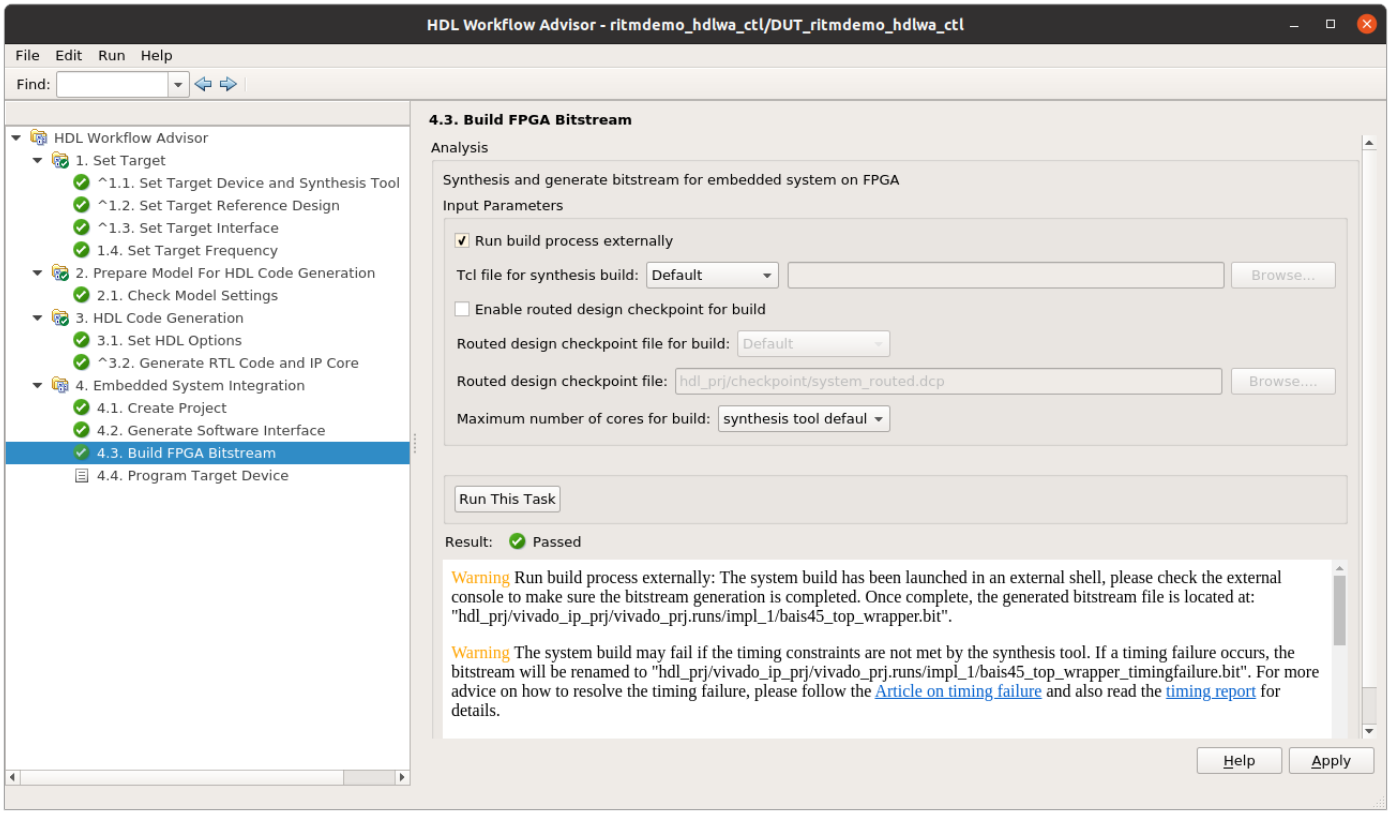
- На шагах 2 и 3 проверяются настройки модели и предоставляется возможность настроить параметры генерации кода.
- На шаге 4.1 создается проект в Xilinx Vivado. Щелкните правой кнопкой мыши по шагу 4.1 и выберите Run to Selected Task:



- На шаге 4.2 создается программная модель. Больше информации о программной модели в разделе Обзор программной модели:

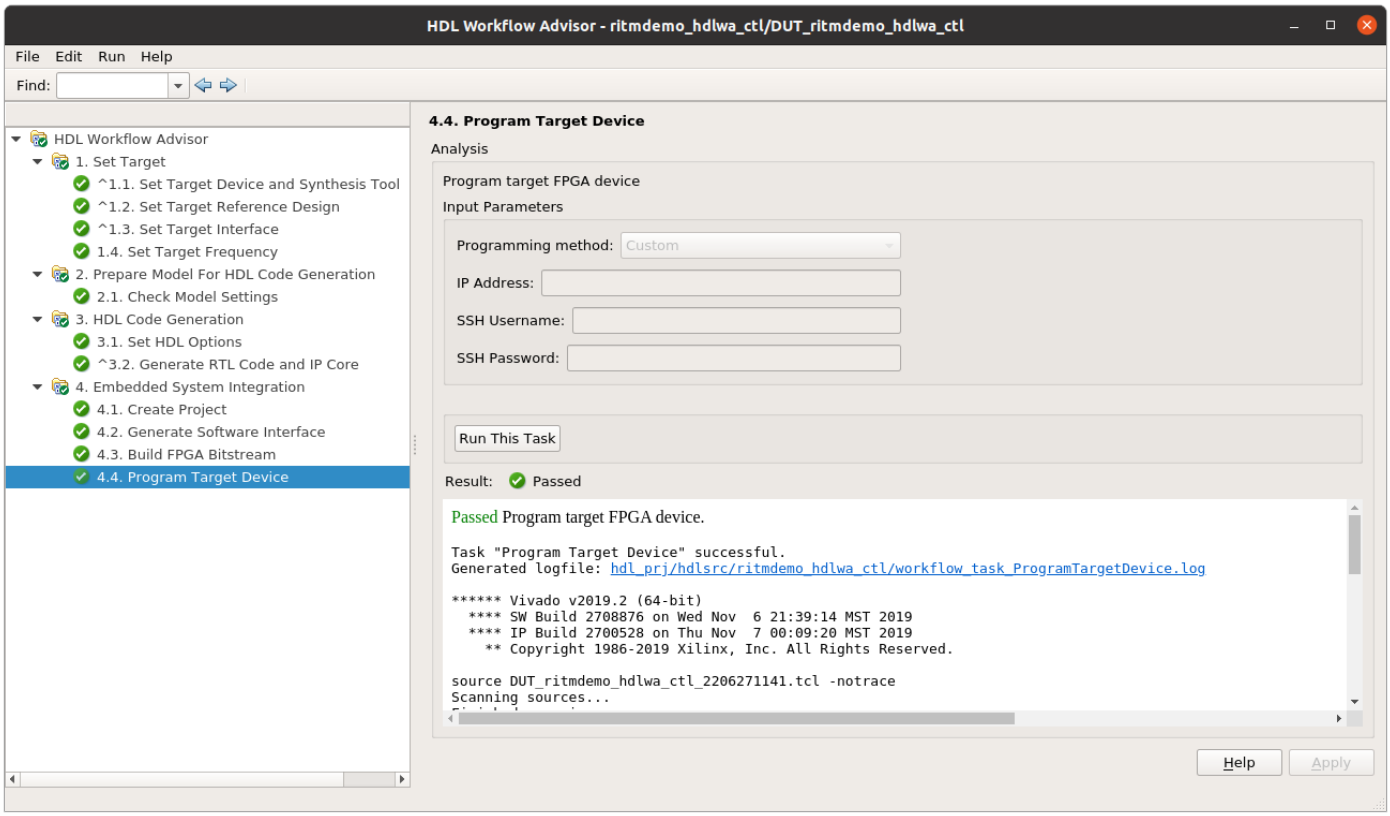


- На шаге 4.3 происходит генерация битового потока:



При включенной опции Run build process externally запуск сборки происходит параллельно с работой MATLAB. При выключенной опции Вы не сможете использовать MATLAB, пока сборка не будет завершена.

- На шаге 4.4 создается прошивка для программирования ПЛИС:



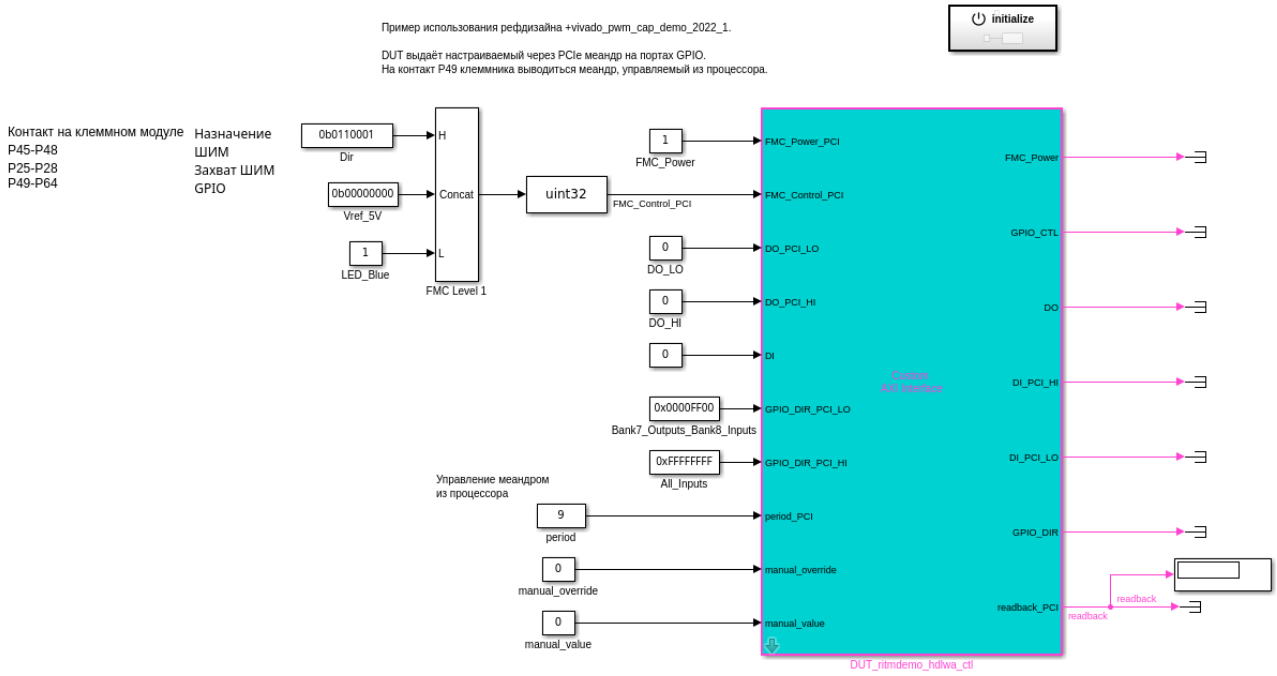
На данном шаге генерируются файлы с расширением .mat и .mcs (прошивка), но не загружаются на ПЛИС. Они необходимы для запуска модели на РИТМ.

Этим завершается обзор процесса генерации прошивки ПЛИС из подсистемы DUT.

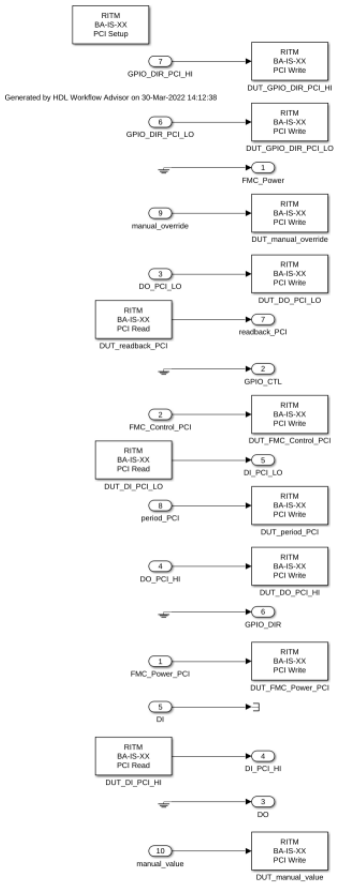
Обзор программной модели

На шаге 4.2 инструмента HDL Workflow Advisor автоматически создается программная модель. Эта модель представляет собой копию оригинальной модели, но с заменой всей подсистемы DUT на специальный блок. Подсистема Initialize добавлена вручную, содержит

сброс IP ядра DUT. Подсистема сбрасывает счетчик в DUT, между запусками модели, для корректной работы:



Таким образом, программная модель содержит в себе блоки и алгоритмы, которые будут запущены на CPU. Кроме того, она содержит сгенерированный блок, который представляет собой интерфейсы между CPU и DUT:



Программную модель нужно запустить на РИТМе, при этом ПЛИС будет прошита автоматически. Последующие запуски одной и той же программной модели не будут приводить к изменению прошивки ПЛИС.

Ознакомьтесь с программной моделью, сгенерированной в этом примере.
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_ctl_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_ctl_BA_IS_55_interface.slx')
```

Подключение аналоговых входов-выходов к алгоритму на ПЛИС

Содержание

- Требования к аппаратному обеспечению
- Требования к программному обеспечению
- Общая структура системы
- Пример использования
  - Демо модель
  - Генерация прошивки для ПЛИС
  - Обзор программной модели

В этом разделе объясняется, как подключить аналоговые входы и выходы к алгоритму, работающему на ПЛИС.

Требования к аппаратному обеспечению

- Установленная базовая плата RITMeX BA-IS-45 или RITMeX BA-IS-55
- FMC расширение для базовой платы ПЛИС RITMeX DA-RT-FMC Level 1

Требования к программному обеспечению

- HDL Coder для генерации кода
- Fixed-Point Designer для работы с HDL Coder
- Xilinx Vivado для программирования ПЛИС:
  - Для программирования BA-IS-55 требуется версия Xilinx Vivado с поддержкой Ultrascale
  - Для программирования BA-IS-45 требуется версия Xilinx Vivado с поддержкой 7 Series
  - Работа верифицирована на версии Xilinx Vivado 2022.1

Общая структура системы

Для лучшего понимания взаимодействия железа и софта обратитесь к разделу Общие сведения о запуске алгоритмов на ПЛИС

Пример использования

В качестве примера рассмотрим программно-аппаратную реализацию сигналов разного уровня напряжения. Логика формирования будет реализована в программной части (SW) в виде синусоидального сигнала меняющейся амплитуды. Сигнал будет оцифрован и возвращен обратно в SW по AXI4-Lite интерфейсу.

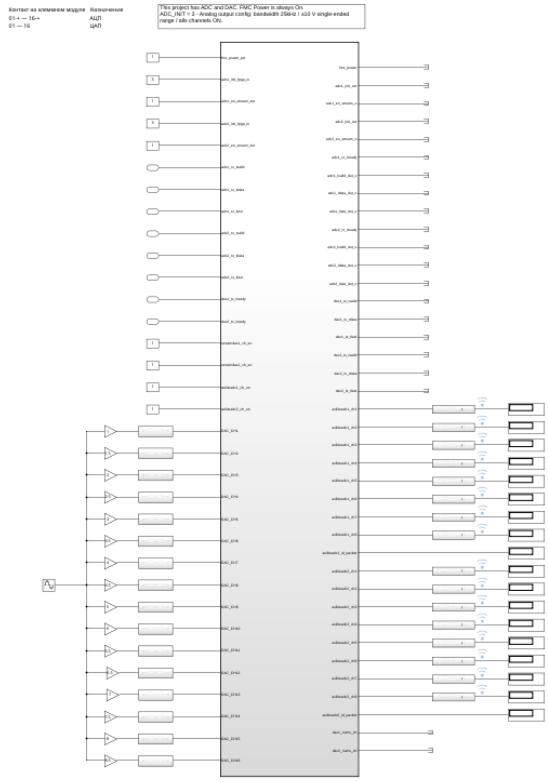
Демо модель

Рассмотрим демо модель.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_analogio.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_analogio_BA_IS_55.slx')
```



Подсистема DUT_ritmdemo_hdlwa_analogio содержит алгоритмы, которые будут реализованы на ПЛИС. Эта часть модели называется DUT (Device Under Test).
Аналоговые входы нужно настроить путём управления микросхемой на FMC субмодуле через входные сигналы adc1_init_fpga_in, adc2_init_fpga_in подсистемы DUT_ritmdemo_hdlwa_analogio. Аналоговые выходы автоматически настраиваются и дополнительного управления не требуют. Так же возможно включать-выключать передачу и приём данных в DUT: adc1_en_stream_dut, adc2_en_stream_dut, streamdac1_ch_en и т.д. Входной сигнал fmc_power_pci подсистемы DUT управляет включением или выключением питания субмодуля FMC. Выходной сигнал fmc_power автоматически привязывается к управляющему пину ПЛИС.
Всё, кроме подсистемы DUT, содержит алгоритмы, которые будут реализованы на процессоре. Эта часть модели называется программная модель или SW.

Рассмотрим аналоговые входы и их инциализацию. Используя блоки ADC1_INIT, ADC2_INIT, в программной модели, мы управляем инициализацией аналоговых входов из процессора.
В каждом номере инициализации аналоговых входов будет: тип интерфейса AXI4-Stream, синхронизация между каналами АЦП включена, полоса фильтра 25kHz, все каналы задействованы. Частота дискретизации аналоговых входов 1 МГц для каждого канала. AXI4-Stream между аналоговыми входами и DUT работает на частоте 125 МГц.

Номера инициализаций аналоговых входов:

- ADC_INIT = 0 - ±2.5 V single-ended диапазон;
- ADC_INIT = 1 - ±5 V single-ended диапазон;
- ADC_INIT = 2 - ±6.25 V single-ended диапазон;
- ADC_INIT = 3 - ±10 V single-ended диапазон;
- ADC_INIT = 4 - ±12.5 V single-ended диапазон;
- ADC_INIT = 5 - 0 V to 5 V single-ended диапазон;
- ADC_INIT = 6 - 0 V to 10 V single-ended диапазон;
- ADC_INIT = 7 - 0 V to 12.5 V single-ended диапазон;
- ADC_INIT = 8 - ±5 V differential диапазон;
- ADC_INIT = 9 - ±10 V differential диапазон;
- ADC_INIT = 10 - ±12.5 V differential диапазон;
- ADC_INIT = 11 - ±20 V differential диапазон;
- ADC_INIT = others - не используются;

В примере рассматривается вариант с ADC_INIT = 3, аналоговые входы работают в ±10 V single-ended диапазоне. Данные по AXI4-Stream приходят в виде отчётов АЦП. Чтобы получить данные для каждого канала отдельно, в DUT есть подсистема конвертации в каналы. Выход подсистемы содержит восемь каналов. В примере данные выведены в часть SW через AXI4-Lite и конвертированы в вольты. Для разных режимов инициализации в подсистеме 10VScaleCH* для конвертации нужно изменить константу scale_one_volt_+-10range в зависимости от номера инициализации:

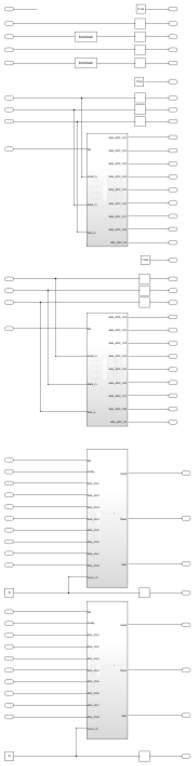
- ADC_INIT = 0 - 13107;
- ADC_INIT = 1 - 6553.5;
- ADC_INIT = 2 - 5242.8;
- ADC_INIT = 3 - 3276.75;
- ADC_INIT = 4 - 2621.4;
- ADC_INIT = 5 - 13107;
- ADC_INIT = 6 - 6553.5;
- ADC_INIT = 7 - 5242.8;

- ADC_INIT = 8 - 6553.5;
- ADC_INIT = 9 - 3276.75;
- ADC_INIT = 10 - 2621.4;
- ADC_INIT = 11 - 1638.375;

Рассмотрим управление аналоговыми выходами. Частота дискретизации аналоговых выходов зависит от количества задействованных каналов (от 1 до 8), времени записи на аналоговый выход (от 2.5 мкс до 30 мкс), размаха сигнала (от -10 В до +10 В), времени установления значения на аналоговых выходах (1 мкс/В) и синхронности обновления каналов (от 0 мкс до 6.3 мкс). AXI4-Stream между DUT и аналоговыми выходами работает на частоте 125 МГц. Всё это необходимо учитывать при расчёте частоты дискретизации для формирования сложных сигналов на аналоговые выходы.

Используя блок FMC_Power, мы управляем включением питания субмодуля FMC. По умолчанию питание включено.

Внутри DUT находится логика отправки и приёма данных от аналоговых входов-выходов в ПЛИС:



Логика содержит формирование AXI4-Stream для аналоговых выходов-выходов. До реализации модели в железе её требуется проверить при помощи симуляции.

Генерация прошивки для ПЛИС

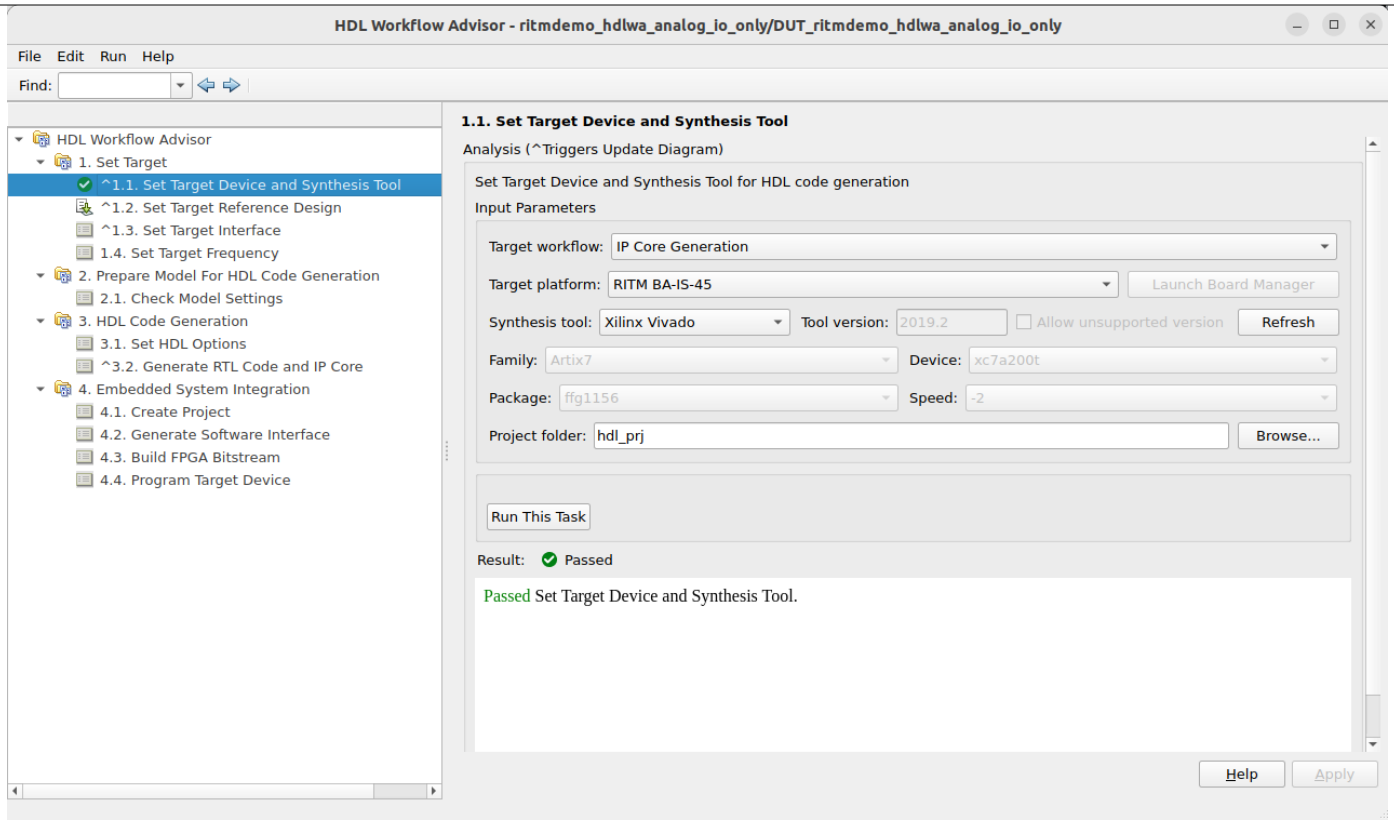
Рассмотрим автоматическую генерацию прошивки ПЛИС из алгоритма в подсистеме DUT.

Запустите HDL Workflow Advisor (HDLWA), щелкнув правой кнопкой мыши по подсистеме DUT_ritmdemo_hdlwa_analogio и выбрав HDL Code -> HDL Workflow Advisor.

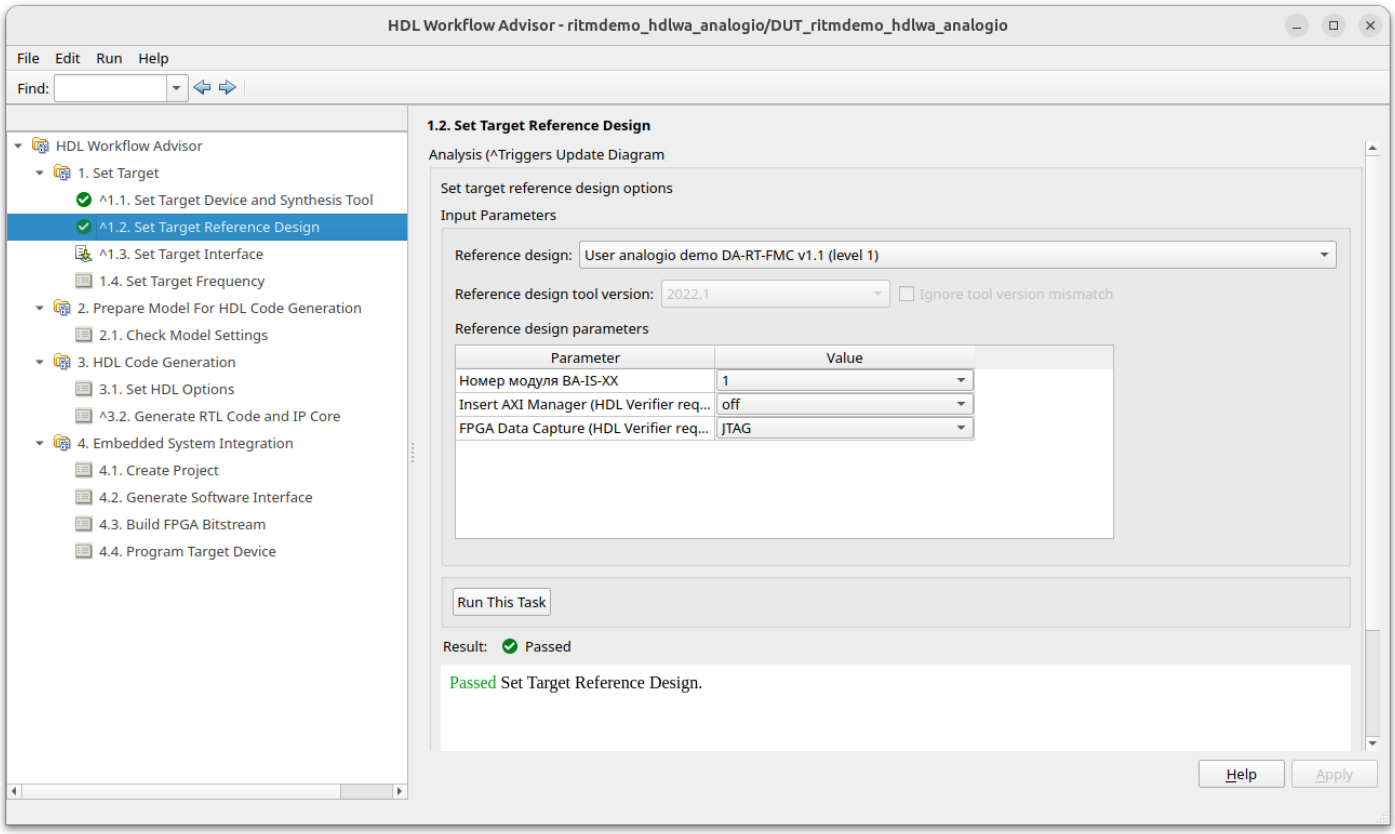
Назначение интерфейсов DUT к интерфейсам на ПЛИС

Рассмотрим привязку интерфейсов в модели DUT к реальным интерфейсам на ПЛИС.

- На шаге 1.1 выставите настройки и нажмите Run This Task:



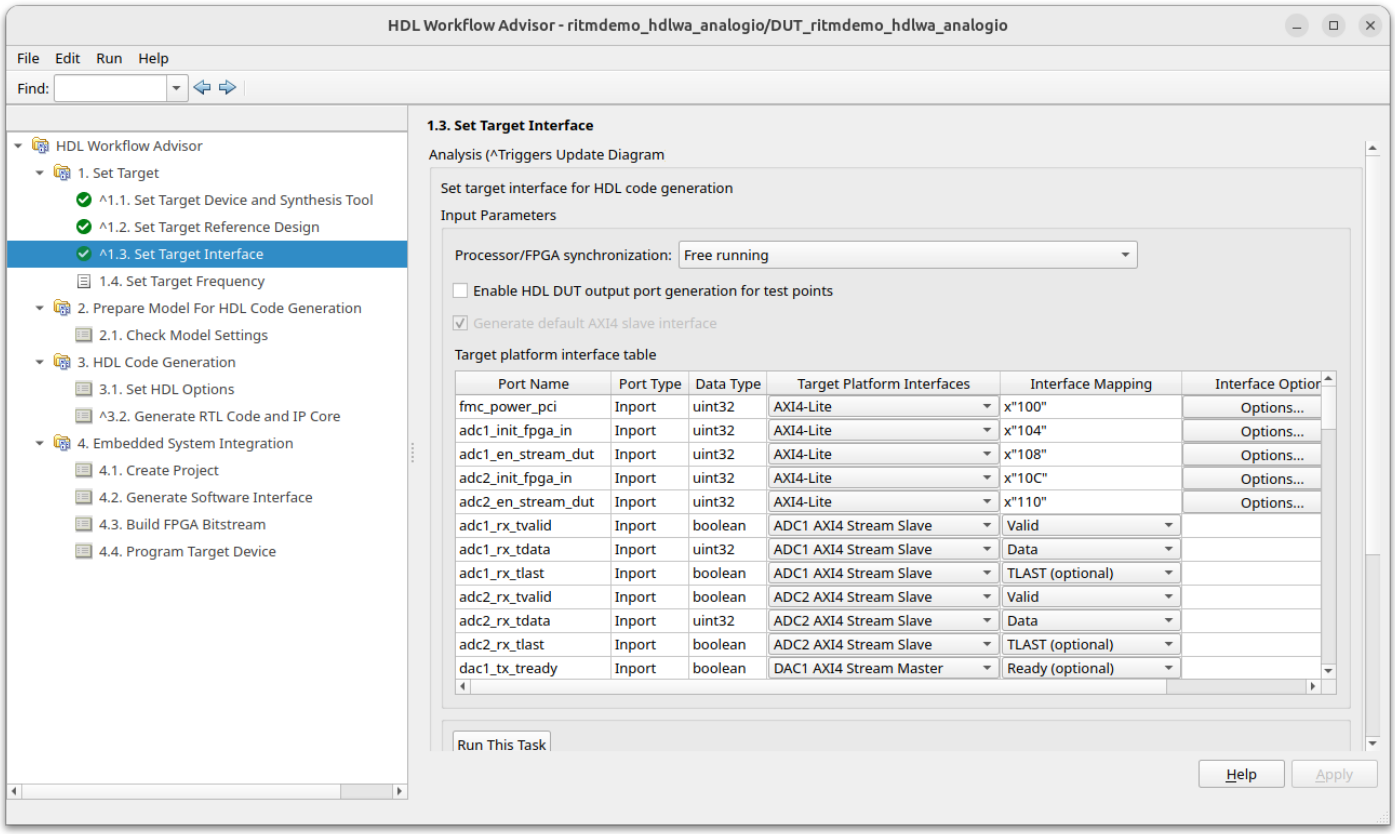
- На шаге 1.2 выставите настройки и нажмите Run This Task:

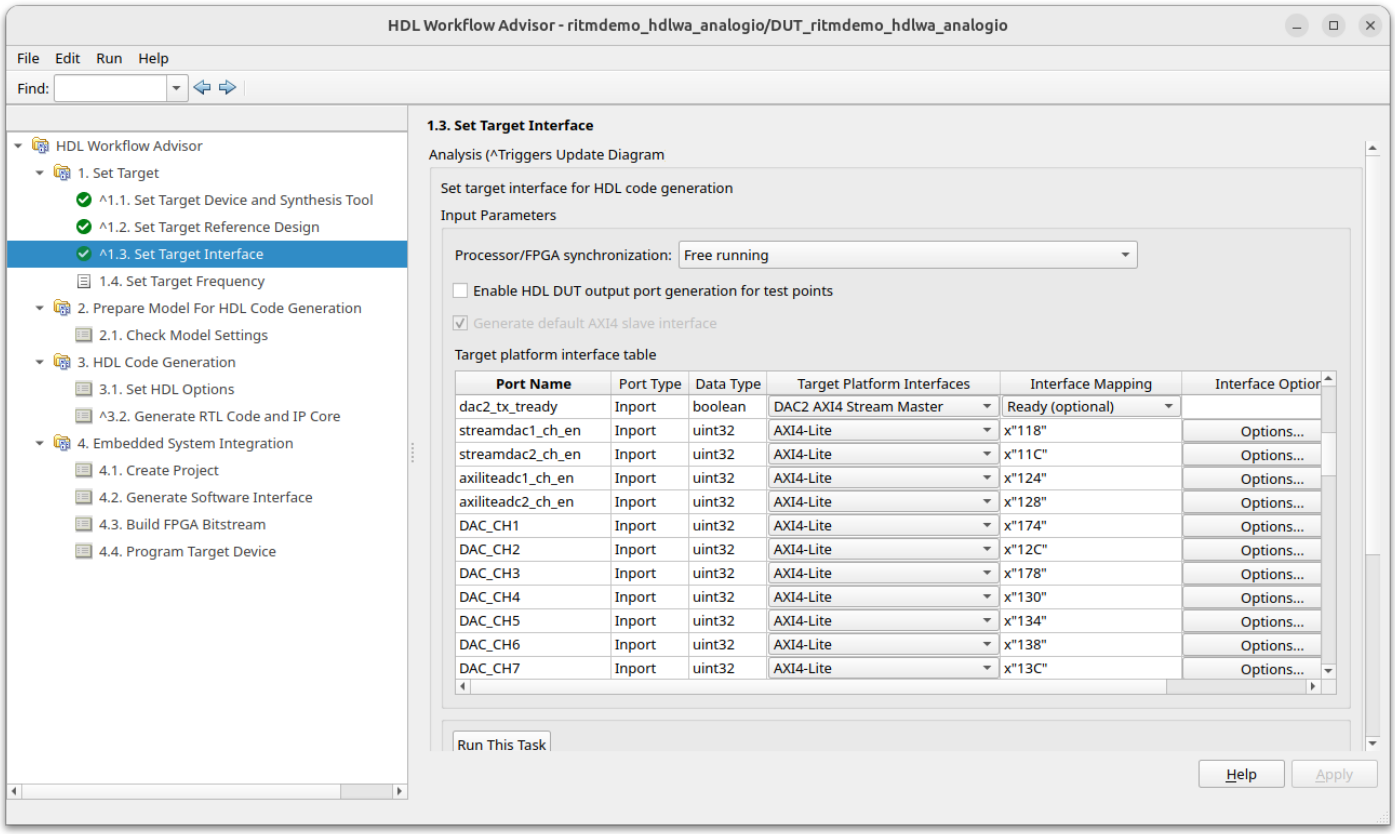


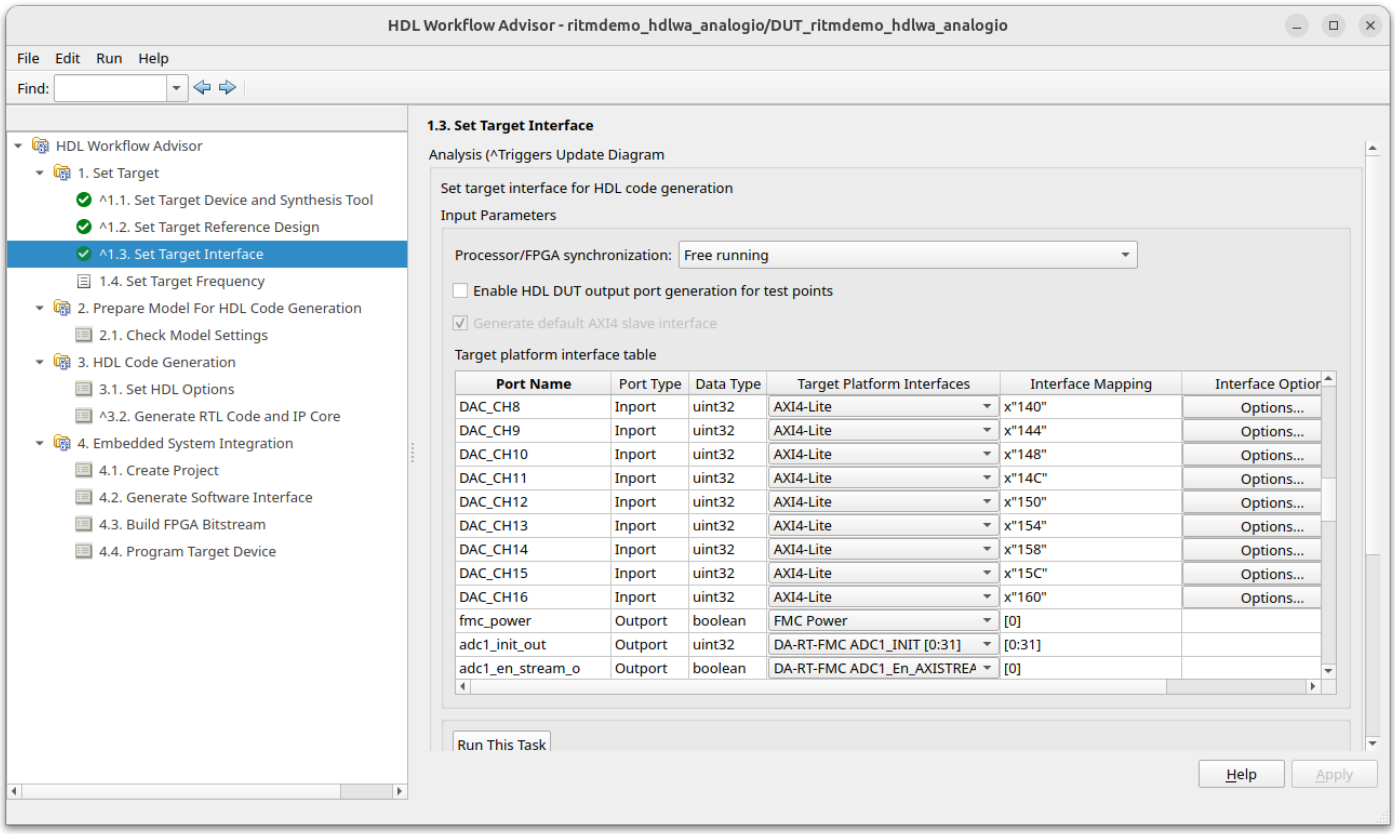
- Номер модуля BA-IS-XX - Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.

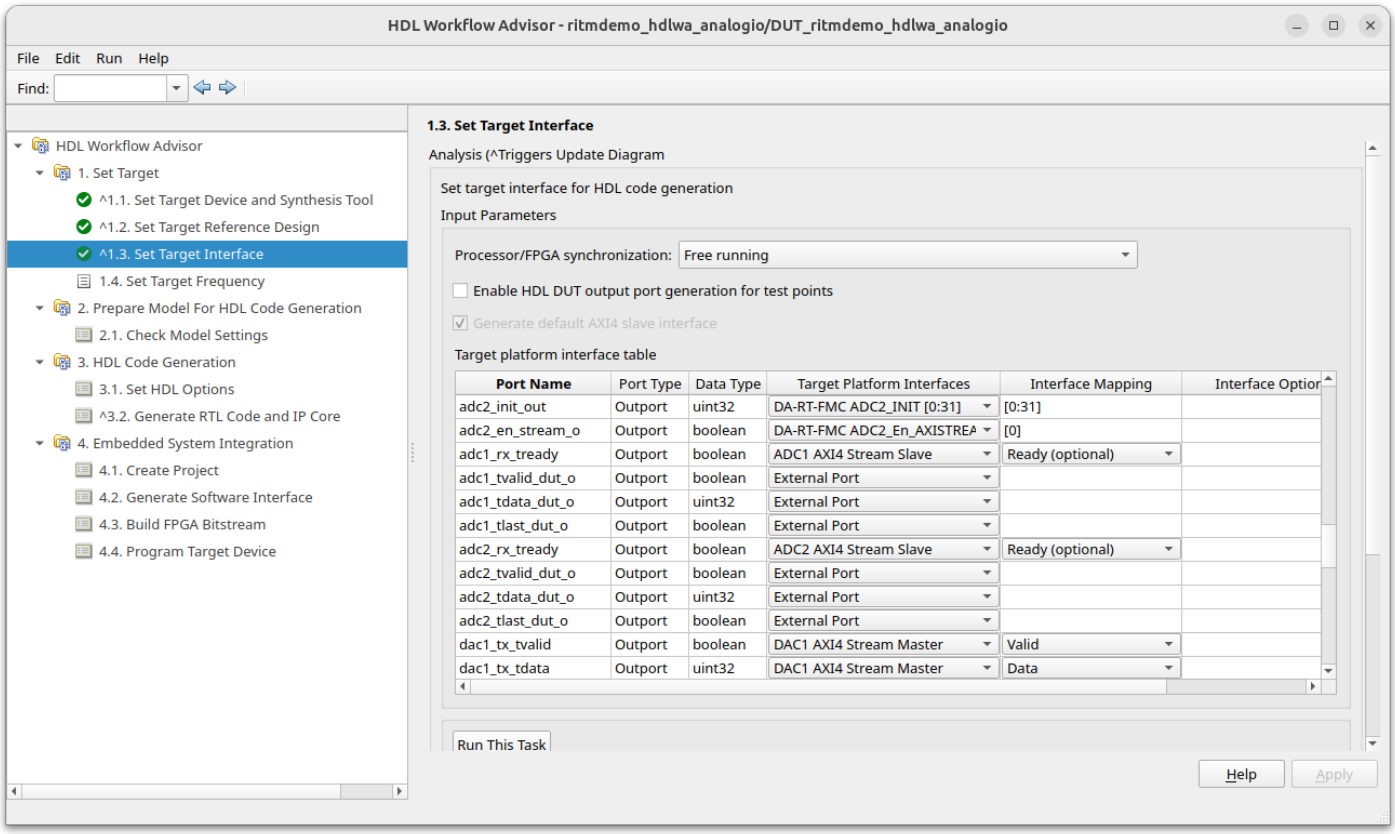
На данном шаге выбирается конфигурация внешних интерфейсов РИТМ, и какие IP ядра будут задействованы.

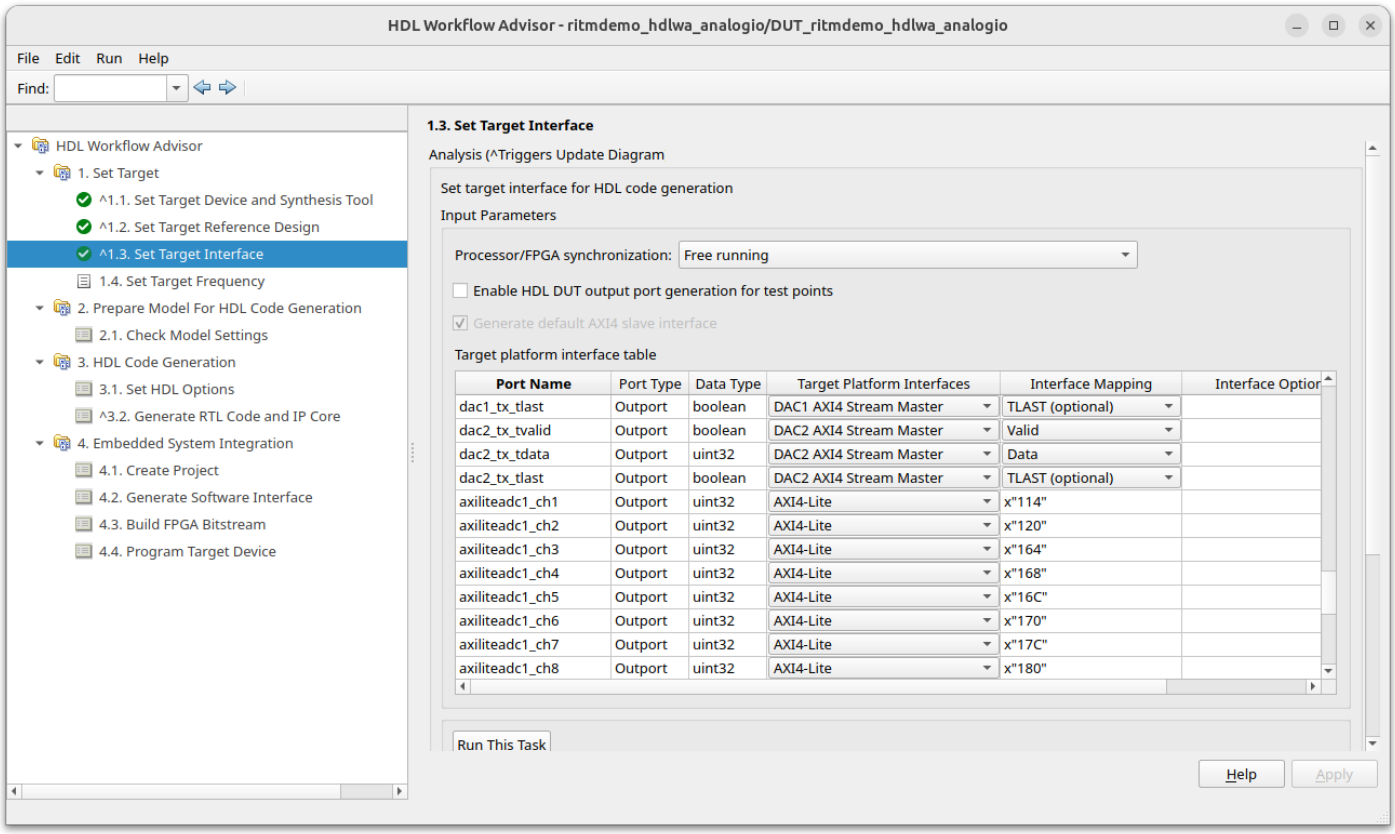
- На шаге 1.3 настройте привязку интерфейсов DUT к интерфейсам ПЛИС:

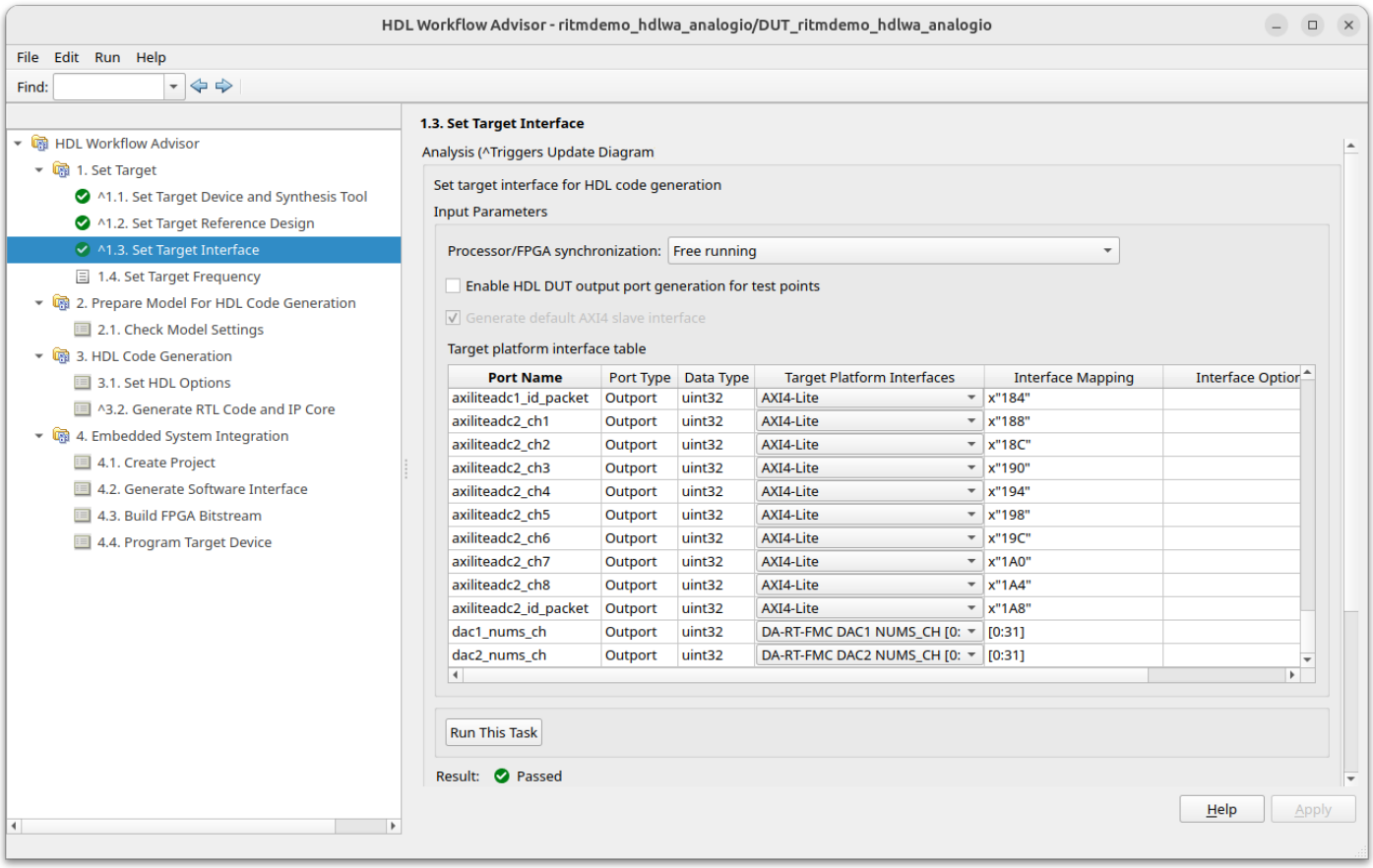

HDL Workflow Advisor - ritmdemo_hdlwa_analogio/DUT_ritmdemo_hdlwa_analogio
File Edit Run Help
Find:
HDL Workflow Advisor
1. Set Target
^1.1. Set Target Device and Synthesis Tool
^1.2. Set Target Reference Design
^1.3. Set Target Interface
1.4. Set Target Frequency
2. Prepare Model For HDL Code Generation
2.1. Check Model Settings
3. HDL Code Generation
3.1. Set HDL Options
^3.2. Generate RTL Code and IP Core
4. Embedded System Integration
4.1. Create Project
4.2. Generate Software Interface
4.3. Build FPGA Bitstream
4.4. Program Target Device
1.3. Set Target Interface
Analysis (^Triggers Update Diagram
Set target interface for HDL code generation
Input Parameters
Processor/FPGA synchronization: Free running
Enable HDL DUT output port generation for test points
Generate default AXI4 slave interface
Target platform interface table
Port Name | Port Type | Data Type | Target Platform Interfaces | Interface Mapping | Interface Optior
fmc_power_pci | Inport | uint32 | AXI4-Lite | x"100" | Options...
adc1_init_fpga_in | Inport | uint32 | AXI4-Lite | x"104" | Options...
adc1_en_stream_dut | Inport | uint32 | AXI4-Lite | x"108" | Options...
adc2_init_fpga_in | Inport | uint32 | AXI4-Lite | x"10C" | Options...
adc2_en_stream_dut | Inport | uint32 | AXI4-Lite | x"110" | Options...
adc1_rx_tvalid | Inport | boolean | ADC1 AXI4 Stream Slave | Valid
adc1_rx_tdata | Inport | uint32 | ADC1 AXI4 Stream Slave | Data
adc1_rx_tlast | Inport | boolean | ADC1 AXI4 Stream Slave | TLAST (optional)
adc2_rx_tvalid | Inport | boolean | ADC2 AXI4 Stream Slave | Valid
adc2_rx_tdata | Inport | uint32 | ADC2 AXI4 Stream Slave | Data
adc2_rx_tlast | Inport | boolean | ADC2 AXI4 Stream Slave | TLAST (optional)
dac1_tx_tready | Inport | boolean | DAC1 AXI4 Stream Master | Ready (optional)
Run This Task
Help
Apply


HDL Workflow Advisor - ritmdemo_hdlwa_analogio/DUT_ritmdemo_hdlwa_analogio
File Edit Run Help
Find:
HDL Workflow Advisor
1. Set Target
^1.1. Set Target Device and Synthesis Tool
^1.2. Set Target Reference Design
^1.3. Set Target Interface
1.4. Set Target Frequency
2. Prepare Model For HDL Code Generation
2.1. Check Model Settings
3. HDL Code Generation
3.1. Set HDL Options
^3.2. Generate RTL Code and IP Core
4. Embedded System Integration
4.1. Create Project
4.2. Generate Software Interface
4.3. Build FPGA Bitstream
4.4. Program Target Device
1.3. Set Target Interface
Analysis (^Triggers Update Diagram
Set target interface for HDL code generation
Input Parameters
Processor/FPGA synchronization: Free running
Enable HDL DUT output port generation for test points
Generate default AXI4 slave interface
Target platform interface table
Port Name | Port Type | Data Type | Target Platform Interfaces | Interface Mapping | Interface Optior
dac2_tx_tready | Inport | boolean | DAC2 AXI4 Stream Master | Ready (optional)
streamdac1_ch_en | Inport | uint32 | AXI4-Lite | x"118" | Options...
streamdac2_ch_en | Inport | uint32 | AXI4-Lite | x"11C" | Options...
axiliteadc1_ch_en | Inport | uint32 | AXI4-Lite | x"124" | Options...
axiliteadc2_ch_en | Inport | uint32 | AXI4-Lite | x"128" | Options...
DAC_CH1 | Inport | uint32 | AXI4-Lite | x"174" | Options...
DAC_CH2 | Inport | uint32 | AXI4-Lite | x"12C" | Options...
DAC_CH3 | Inport | uint32 | AXI4-Lite | x"178" | Options...
DAC_CH4 | Inport | uint32 | AXI4-Lite | x"130" | Options...
DAC_CH5 | Inport | uint32 | AXI4-Lite | x"134" | Options...
DAC_CH6 | Inport | uint32 | AXI4-Lite | x"138" | Options...
DAC_CH7 | Inport | uint32 | AXI4-Lite | x"13C" | Options...
Run This Task
Help
Apply


HDL Workflow Advisor - ritmdemo_hdlwa_analogio/DUT_ritmdemo_hdlwa_analogio
File Edit Run Help
Find:
HDL Workflow Advisor
1. Set Target
^1.1. Set Target Device and Synthesis Tool
^1.2. Set Target Reference Design
^1.3. Set Target Interface
1.4. Set Target Frequency
2. Prepare Model For HDL Code Generation
2.1. Check Model Settings
3. HDL Code Generation
3.1. Set HDL Options
^3.2. Generate RTL Code and IP Core
4. Embedded System Integration
4.1. Create Project
4.2. Generate Software Interface
4.3. Build FPGA Bitstream
4.4. Program Target Device
1.3. Set Target Interface
Analysis (^Triggers Update Diagram
Set target interface for HDL code generation
Input Parameters
Processor/FPGA synchronization: Free running
Enable HDL DUT output port generation for test points
Generate default AXI4 slave interface
Target platform interface table
Port Name | Port Type | Data Type | Target Platform Interfaces | Interface Mapping | Interface Optior
DAC_CH8 | Inport | uint32 | AXI4-Lite | x"140" | Options...
DAC_CH9 | Inport | uint32 | AXI4-Lite | x"144" | Options...
DAC_CH10 | Inport | uint32 | AXI4-Lite | x"148" | Options...
DAC_CH11 | Inport | uint32 | AXI4-Lite | x"14C" | Options...
DAC_CH12 | Inport | uint32 | AXI4-Lite | x"150" | Options...
DAC_CH13 | Inport | uint32 | AXI4-Lite | x"154" | Options...
DAC_CH14 | Inport | uint32 | AXI4-Lite | x"158" | Options...
DAC_CH15 | Inport | uint32 | AXI4-Lite | x"15C" | Options...
DAC_CH16 | Inport | uint32 | AXI4-Lite | x"160" | Options...
fmc_power | Outport | boolean | FMC Power | [0]
adc1_init_out | Outport | uint32 | DA-RT-FMC ADC1_INIT [0:31] | [0:31]
adc1_en_stream_o | Outport | boolean | DA-RT-FMC ADC1_En_AXISTREA | [0]
Run This Task
Help
Apply


HDL Workflow Advisor - ritmdemo_hdlwa_analogio/DUT_ritmdemo_hdlwa_analogio
File Edit Run Help
Find:
HDL Workflow Advisor
1. Set Target
^1.1. Set Target Device and Synthesis Tool
^1.2. Set Target Reference Design
^1.3. Set Target Interface
1.4. Set Target Frequency
2. Prepare Model For HDL Code Generation
2.1. Check Model Settings
3. HDL Code Generation
3.1. Set HDL Options
^3.2. Generate RTL Code and IP Core
4. Embedded System Integration
4.1. Create Project
4.2. Generate Software Interface
4.3. Build FPGA Bitstream
4.4. Program Target Device
1.3. Set Target Interface
Analysis (^Triggers Update Diagram
Set target interface for HDL code generation
Input Parameters
Processor/FPGA synchronization: Free running
Enable HDL DUT output port generation for test points
Generate default AXI4 slave interface
Target platform interface table
Port Name | Port Type | Data Type | Target Platform Interfaces | Interface Mapping | Interface Optior
adc2_init_out | Outport | uint32 | DA-RT-FMC ADC2_INIT [0:31] | [0:31]
adc2_en_stream_o | Outport | boolean | DA-RT-FMC ADC2_En_AXISTREA | [0]
adc1_rx_tready | Outport | boolean | ADC1 AXI4 Stream Slave | Ready (optional)
adc1_tvalid_dut_o | Outport | boolean | External Port
adc1_tdata_dut_o | Outport | uint32 | External Port
adc1_tlast_dut_o | Outport | boolean | External Port
adc2_rx_tready | Outport | boolean | ADC2 AXI4 Stream Slave | Ready (optional)
adc2_tvalid_dut_o | Outport | boolean | External Port
adc2_tdata_dut_o | Outport | uint32 | External Port
adc2_tlast_dut_o | Outport | boolean | External Port
dac1_tx_tvalid | Outport | boolean | DAC1 AXI4 Stream Master | Valid
dac1_tx_tdata | Outport | uint32 | DAC1 AXI4 Stream Master | Data
Run This Task
Help
Apply



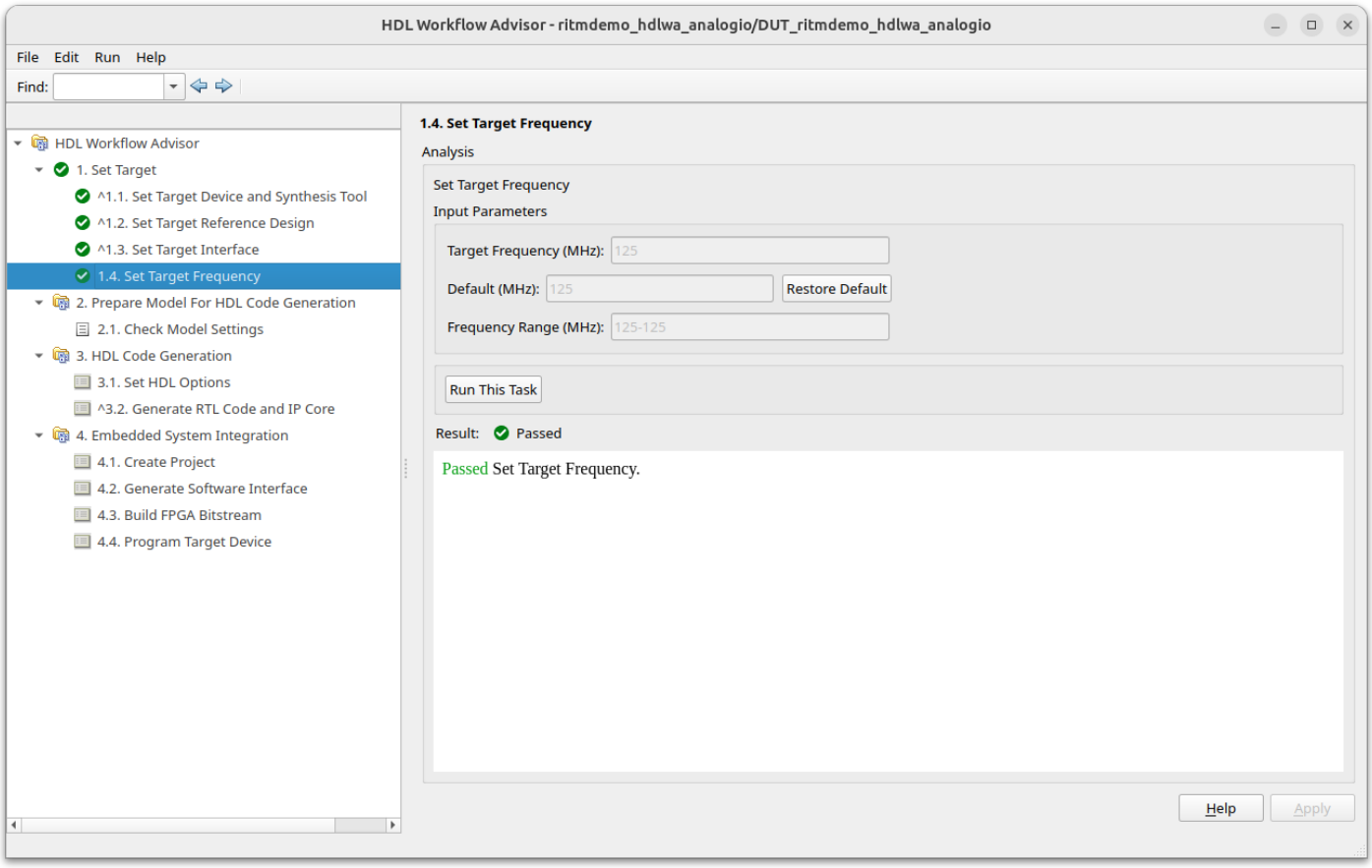


При работе с FMC модулем используются интерфейсы:
- AXI4-Lite - Интерфейс для обмена данными между процессором и ПЛИС через PCIe;
- ADC* AXI4 Stream Slave - Интерфейс для приёма данных от аналоговых входов;
- DAC* AXI4 Stream Master - Интерфейс для передачи данных в аналоговые выходы;
- DA-RT-FMC ADC1_INIT - Выходной порт для инициализации первых восьми аналоговых входов;
- DA-RT-FMC ADC2_INIT - Выходной порт для инициализации вторых восьми аналоговых входов;
- DA-RT-FMC ADC1_En_AXISTREAM - Выходной порт включения приёма данных по AXI4 Stream для первых восьми аналоговых входов;
- DA-RT-FMC ADC2_En_AXISTREAM - Выходной порт включения приёма данных по AXI4 Stream для вторых восьми аналоговых входов;
- FMC-Power - Питание модуля FMC.

- DA-RT-FMC DAC1 NUMS_CH - Выходной порт для инициализации количества каналов первых восьми аналоговых выходов;
- DA-RT-FMC DAC2 NUMS_CH - Выходной порт для инициализации количества каналов втторых восьми аналоговых выходов;

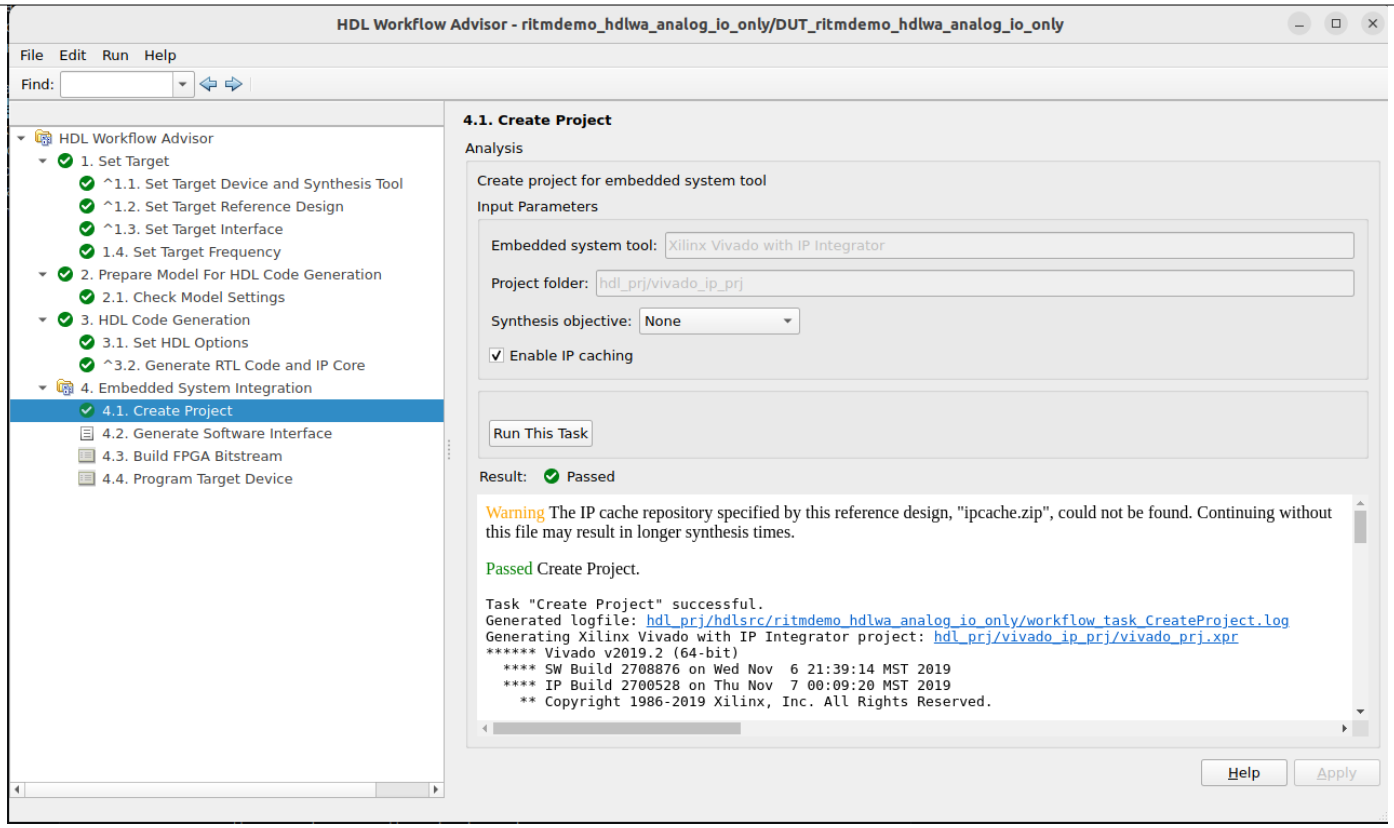
При назначении интерфейса происходит автоматическое назначение адреса порта.

- На шаге 1.4 настройте тактовую частоту DUT:

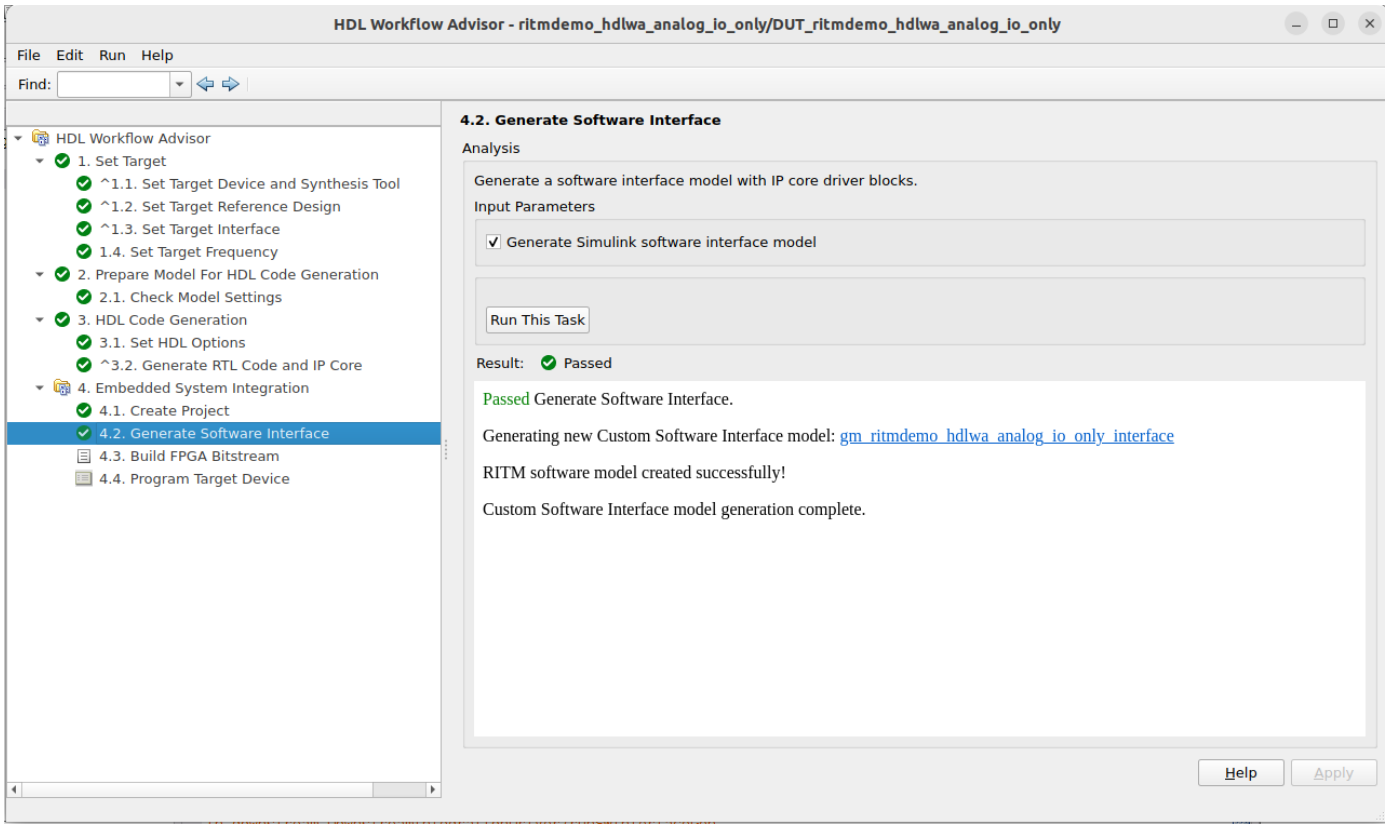


Генерация HDL кода и интеграция с проектом Xilinx Vivado

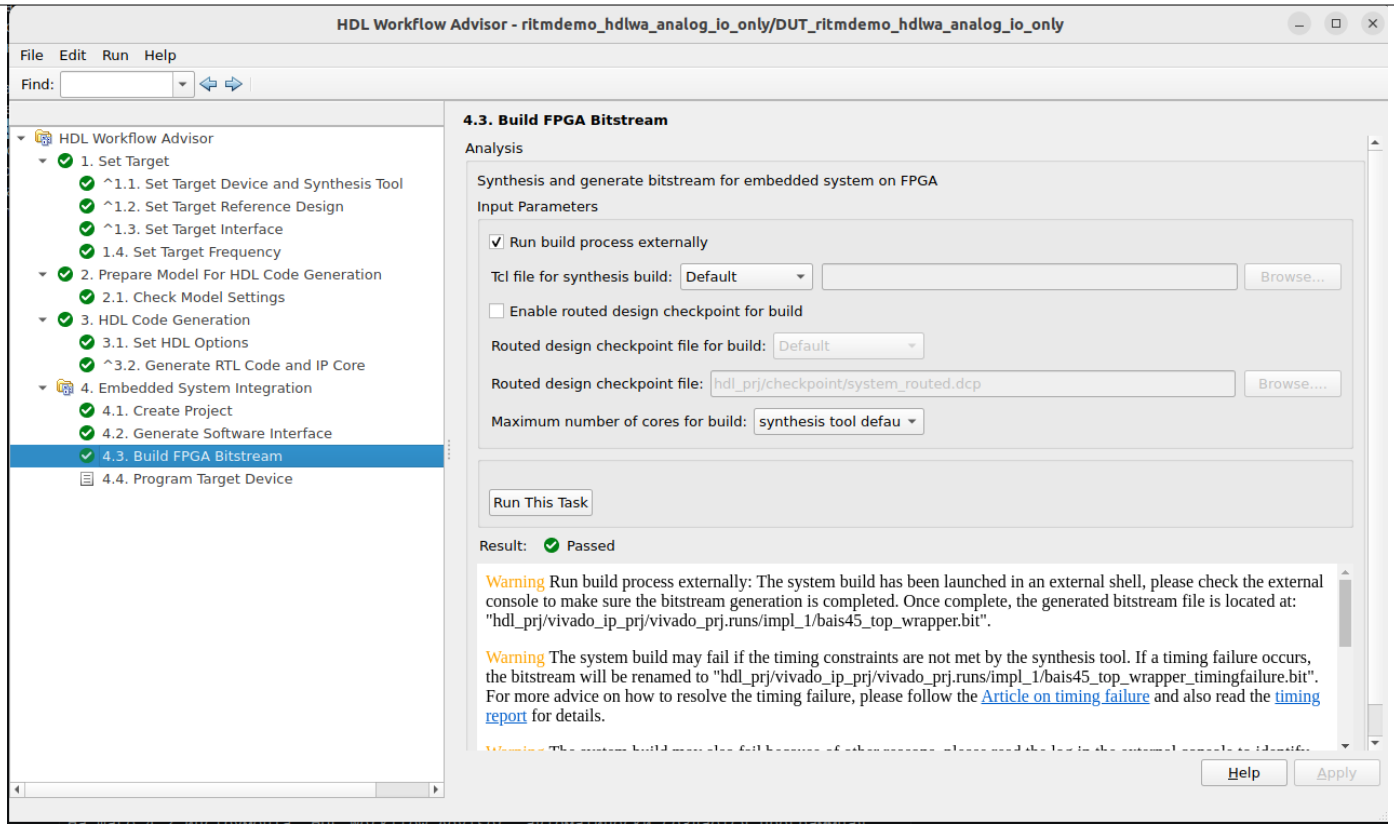
- На шагах 2 и 3 проверяются настройки модели и предоставляется возможность настроить параметры генерации кода.
- На шаге 4.1 создается проект в Xilinx Vivado. Щелкните правой кнопкой мыши по шагу 4.1 и выберите Run to Selected Task:



- На шаге 4.2 создается программная модель. Больше информации о программной модели в разделе Обзор программной модели:

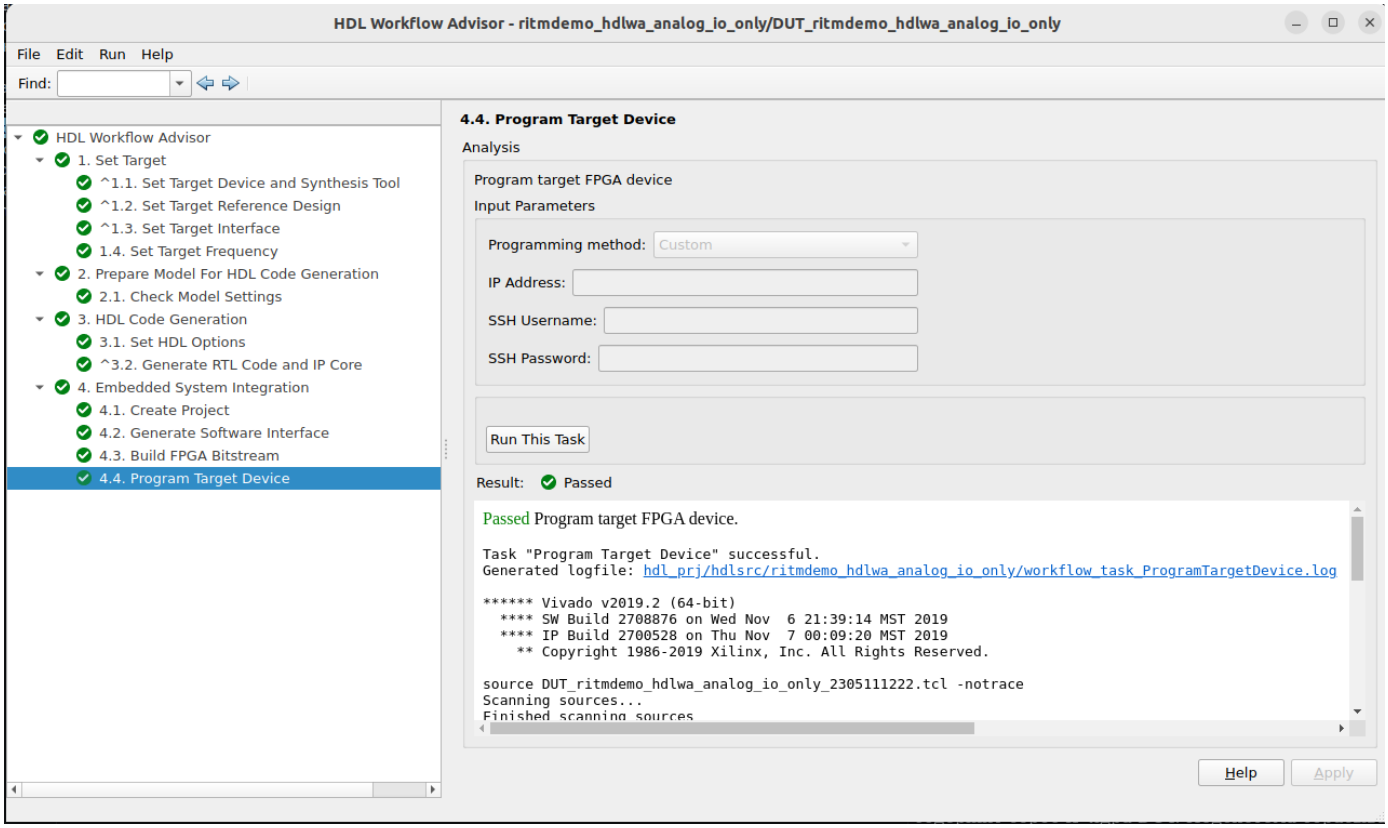


- На шаге 4.3 происходит генерация битового потока:



При включенной опции Run build process externally запуск сборки происходит параллельно с работой MATLAB. При выключенной опции Вы не сможете использовать MATLAB, пока сборка не будет завершена.

- На шаге 4.4 создается прошивка для программирования ПЛИС:

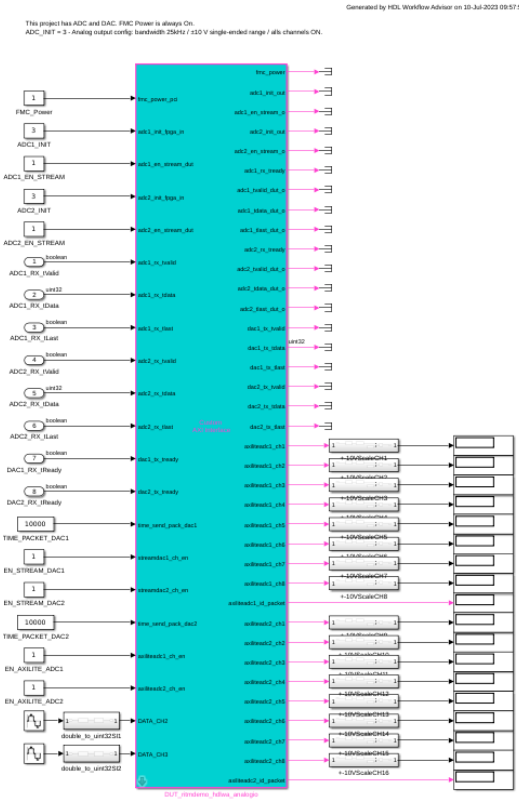


На данном шаге генерируются файлы с расширением .mat и .mcs (прошивка), но не загружаются на ПЛИС. Они необходимы для запуска модели на РИТМ.

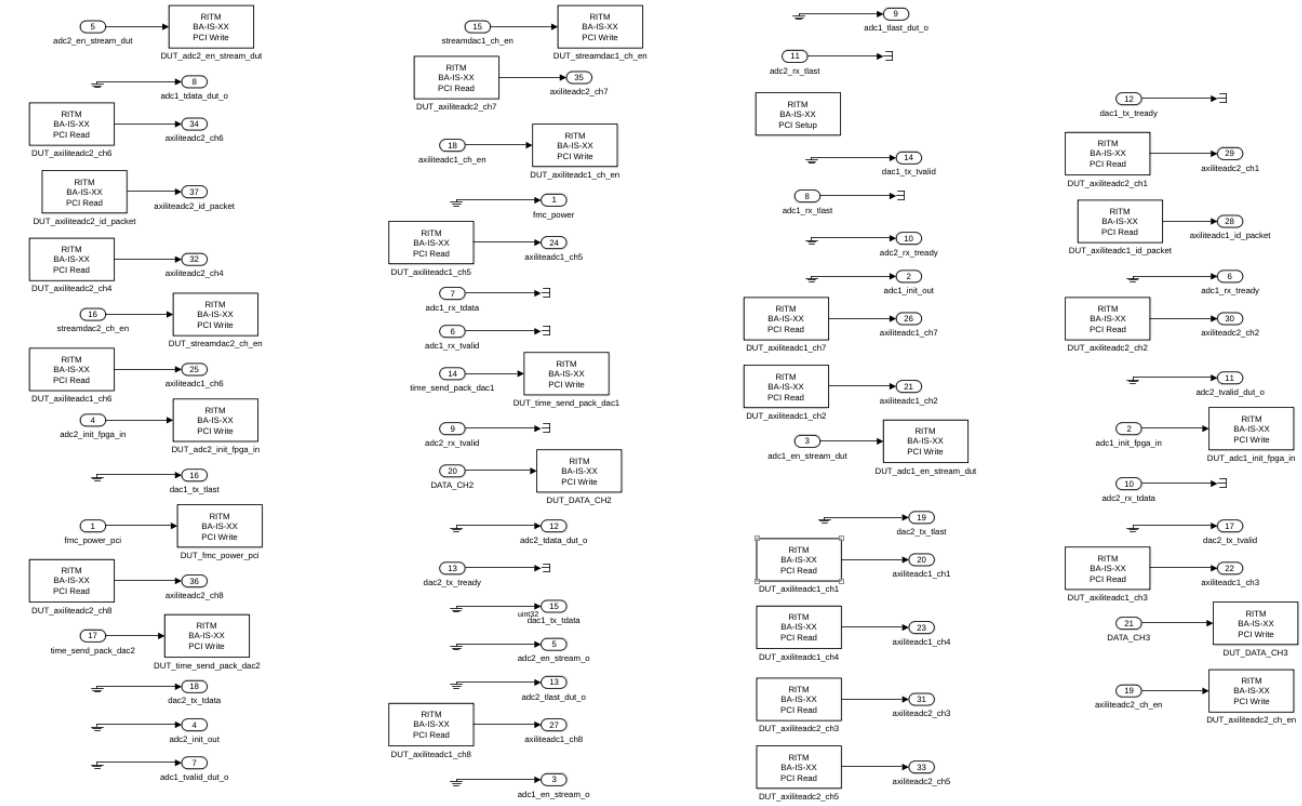
Этим завершается обзор процесса генерации прошивки ПЛИС из подсистемы DUT.

Обзор программной модели

На шаге 4.2 инструмента HDL Workflow Advisor автоматически создается программная модель. Эта модель представляет собой копию оригинальной модели, но с заменой всей подсистемы DUT на специальный блок:



Таким образом, программная модель содержит в себе блоки и алгоритмы, которые будут запущены на CPU. Кроме того, она содержит сгенерированный блок, который представляет собой интерфейсы между CPU и DUT:



Программную модель нужно запустить на РИТМе, при этом ПЛИС будет прошита автоматически. Последующие запуски одной и той же программной модели не будут приводить к изменению прошивки ПЛИС.

Ознакомьтесь с программной моделью, сгенерированной в этом примере.
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_analogio_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_analogio_BA_IS_55_interface.slx')
```

Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС

Содержание

- Модели для цифровых входов-выходов
  - Демо модель default
  - Демо модель pwm_cap
  - Демо модель gpiod64_l1
  - Демо модель gpiod64_l1l2
- Модели для аналоговых входов-выходов
  - Демо модель analogio_gpio
  - Демо модель analogio_gpiod64
- Модель передачи типов данных для входов-выходов подсистемы DUT
- Модели для прямого доступа к памяти DMA
  - Демо модель dma
- Модели смешанного типа
  - Демо модель dma/аналог/цифра
- Многочастотные модели
  - Демо модель multirate_pid

В этом разделе представлены базовые модели для подключения цифровых/аналоговых входов и выходов к алгоритму, работающему на ПЛИС.

Модели для цифровых входов-выходов

Демо модель default

Следующие модели показывают базовые примеры без использования установленных FMC уровней.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_default.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_default_BA_IS_55.slx')
```

Пример можно использовать как основу больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_default_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_default_BA_IS_55_interface.slx')
```

Демо модель pwm_cap

Модель pwm_cap показывает пример использования GP-RT-FMC-D Level 1, IP ядер pwm и pwm_cap, входящих в пакет поддержки “РИТМ”.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_pwm_cap.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_pwm_cap_BA_IS_55.slx')
```

Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_pwm_cap_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_pwm_cap_BA_IS_55_interface.slx')
```

Демо модель gpiod64_l1

Модель gpiod64_l1 показывает пример использования GP-RT-FMC-D64 Level 1.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_gpiod64_l1.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_gpiod64_l1_BA_IS_55.slx')
```

Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_gpiod64_l1_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_gpiod64_l1_BA_IS_55_interface.slx')
```

Демо модель gpiod64_l1l2

Модель gpiod64_l1l2 показывает пример использования GP-RT-FMC-D64 Level 1 и GP-RT-

FMC-D64 Level 2.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_gpiod64_l1l2.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_gpiod64_l1l2_BA_IS_55.slx')
```

Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_gpiod64_l1l2_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_gpiod64_l1l2_BA_IS_55_interface.slx')
```

Модели для аналоговых входов-выходов

Демо модель analogio_gpio

Модель analogio_gpio показывает базовый пример совместного использования DA-RT-FMC Level 1 и GP-RT-FMC-D Level 2.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_analogio_gpio.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_analogio_gpio_BA_IS_55.slx')
```

Пример можно использовать как основу для больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_analogio_gpio_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_analogio_gpio_BA_IS_55_interface.slx')
```

Демо модель analogio_gpiod64

Модель analogio_gpiod64 показывает базовый пример совместного использования DA-RT-FMC Level 1 и GP-RT-FMC-D64 Level 2.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_analogio_gpiod64.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_analogio_gpiod64_BA_IS_55.slx')
```

Пример можно использовать как основу для больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_analogio_gpiod64_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_analogio_gpiod64_BA_IS_55_interface.slx')
```

Модель передачи типов данных для входов-выходов подсистемы DUT

Модель datatypes показывает пример передачи различных типов данных в и из подсистемы DUT. Для модуля BA-IS-45 и модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_datatypes.slx')
```

Модели для прямого доступа к памяти DMA

Демо модель dma

Модель dma показывает базовый пример использования DMA и DUT.
Для модуля BA-IS-45.

```
open_system('ritmdemo_hdlwa_dma.slx')
```

Для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_dma_BA_IS_55.slx')
```

Пример можно использовать как основу для больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программные модели:
Для модуля BA-IS-45.

```
open_system('gm_ritmdemo_hdlwa_dma_interface.slx')
```

Для модуля BA-IS-55.

```
open_system('gm_ritmdemo_hdlwa_dma_BA_IS_55_interface.slx')
```

Модели смешанного типа

Демо модель dma/аналог/цифра

Модель dma/аналог/цифра показывает базовый пример использования DA-RT-FMC Level 1, GP-RT-FMC-D64 Level 2 и DMA с DUT для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_dma_analogio_gpio_BA_IS_55.slx')
```

Пример можно использовать как основу для больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программная модель:

```
open_system('gm_ritmdemo_hdlwa_dma_analogio_gpio_BA_IS_55_interface.slx')
```

Многочастотные модели

Демо модель multirate_pid

Модель показывает базовый пример многочастотной модели с использованием DA-RT-FMC Level 1 в DUT для модуля BA-IS-55.

```
open_system('ritmdemo_hdlwa_multirate_pid_BA_IS_55.slx')
```

Пример можно использовать как основу для больших проектов с автоматической генерацией прошивки ПЛИС из алгоритма в подсистеме DUT.
Программная модель:

```
open_system('gm_ritmdemo_hdlwa_multirate_pid_BA_IS_55_interface.slx')
```

Моделирование силовой электроники в реальном времени на ПЛИС

Содержание

- Требования к аппаратному обеспечению
- Требования к программному обеспечению
- Общие положения
- Создание и подготовка Simscape-модели
- Генерация модели в пространстве состояний
  - **Этапы перевода модели Simscape в модель реализации HDL:**
  - **Работа модели реализации HDL**
  - **Валидация модели реализации HDL**
- Генерация HDL-кода

В этом разделе показывается, как сгенерировать HDL-код для модели с элементами силовой электроники, которая использует блоки Simscape, и запустить такую модель в реальном времени на КПМ РИТМ с модулем ПЛИС.

Требования к аппаратному обеспечению

- Установленная базовая плата RITMeX BA-IS-45 или RITMeX BA-IS-55.

Требования к программному обеспечению

- HDL Coder для генерации кода
- Fixed-Point Designer для работы с HDL Coder
- Xilinx Vivado для программирования ПЛИС:
  - Для программирования BA-IS-55 требуется версия Xilinx Vivado с поддержкой Ultrascale
  - Для программирования BA-IS-45 требуется версия Xilinx Vivado с поддержкой 7 Series
  - Работа верифицирована на версии Xilinx Vivado 2019.2

Общие положения

Процесс подготовки и запуска Simscape-модели на ПЛИС в реальном времени состоит из нескольких шагов:

- Создание и подготовка Simscape-модели
- Генерация модели в пространстве состояний пригодной для реализации HDL с помощью *Simscape HDL Workflow Advisor*.
- Генерация HDL-кода из модели в пространстве состояний для запуска на ПЛИС.

Схематично процесс запуска модели Simcape-модели на ПЛИС выглядит следующим образом:

Для лучшего понимания взаимодействия железа и софта обратитесь к разделу Общие сведения о запуске алгоритмов на ПЛИС.

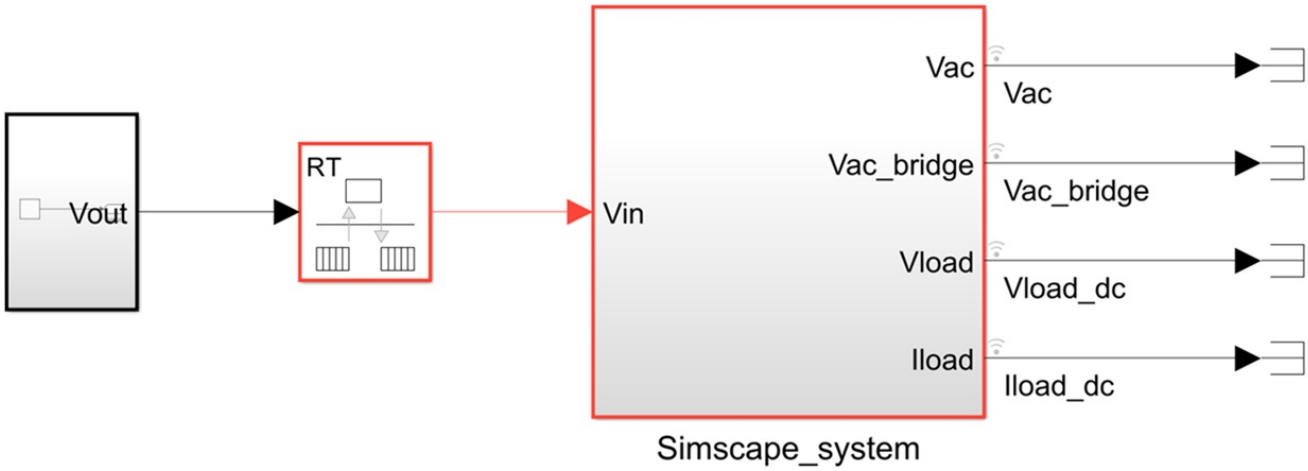
Создание и подготовка Simscape-модели

В качестве примера рассмотрим схему двухполупериодного мостового выпрямителя.

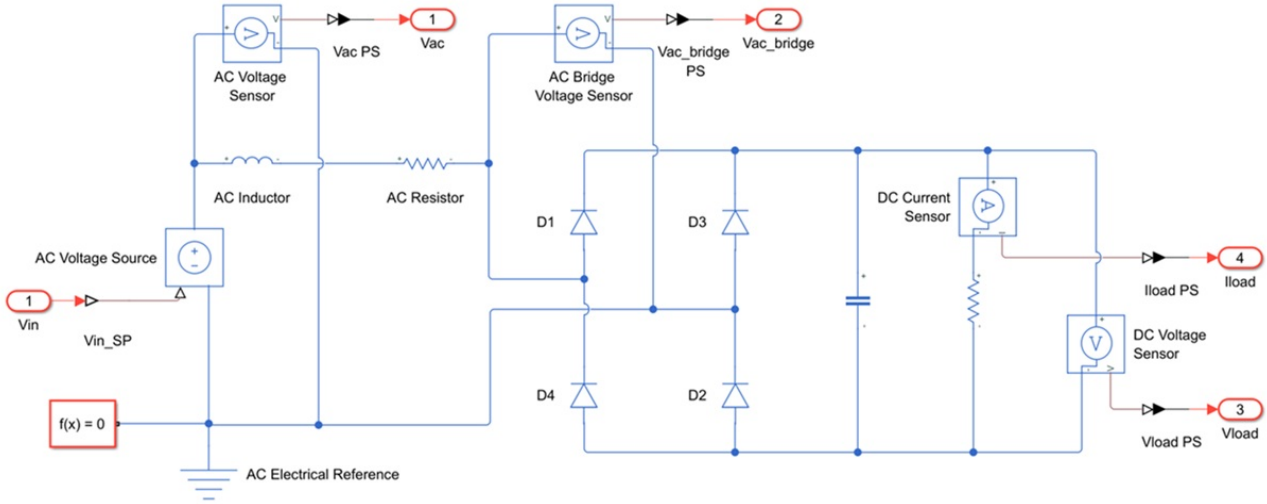
```
open_system('ritmdemo_bridge_rectifier.slx')
```

Данная модель в примере будет адаптирована для запуска на базовой плате RITMeX BA-IS-55. Копию демо-модели следует сохранить в текущую рабочую папку для дальнейшей работы, для этого в командной строке введите:

```
save_system('ritmdemo_bridge_rectifier','bridge_rectifier')
open_system('bridge_rectifier')
set_param('bridge_rectifier', 'SimulationCommand', 'Update')
```



На верхнем уровне модели расположен блок в виде подсистемы Simscape_system, который моделирует физический процесс мостового выпрямителя. Данная подсистема принимает входной синусоидальный сигнал, блок Rate Transition выполняет дискретизацию входного сигнала. Выходные данные (переменное напряжение на источнике Vac, переменное напряжение на входе выпрямителя Vac_bridge, выпрямленное напряжение на нагрузке Vload, ток на нагрузке Iload), полученные по результатам моделирования, можно посмотреть в окне *Data Inspector*. Чтобы увидеть схему мостового выпрямителя, перейдите в подсистему Simscape_system.

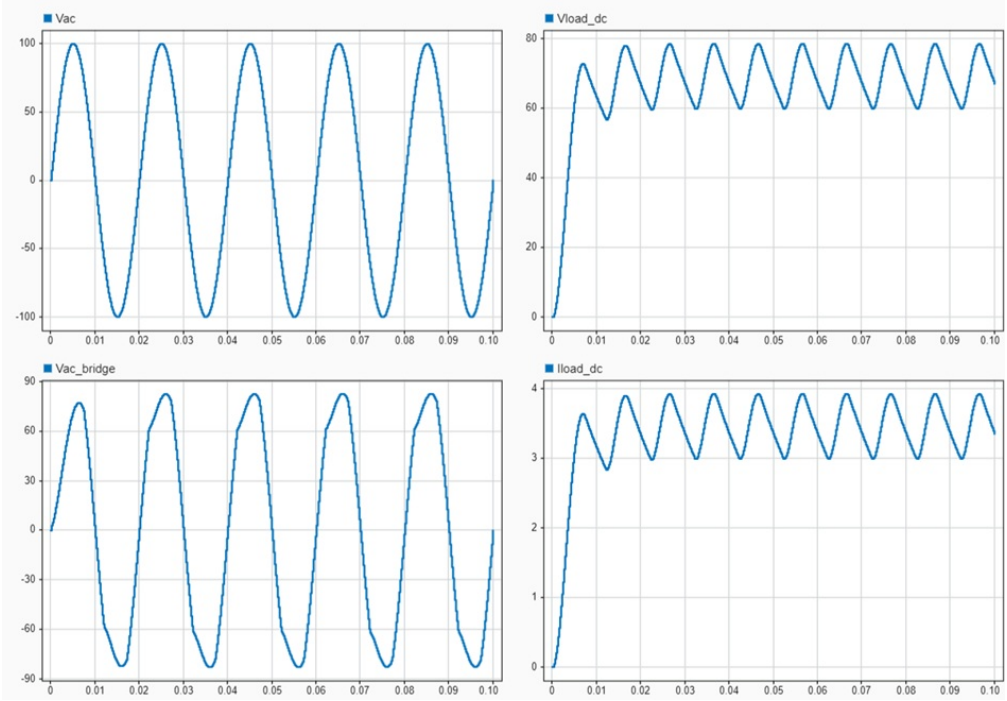


В данном примере мостовой выпрямитель состоит из набора резисторов, индуктивностей и конденсаторов, представляющих собой линейные блоки, а также диодов, представляющих собой переключаемый линейный блок. На интерфейсах портов ввода и вывода модель имеет блоки Simulink-PS Converter и PS-Simulink Converter. Для перехода к дальнейшим шагам, необходимо убедиться в правильности настройки модели. Перейдите к настройкам решателей и убедитесь, что они выставлены для совместимости с *Simscape HDL Workflow Advisor*:

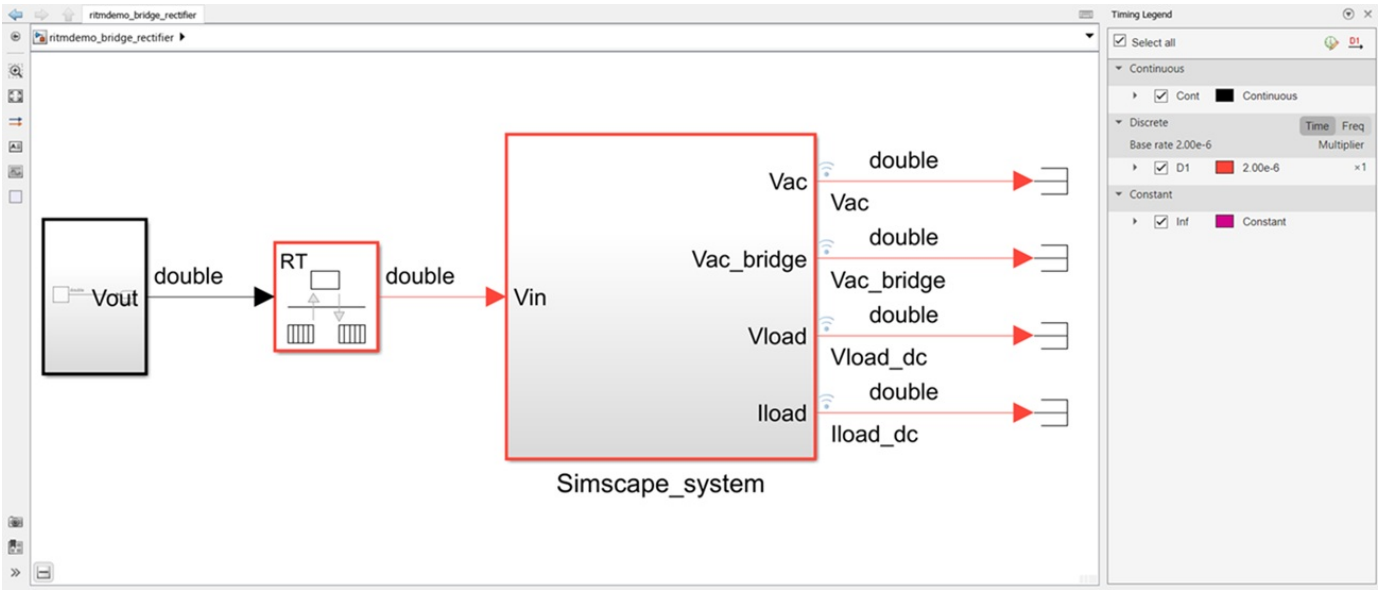
- Перейдите в настройки модели *Configuration Parameters => Solver* и переведите тип решателя в *Fixed-step*
- Откройте диалоговое окно *Block Settings* для блока Solver Configuration, затем включите опцию *Use local Solver*, а *Solver type* переведите в *Backward Euler*;
- В выпадающей вкладке *Use fixed-cost runtime consistency iterations* блока Solver Configuration выставите поле *Nonlinear iterations* в значение 2. В зависимости от этого значения, будет выставлен шаг модели состояний на этапе Генерация модели в пространстве состояний.

Для более подробной информации смотрите раздел документации Get Started with Simscape Hardware-in-the-Loop Workflow.
Чтобы увидеть работу схемы, запустите симуляцию модели и затем откройте окно *Data Inspector*.

```
sim('ritmdemo_bridge_rectifier')
```



Обратите внимание, какой шаг расчета использует исходная модель. Чтобы вывести легенду шага расчета пройдите в *Debug => Information Overlays => Timing Legend*. Управляющий сигнал подсвечен черным цветом - Continuous, блоки моделирующие электрическую схему красным цветом - 2 мкс. Также обратите внимание, что на данном этапе тип данных у входных и выходных сигналов пока что Double, в дальнейшем будет проведена замена на типы данных поддерживаемые для работы на ПЛИС.



Генерация модели в пространстве состояний

Чтобы перейти от модели Simcsape к модели в пространстве состояний, которая состоит из базовых направленных блоков Simulink пригодных для генерации HDL-кода, используйте *Simscape HDL Workflow Advisor*. С помощью *Simscape HDL Workflow Advisor* будет сгенерирована так называемая **модель реализации HDL**, из которой далее будет сгенерирован HDL-код для ПЛИС. Чтобы открыть *Simscape HDL Workflow Advisor*, выполните команду.

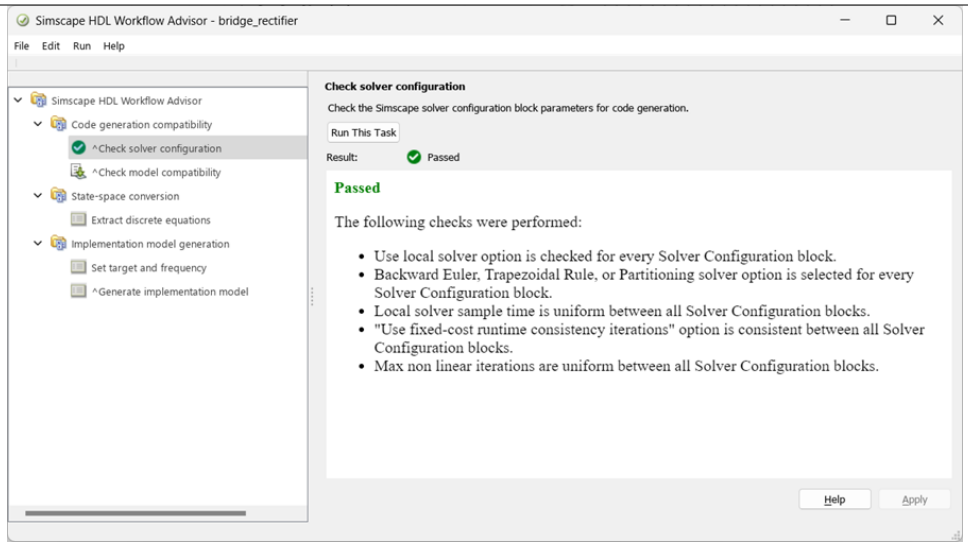
```
sschdladvisor('ritmdemo_bridge_rectifier')
```

Эта команда обновляет кэш *Simscape HDL Workflow Advisor* и открывает его. Чтобы узнать больше об *Simscape HDL Workflow Advisor* и различных задачах, щелкните правой кнопкой мыши на папку или задачу и выберите *“What’s This?”*. Для более подробной информации также смотрите Simscape HDL Workflow Advisor Tasks.

Этапы перевода модели Simscape в модель реализации HDL:

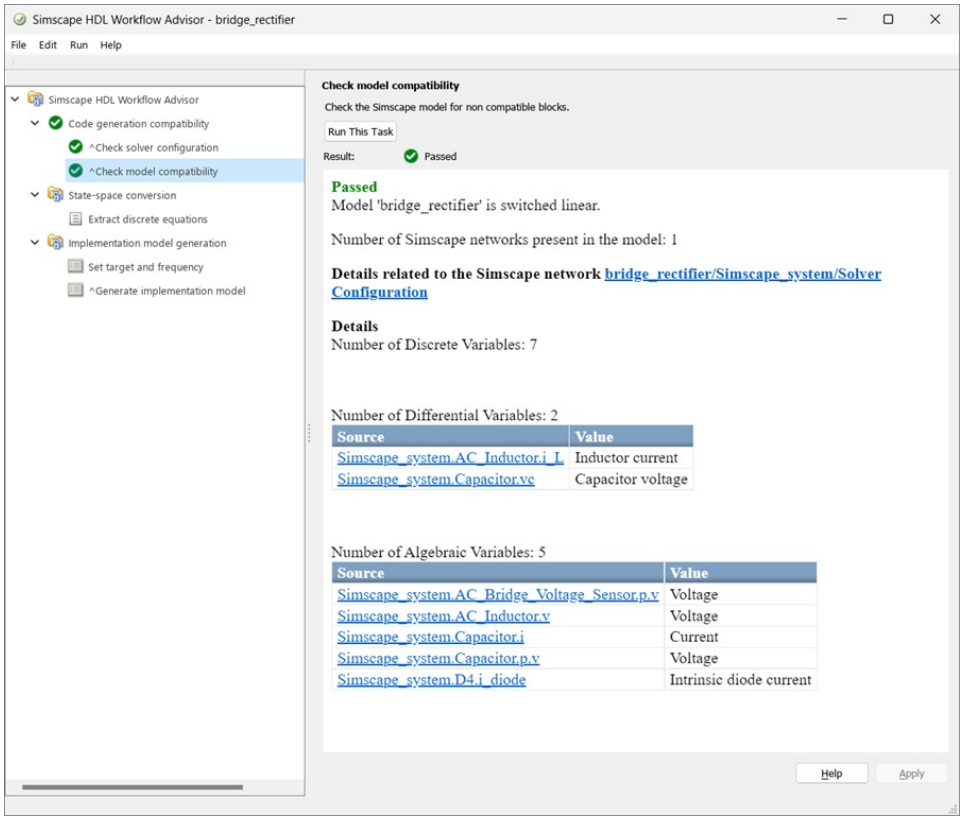
1. Запустите задачу проверки настроек решателя *“Check solver configuration”*

Чтобы запустить рабочий процесс HDL, выберите шаг *“Check solver configuration”* и нажмите кнопку *Run This Task*. Задача *“Check solver configuration”* проверяет корректность настроек решателей, а также согласованы ли между собой блоки с *Solver Configuration* внутри каждой сети в вашей модели Simscape.



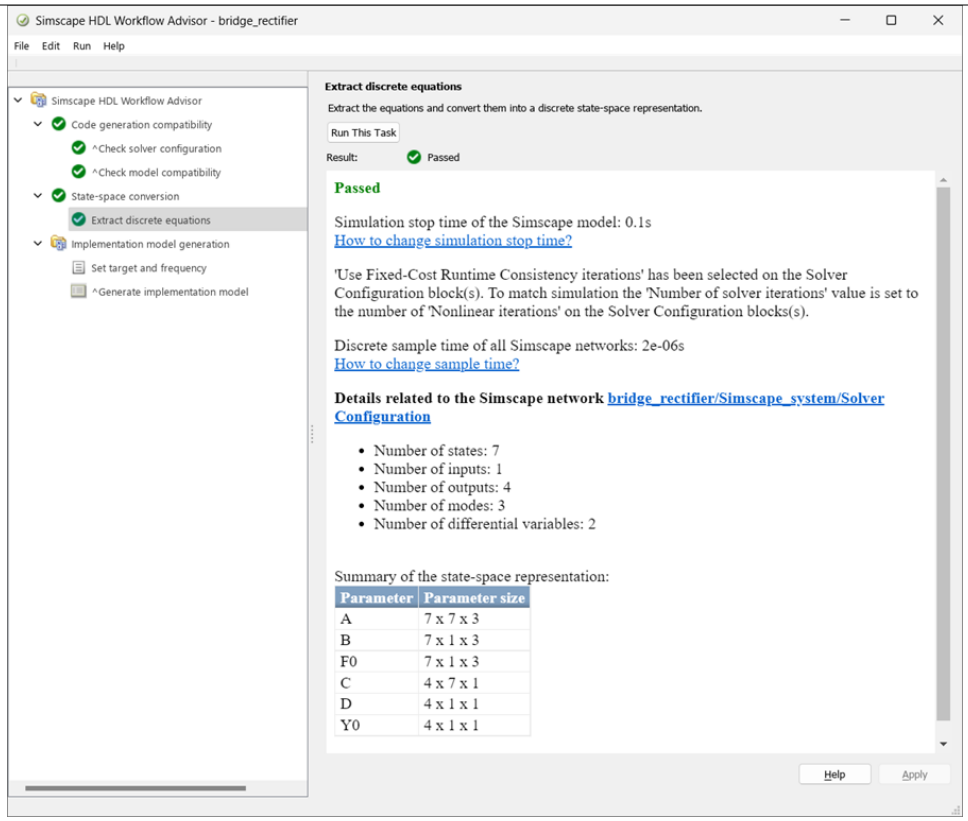
2. Запустите задачу проверки совместимости модели *“Check model compatibility”*

Далее, запустите задачу *“Check model compatibility”*. Она проверяет, используются ли в модели Simscape HDL-совместимые блоки.



3. Запустите задачу извлечения дискретных уравнений *“Extract discrete equations”*

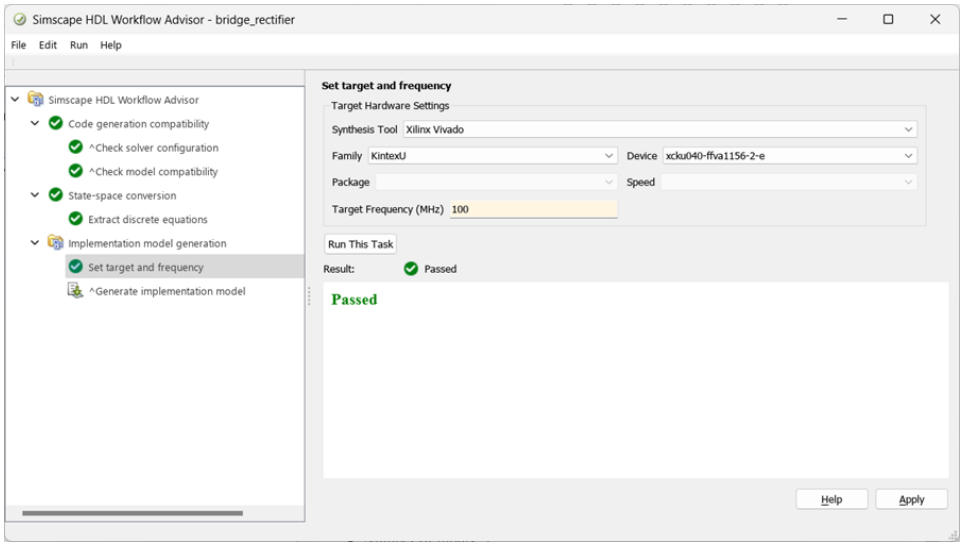
Обратите внимание, что данная задача может быть разделена на две составные задачи в версиях MATLAB старше 2023b.
Запустите задачу *“Extract discrete equations”*. Она извлекает дифференциально-алгебраические уравнения из модели Simscape и преобразовывает их в эквивалентное дискретное представление в пространстве состояний. Если эта задача проходит успешно, она отображает количество состояний, входов, выходов, режимов и дифференциальных переменных, а также отображает дискретный шаг расчета, количество параметров и размер параметра для каждой сети Simscape, присутствующей в модели.
Количество режимов ограничено количеством переключателей, присутствующих в вашей модели Simscape. Максимально возможное количество режимов — 2^n, где n — количество переключателей. Все режимы, которые генерирует *Simscape HDL Workflow Advisor*, выполняются согласно входным параметрам с использованием логики переключения. *Simscape HDL Workflow Advisor* выбирает допустимое количество режимов в зависимости от конструкции вашей модели Simscape.



4. Запустите задачу выбора целевого устройства и частоты *“Set target and frequency”*

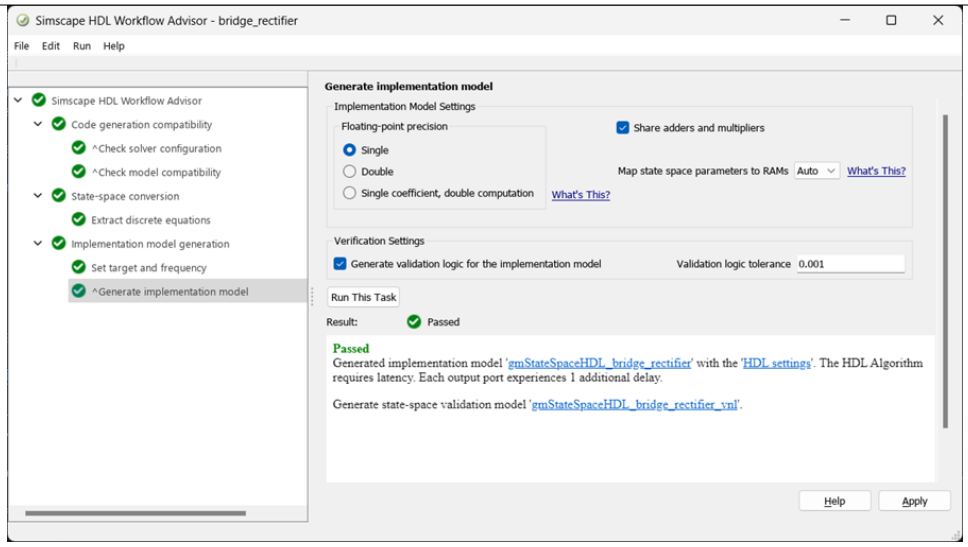
В этой задаче пользователем предоставляется информация, необходимая для настроек целевого оборудования - ПЛИС внутри КПМ РИТМ, прежде чем создавать модель реализации HDL. Это помогает включить оптимизацию HDL, необходимую для развертывания на заданном оборудовании. На правой панели выберите значение поля *Synthesis Tool* как *Xilinx Vivado*, а затем зададите поля *Family, Device, Package* и *Speed* для целевого оборудования и установите *Target Frequency* (MHz) в текстовом поле.
Следует применить следующие настройки для доступных в КПМ РИТМ модулей:

- Для модуля ПЛИС ВА-IS-55 - Family: *KintexU*, Device: *xcku040-ffva1156-2-e*, Target frequency: *100*;
- Для модуля ПЛИС ВА-IS-45 - Family: *Artix7*, Device: *xc7a200t*, Package: *ffg1156*, Speed: *-2*, Target frequency: *100*.

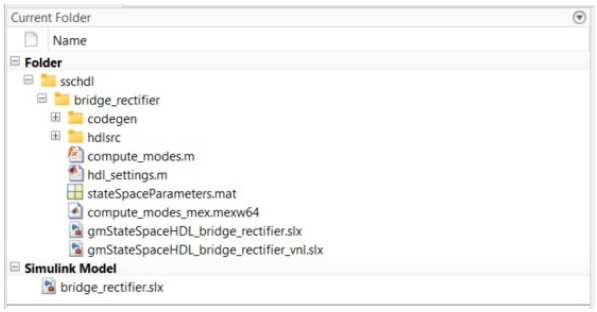


5. Запустите задачу генерации модели реализации *“Generate implementation model”*

На данном шаге происходит формирование и сравнение функциональности модели реализации HDL с исходной моделью Simscape. Для этого установите флажок *Generate validation logic for the implementation model*. Погрешность установите равной 0,001, а также установите опцию *floating-point precision* в значение Single.
В этой задаче *Simscape HDL Workflow Advisor* генерирует модель реализации HDL и модель проверки пространства состояний. Модель реализации имеет то же имя, что и исходная модель Simscape, и использует префикс “gmStateSpaceHDL_”. Модель проверки модели пространства состояний имеет то же имя, что и модель реализации, но использует постфикс “_vnl”.

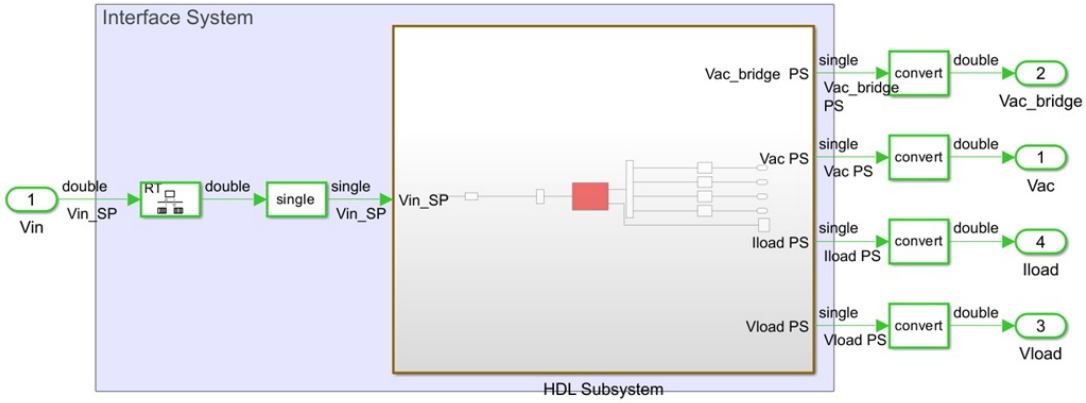


После завершения всех этапов, в рабочей директории появится папка со сгенерированной моделью реализации *gmStateSpaceHDL_bridge_rectifier.slx* и файлом параметров модели пространства состояний *stateSpaceParameter.mat*.

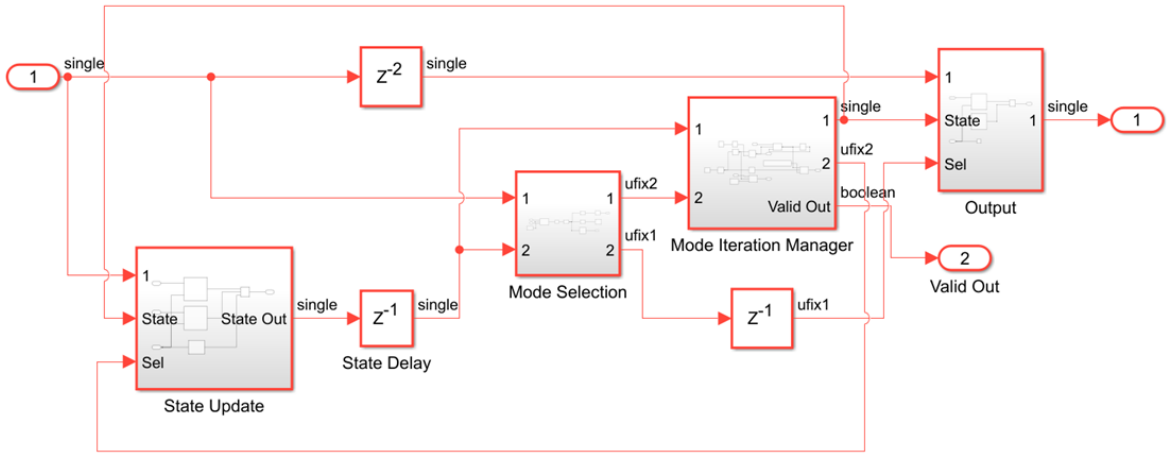


Работа модели реализации HDL

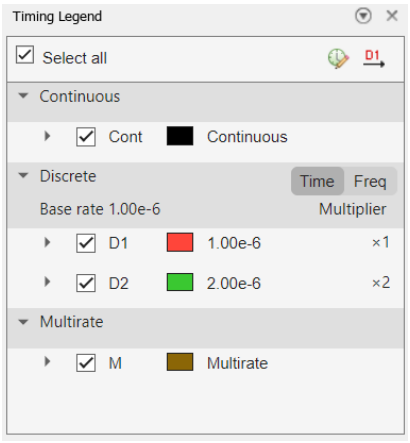
В задаче *Generate implementation model* щелкните ссылку, чтобы открыть модель реализации. Модель содержит подсистему Simscape_system, которая содержит блок HDL Subsystem. Подсистема HDL Subsystem содержит модель пространства состояний физической электрической сети с выпрямителем, которая сгенерировалась из модели Simscape.



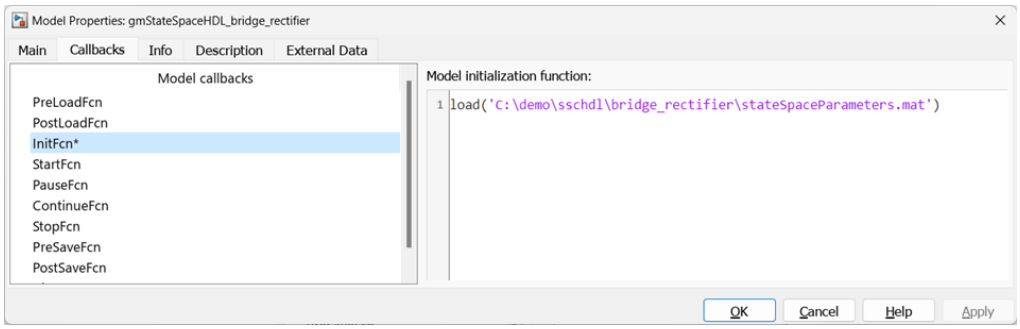
Порты этой подсистемы используют те же имена, что и блоки Simulink-PS Converter и PS-Simulink Converter в исходной модели Simscape. Если зайти внутрь этой подсистемы, можно увидеть систему из базовых блоков, которые моделируют уравнения в пространстве состояний.



В данной подсистеме стоит обратить внимание на типы данных и легенду времени. Типы данных были приведены к те типам, что поддерживаются ПЛИС на основании настроек *Simscape HDL Workflow Advisor*. При этом после преобразования в модель пространства состояний шаг расчета уменьшился в 2 раза (Значение *Nonlinear iterations* (равное 2) блока Solver Configuration учитывается на этапе 5) и равен 1.e-6 с. *Nonlinear iterations* - это сколько раз модель в пространстве состояний выполняется для каждого режима. *Simscape HDL Workflow Advisor* генерирует количество итераций, которые требуются для выполнения модели в пространстве состояний, автоматически, а также дает советы по выбору в случае некорректных значений.



Обратите внимание, что в сгенерированную модель была добавлена *InitFnc*. Чтобы ее посмотреть перейдите по: *Model Properties => Callbacks => InitFnc*. Данная функция подгружает в рабочую область *stateSpaceParameter.mat* при инициализации.



Запустите модель и оцените результаты.

Валидация модели реализации HDL

Чтобы сравнить функциональность модели реализации HDL с исходной моделью Simscape, откройте и смоделируйте модель проверки пространства состояний.

```
open_system('sschdl/ritmdemo_bridge_rectifier/gmStateSpaceHDL_ritmdemo_bridge_rectifier_vnl')
sim('gmStateSpaceHDL_ritmdemo_bridge_rectifier_vnl')
```

Если по окончании симуляции не было получено никаких предупреждений, это указывает на совпадение выходных данных с требуемой точностью. Для более системной проверки смотрите раздел документации Validate HDL Implementation Model to Simscape Algorithm.

Генерация HDL-кода

В результате прохождения предыдущих этапов в рабочей директории находится сгенерированная модель пространства состояний исходной Simscape-модели. Благодаря автоматической генерации модели для валидации полученная модель проверена на

соответствие с заданной точностью оригинальной модели и не содержит физических компонентов Simscape. Таким образом можно приступить к созданию модели для генерации HDL-кода под ПЛИС BA-IS-45 или BA-IS-55 в КПМ РИТМ при помощи *HDL Workflow Advisor*. Перед началом этого этапа, необходимо ознакомиться с содержанием документации Пакета поддержки Комплекса полунатурного моделирования “РИТМ” в разделах Общие сведения о запуске алгоритмов на ПЛИС и Дополнительные демо модели для подключения входов-выходов к алгоритму на ПЛИС. В этих разделах следует ознакомиться с понятием DUT-подсистемы и демонстрациями моделей с референсным дизайном. Демо модель с референсным дизайном - модель Simulink, которая содержит подсистему для ПЛИС, с определенным предустановленным функционалом для прошивки ПЛИС, например, для работы с ЦАП/АЦП на FMC-модуле. Внутрь подсистемы можно поместить любую HDL-компилируемую модель с “физикой” или иным алгоритмом и подключить сигналы от нее к выходным/входным портам подсистемы, например, для их выдачи с ЦАП. В пакете поддержки представлены следующие референсные дизайны:

- Демо модель default.
  Для BA-IS-45.

```
open_system('ritmdemo_hdlwa_default.slx')
```

Для BA-IS-55.

```
open_system('ritmdemo_hdlwa_default_BA_IS_55.slx')
```

- Демо модель pwm_cap для цифровых входов-выходов.
  Для BA-IS-45.

```
open_system('ritmdemo_hdlwa_pwm_cap.slx')
```

Для BA-IS-55.

```
open_system('ritmdemo_hdlwa_pwm_cap_BA_IS_55.slx')
```

- Демо модель analogio_gpio для аналоговых входов-выходов.

Для BA-IS-45.

```
open_system('ritmdemo_hdlwa_analogio_gpio.slx')
```

Для BA-IS-55.

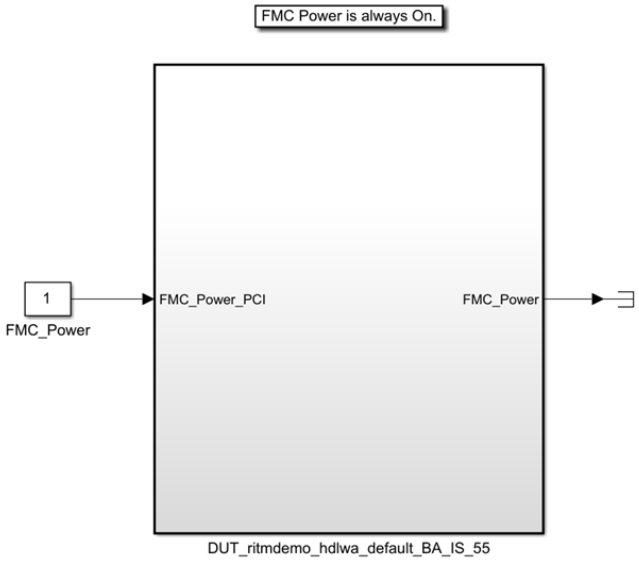
```
open_system('ritmdemo_hdlwa_analogio_gpio_BA_IS_55.slx')
```

Чтобы сгенерировать HDL-код в данном примере в качестве референса будет использоваться демо-модель *default* для BA-IS-55.

```
open_system('ritmdemo_hdlwa_default_BA_IS_55.slx')
```

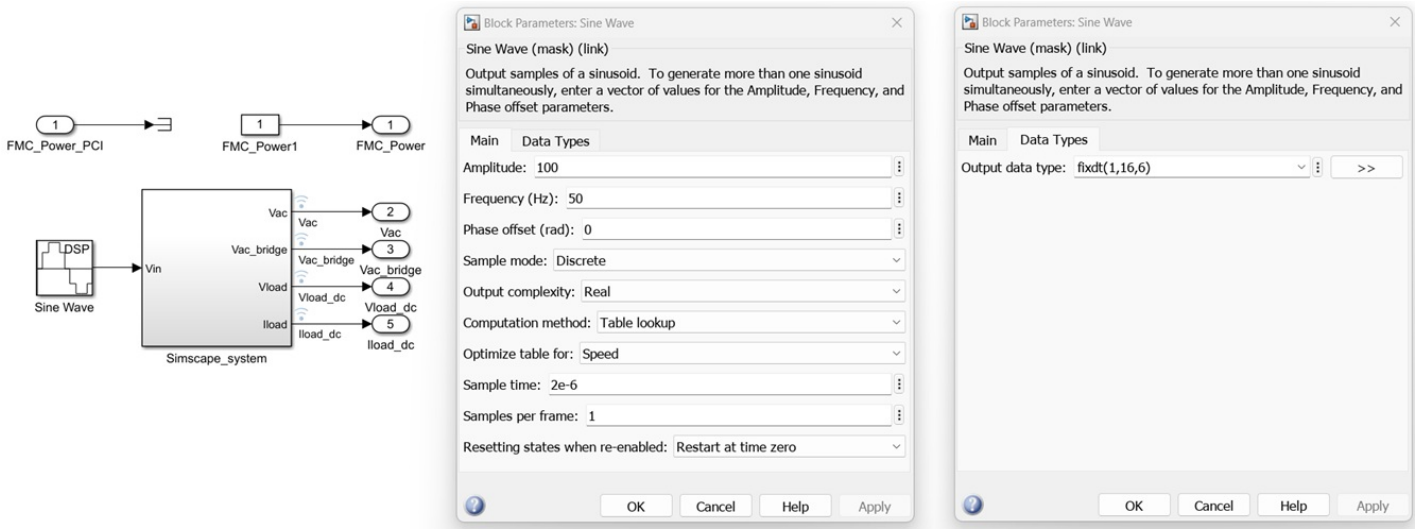
Сохраните эту модель локально с именем bridge_rectifier_RITM.slx в ту же рабочую папку, где сохранена bridge_rectifier.slx.

```
save_system('ritmdemo_hdlwa_default_BA_IS_55','bridge_rectifier_RITM')
open_system('bridge_rectifier_RITM')
```



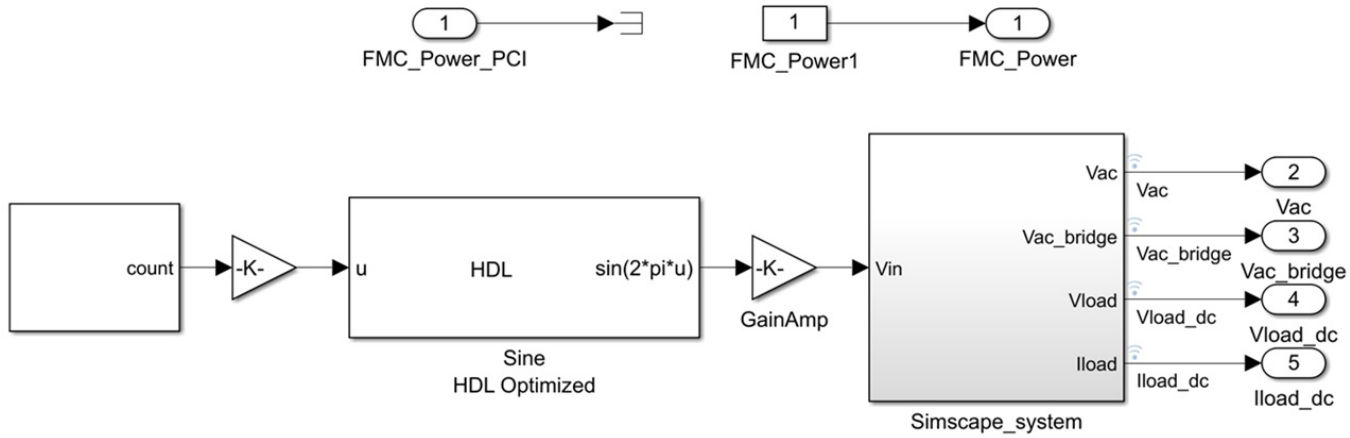
В открывшейся модели необходимо переименовать подсистему DUT_ritmdemo_hdlwa_default_BA_IS_55 на DUT_ritmdemo_bridge_rectifier_RITM. Все элементы модели, которые находятся в подсистеме DUT, будут работать на ПЛИС. Следует поместить внутрь подсистемы DUT_ritmdemo_bridge_rectifier_RITM модель реализации gmStateSpaceHDL_bridge_rectifier. При этом

необходимо заменить блок Sine Wave для управления источником напряжения исходной модели на HDL-компилируемый Sine Wave из библиотеки *DSP System Toolbox*, сохранив прежние параметры.
Во избежание конфликта имён других моделей переименуйте сгенерированный файл *stateSpaceParameter.mat* на *bridge_rectifier_stateSpaceParam.mat* и затем перенесите содержимое InitFnc модели реализации gmStateSpaceHDL_bridge_rectifier в модель с референсным дизайном bridge_rectifier_RITM.



Если HDL-компилируемый Sine Wave из библиотеки *DSP System Toolbox* недоступен, данный блок можно заменить конструкцией представленной ниже. Параметры такого набора блоков можно изучить в модели:

```
open_system('ritmdemo_bridge_rectifier_RITM.slx')
```



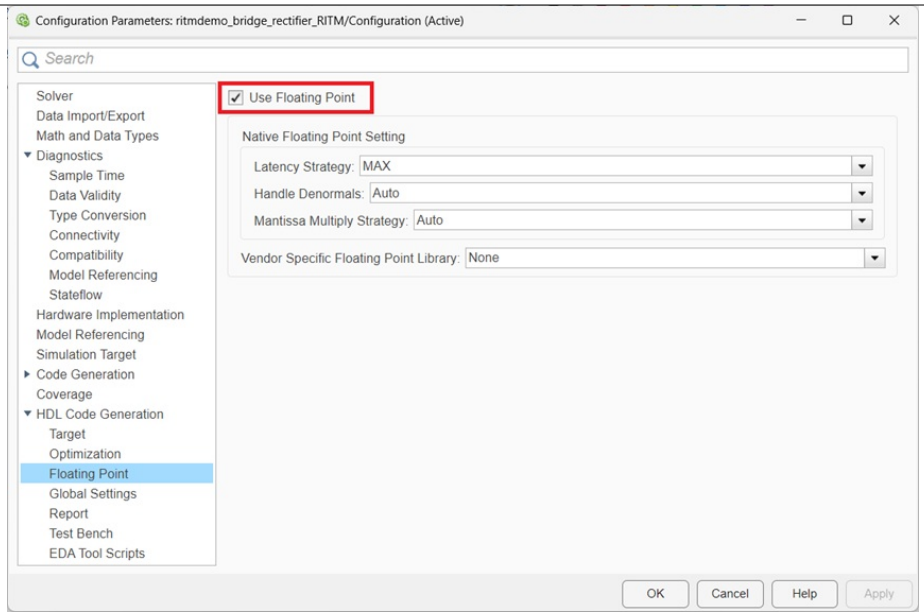
Запустите *HLD Workflow Advisor*, кликнув по подсистеме DUT_ritmdemo_bridge_rectifier_RITM правой кнопкой мыши *HDL Code => HLD Workflow Advisor*, или выполните команду:

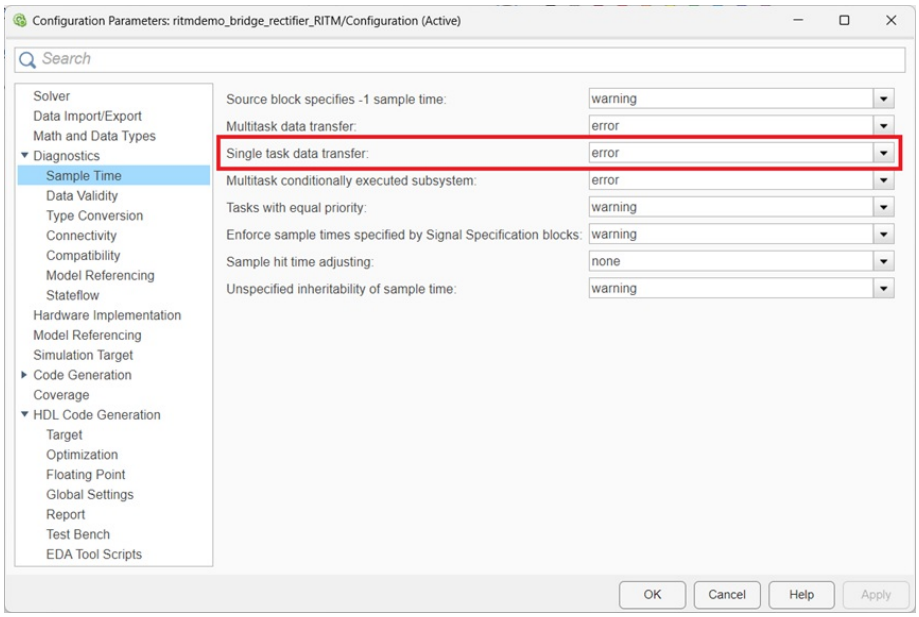
```
hdladvisor('bridge_rectifier_RITM/DUT_ritmdemo_bridge_rectifier_RITM')
```

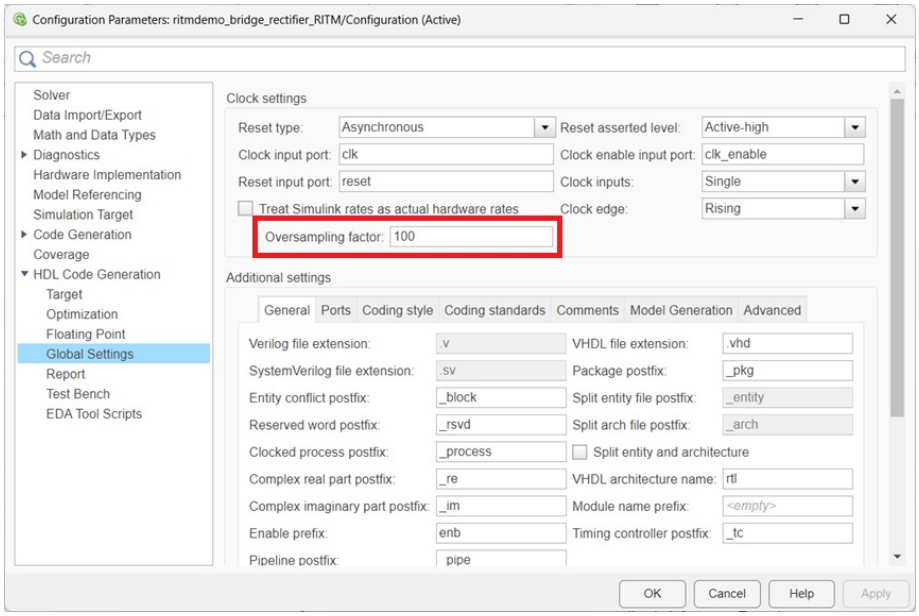
Пройдите все задачи, которые предлагает *HLD Workflow Advisor*. Процесс генерации HDL-кода не отличается от инструкции, описанной в документации в разделах Подключение цифровых входов-выходов к алгоритму на ПЛИС и Подключение аналоговых входов-выходов к алгоритму на ПЛИС. Однако, есть ряд нюансов:

- Модель использует тип данных Single с плавающей запятой, поэтому для задачи 3.2 *HLD Workflow Advisor* для генерации HDL-кода требуется включенный режим *Native Floating Point* - встроенная поддержку чисел с плавающей запятой в HDL Coder. Перейдите в настройки модели по следующему пути *Configuration Parameters => HDL Code Generation => Floating Point* и включите *Use Floating Point*;
- Также в задача 3.2 потребуется поставить опцию *Single task data transfer* в Error, найти это опцию можно по следующему адресу *Configuration Parameters => Diagnostics => Sample Time*;
- Необходимо задать параметр *Oversampling factor* для ПЛИС по адресу *Configuration Parameters => HDL Code Generation => Global Settings*. Этот параметр можно рассчитать по следующей формуле *Oversampling factor = step size of state-space model* target frequency / solver iterations*, для модели из примера: 2 мкс* 100 МГЦ / 2 = 100.

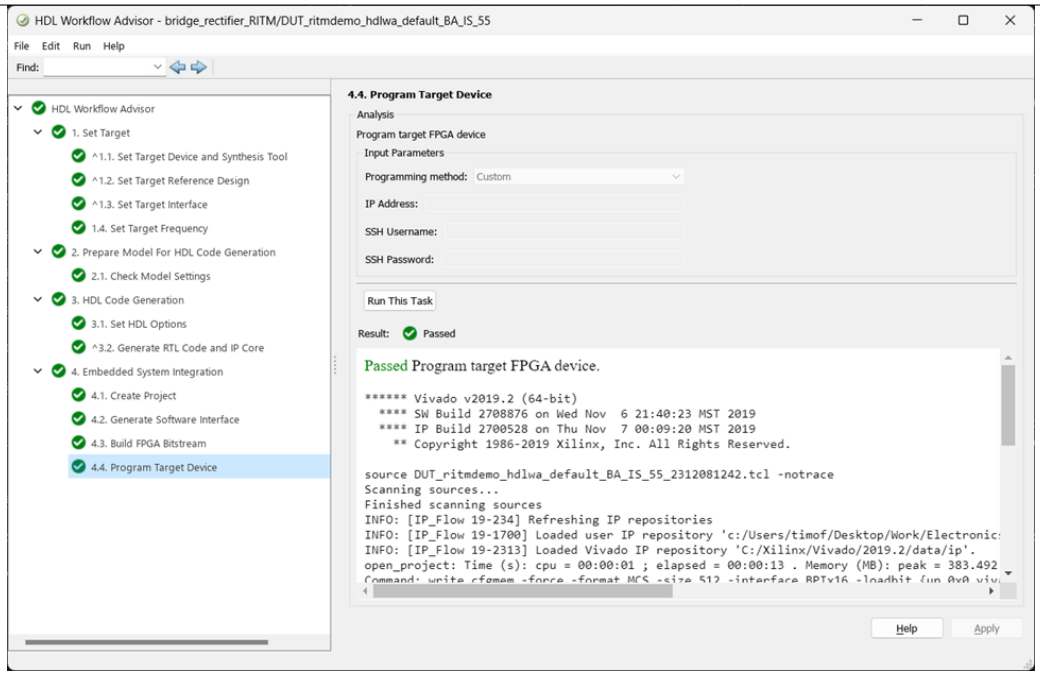
Для более подробной информации изучите разделы документации Clock-Rate Pipelining, Oversampling Parameters и Generate a Global Oversampling Clock.



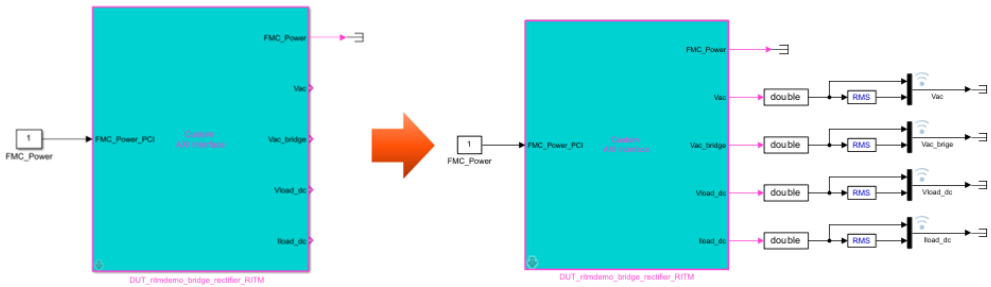




Если выполнить все требования *HLD Workflow Advisor*, то генерация HDL-кода для подсистемы DUT пройдет успешно.



На шаге 4.2 инструмента HDL Workflow Advisor автоматически создается программная модель.Эта модель представляет собой копию оригинальной модели, но с заменой всей подсистемы DUT на специальный блок, название такой модели будет иметь постфикс “_interface”. Теперь для средств визуализации к выходам DUT подсистеме подключим ряд блоков для вычисления действующего значения токов и напряжений, а затем залогируем эти сигналы в *Data Inspector*. При этот все блоки вне подсистемы DUT_ritmdemo_bridge_rectifier_RITM будут работать на CPU КПМ РИТМ.



Обратите внимание, что в рабочей папке появились специальные файлы, например MAT-файл для с префиксом *DUT_*. Эти файлы необходимы для прошивки ПЛИС и запуска в реальном времени.
Теперь данную модель можно запускать в реальном времени на КПМ РИТМ, при этом нужно провести настройку по запуску Interface-модели на машине в соответствии с Начало работы с пакетом поддержки “РИТМ”. При первом запуске ПЛИС будет прошита автоматически. Обратите внимание, что при прошивке ПЛИС, КПМ РИТМ выполнит перезагрузку. Последующие запуски одной и той же программной модели не будут приводить к изменению прошивки ПЛИС.

Пример сгенерированной модели для ПЛИС (*Примечание: для запуска в папке с моделью должны находиться вспомогательные файлы: DUT_ritmdemo_bridge_rectifier_RITM_2412171705.mat и bridge_rectifier_stateSpaceParam.mat*).

```
open_system('ritmdemo_bridge_rectifier_RITM_interface.slx')
```

Настраиваемые модули на базе ПЛИС

Модули ввода-вывода на базе ПЛИС и цифровых или аналоговых субмодулей. Позволяют реализовать гибкий функционал в зависимости от требований заказчика.

Модуль цифровых входов/выходов
Настраиваемый модуль цифровых входов/выходов на базе ПЛИС, поддерживающий до 128 цифровых каналов.

Модуль генерации ШИМ
Настраиваемый модуль генерации ШИМ на базе ПЛИС, поддерживающий от 1 до 128 каналов генерации ШИМ.

Модуль захвата ШИМ
Настраиваемый модуль захвата ШИМ на базе ПЛИС, поддерживающий от 1 до 128 каналов захвата ШИМ.

Модуль SPI
Настраиваемый модуль для работы с SPI на базе ПЛИС, поддерживающий SPI Master и SPI Slave.

Модуль I2C
Настраиваемый модуль для работы с I2C на базе ПЛИС, поддерживающий I2C Master и I2C Slave.

Модуль квадратурного декодера
Настраиваемый модуль для захвата сигналов квадратурного энкодера.

Модуль квадратурного энкодера
Настраиваемый модуль для симуляции сигналов квадратурного энкодера.

Модуль приемопередатчика UART
Настраиваемый модуль приемопередатчика UART.

Модуль захвата ШИМ через UART
Настраиваемый модуль для захвата ШИМ через UART.

Модуль аналоговых выходов
Настраиваемый модуль аналоговых выходов на базе ПЛИС, поддерживающий 16 независимых выходов.

Модуль аналоговых входов
Настраиваемый модуль аналоговых входов на базе ПЛИС, поддерживающий 16 независимых входов.

Модуль усреднения с аналоговых входов
Настраиваемый модуль усреднения RMS-SMA аналоговых входов, поддерживающий 16 независимых входов.

Модуль генератор частоты на аналоговые выходы
Настраиваемый модуль генератор частоты на аналоговые выходы на базе ПЛИС, поддерживающий 16 независимых выходов.

Модуль генератор синуса на аналоговые выходы

Настраиваемый модуль генератор синуса на аналоговые выходы на базе ПЛИС, поддерживающий 16 независимых выходов.

Модули FMC-RF1 аналоговых входов/выходов

Настраиваемые модули аналоговых входов/выходов на базе ПЛИС, поддерживающий 2 независимых выходов и 2 независимых входов.

Модуль аналоговых выходов

Содержание

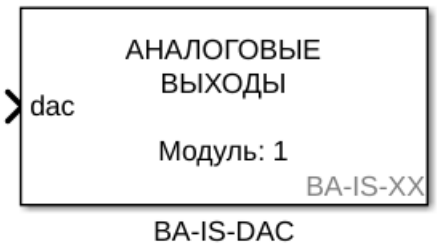
- Описание
- Библиотека блоков

Описание

Настраиваемый модуль аналоговых выходов на базе ПЛИС для машины реального времени РИТМ.

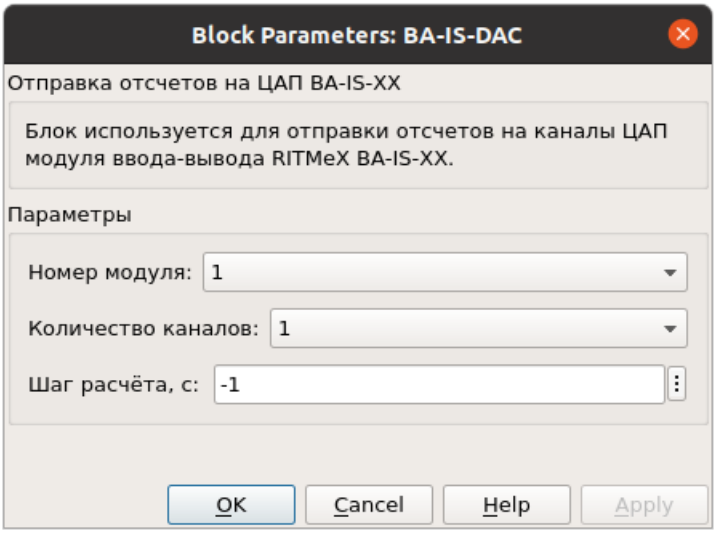
- 16 независимых выходов;
- Диапазон напряжений -10..+10 В;
- Разрядность ЦАП 16 бит;
- Одновременная установка значений по всем каналам;
- Рекомендуемый выходной ток на канал не более 1 мА (со стандартным клеммным модулем);
- Скорость установки выходного напряжения 1 В/мкс.

Библиотека блоков



BA-IS-DAC

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого

же типа.

- ***Количество каналов***: Задает количество каналов аналогового выхода.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

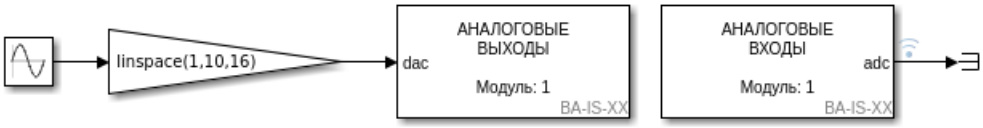
Входы блока

- **dac**: Данные для отправки на аналоговые выходы. Тип данных *double*, может быть массивом в зависимости от значения параметра *Количество каналов*.

Примеры и демонстрации BA-IS-DAC/BA-IS-ADC

Пример работы с каналами ЦАП и АЦП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_dac_adc.slx')
```



Назначение пинов BA-IS-DAC

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль аналоговых входов

Содержание

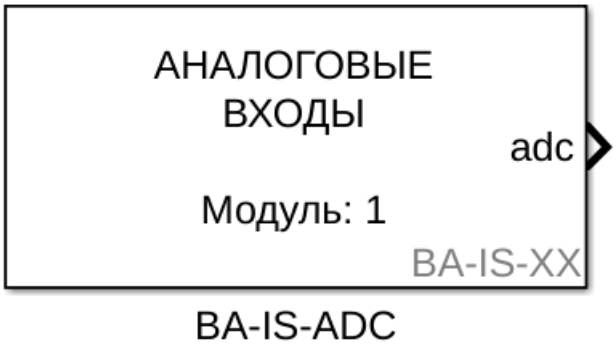
- Описание
- Библиотека блоков

Описание

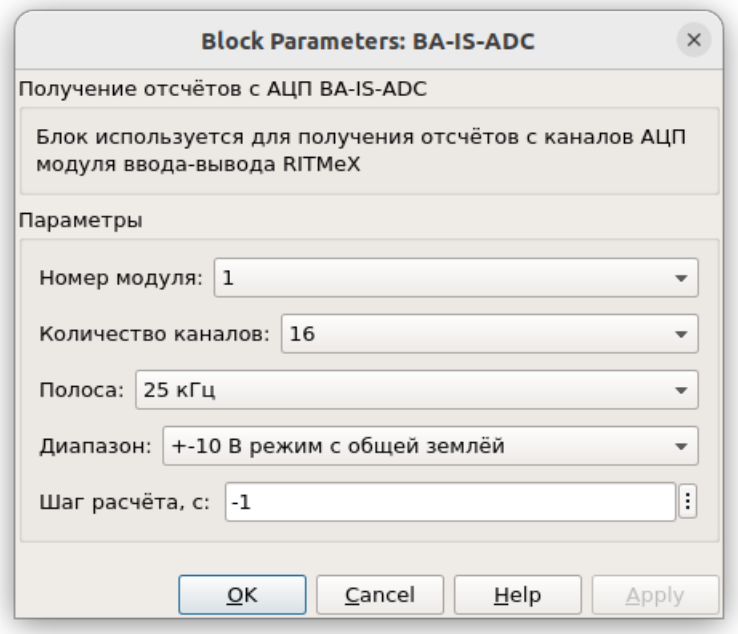
Настраиваемый модуль аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

- от 1 до 32 независимых входов;
- Входной диапазон напряжений: биполярный ±12.5 В, ±10 В, ±6.25 В, ±5 В, ±2.5 В; униполярный 0 В до 12.5 В, 0 В до 10 В, 0 В до 5 В и биполярный дифференциальный ±20 В, ±12.5 В, ±10 В, ±5 В.
- Разрядность АЦП 16 бит;
- Одновременное считывание значений по всем каналам;
- Входное сопротивление каждого канала 1 МОм;
- Частота дискретизации 1 МГц.
- Две полосы на каждый канал 25 кГц и 220 кГц.

Библиотека блоков



Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество каналов***: Задает количество каналов аналогового входа.
- ***Полоса***: Полоса пропускания 25 или 220 кГц.
- ***Диапазон***: Диапазон входного напряжения для каждого канала.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит получение данных с аналоговых входов.

Выходы блока

- **adc**: Данные, принятые с аналоговых входов. Тип данных *double*, может быть массивом в зависимости от значения параметра *Количество каналов*.

Назначение пинов BA-IS-ADC

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль генератор частоты на аналоговые выходы

Содержание

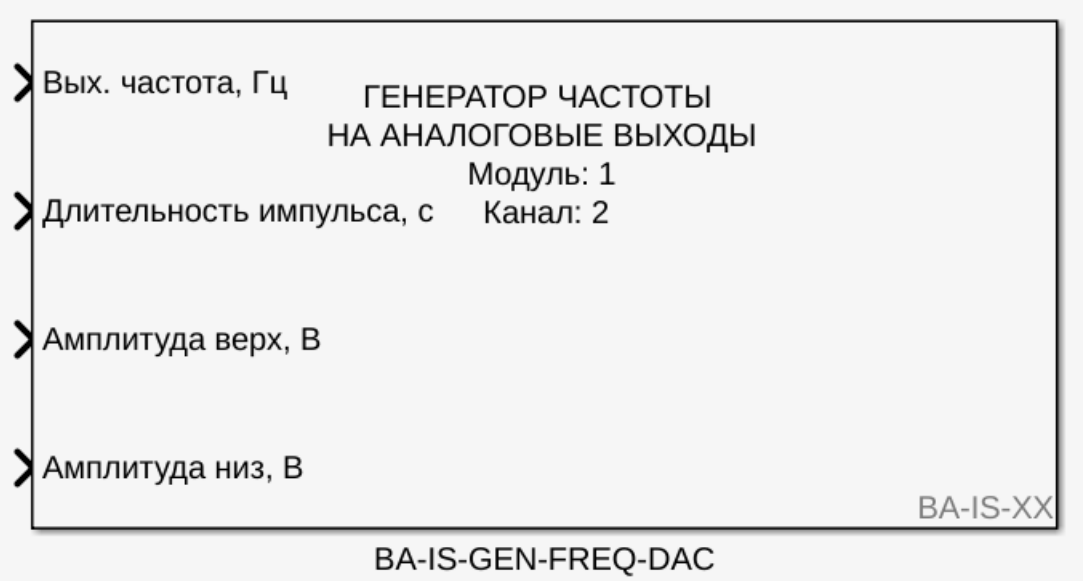
- Описание
- Библиотека блоков
  - Блок генератор частоты на аналоговые выходы

Описание

Настраиваемый модуль генератор частоты на аналоговые выходы на базе ПЛИС для машины реального времени РИТМ.

- ЧИМ/ШИМ сигналы;
- Частота на выходе 0 Гц до 100 кГц;
- Уровень +-10 В.

Библиотека блоков

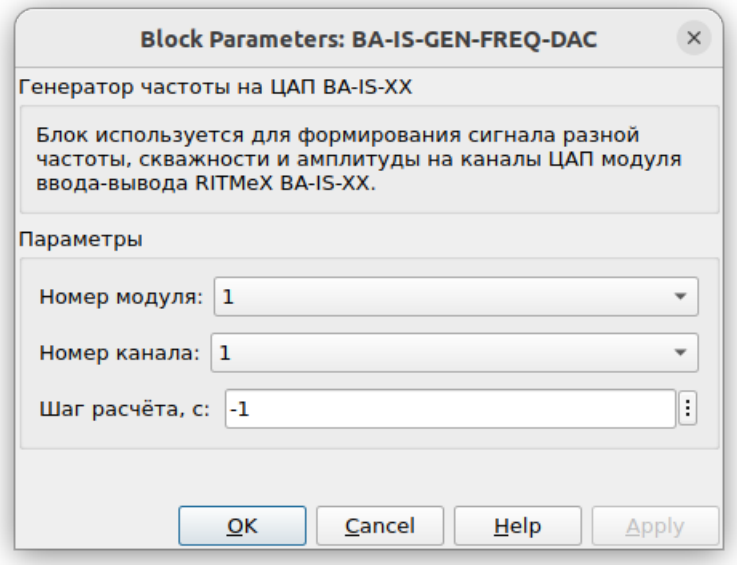


BA-IS-GEN-FREQ-DAC

Блок генератор частоты на аналоговые выходы

Блок используется для формирования сигнала разной частоты, скважности и амплитуды на каналы ЦАП модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала аналогового выхода.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

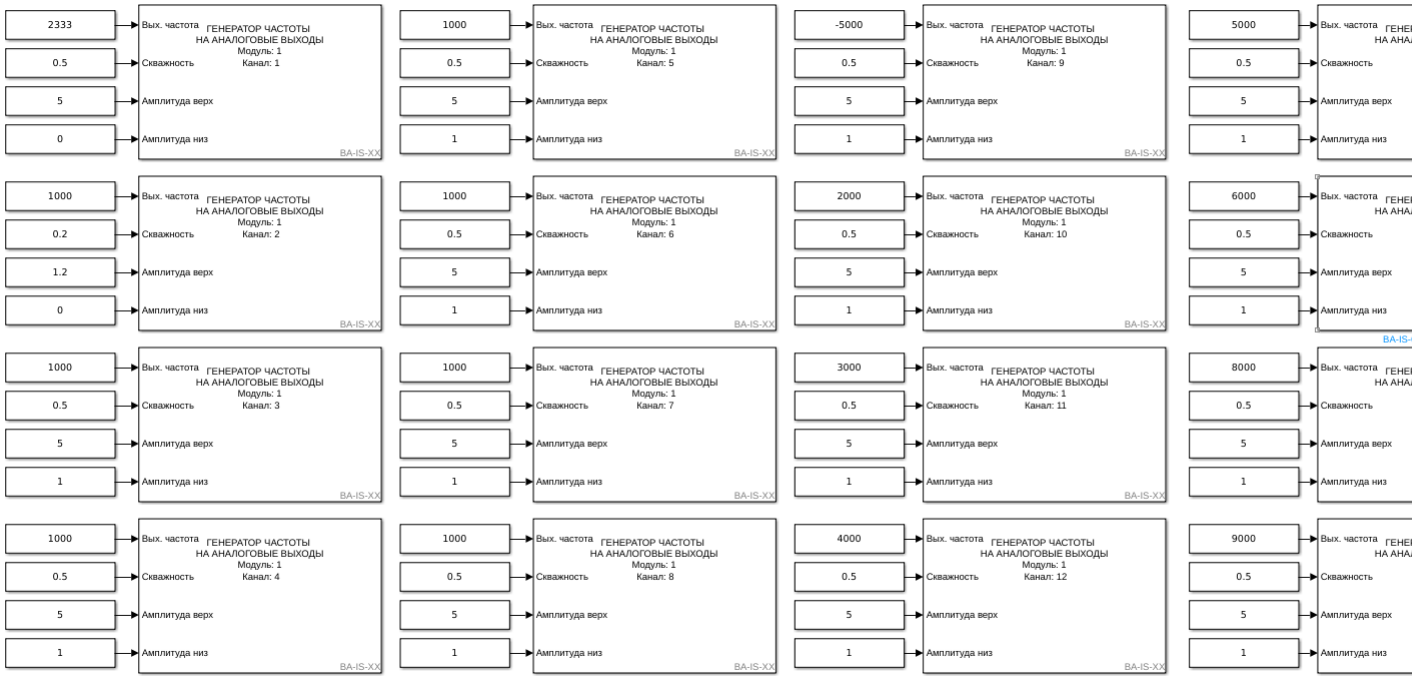
Входы блока

- **Вых. частота**: Выходная частота на канале. Тип данных *double*, от 0 до 100 кГц. Максимальная выходная частота зависит от одновременно включённых каналов.
- **Длительность импульса**: Длительность высокого (положительного) уровня импульса. Тип данных *double*, диапазон зависит от входной частоты.
- **Амплитуда верх**: Амплитуда верхнего значения ЧИМ/ШИМ. Тип данных *double*, может быть +- 10 В.
- **Амплитуда низ**: Амплитуда нижнего значения ЧИМ/ШИМ. Тип данных *double*, может быть +- 10 В.

Примеры и демонстрации BA-IS-GEN-FREQ-DAC

Пример работы с генератором частоты на каналами ЦАП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_gen_pfm_pwm.slx')
```



Модуль генератор синуса на аналоговые выходы

Содержание

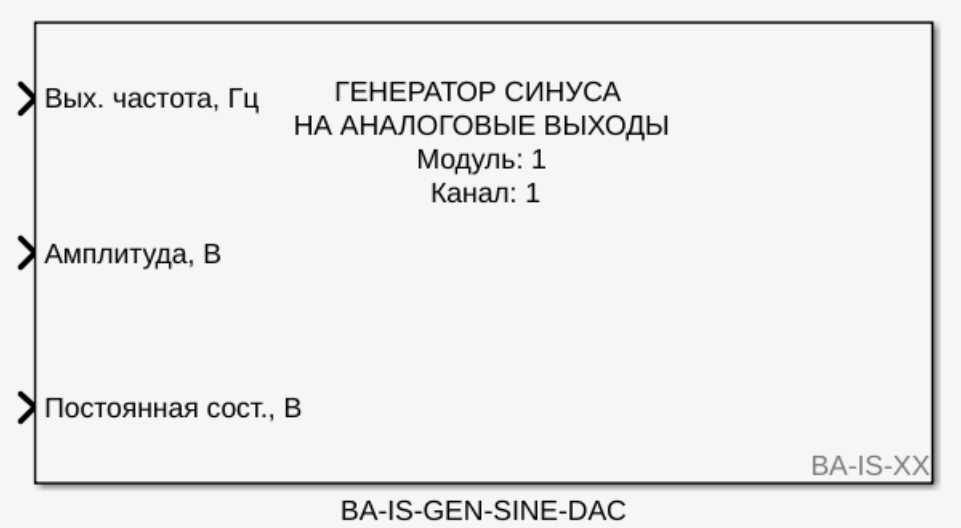
- Описание
- Библиотека блоков
  - Блок генератор синуса на аналоговые выходы

Описание

Настраиваемый модуль генератор синуса на аналоговые выходы на базе ПЛИС для машины реального времени РИТМ.

- Sine сигнал;
- Частота на выходе 0 Гц до 10 кГц;
- Уровень +-10 В.

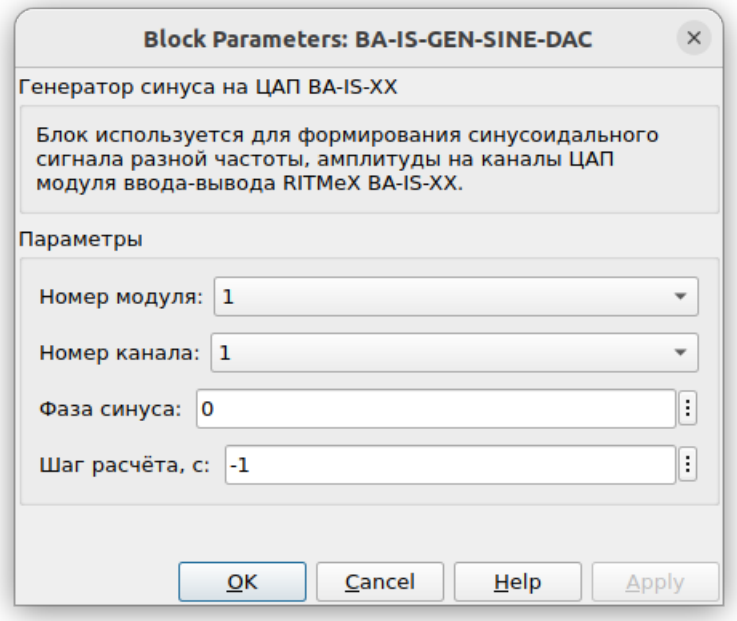
Библиотека блоков



Блок генератор синуса на аналоговые выходы

Блок используется для формирования синусоидального сигнала разной частоты, амплитуды на каналы ЦАП модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала аналогового выхода.
- ***Фаза синуса***: Задает фазу сигнала, в градусах.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

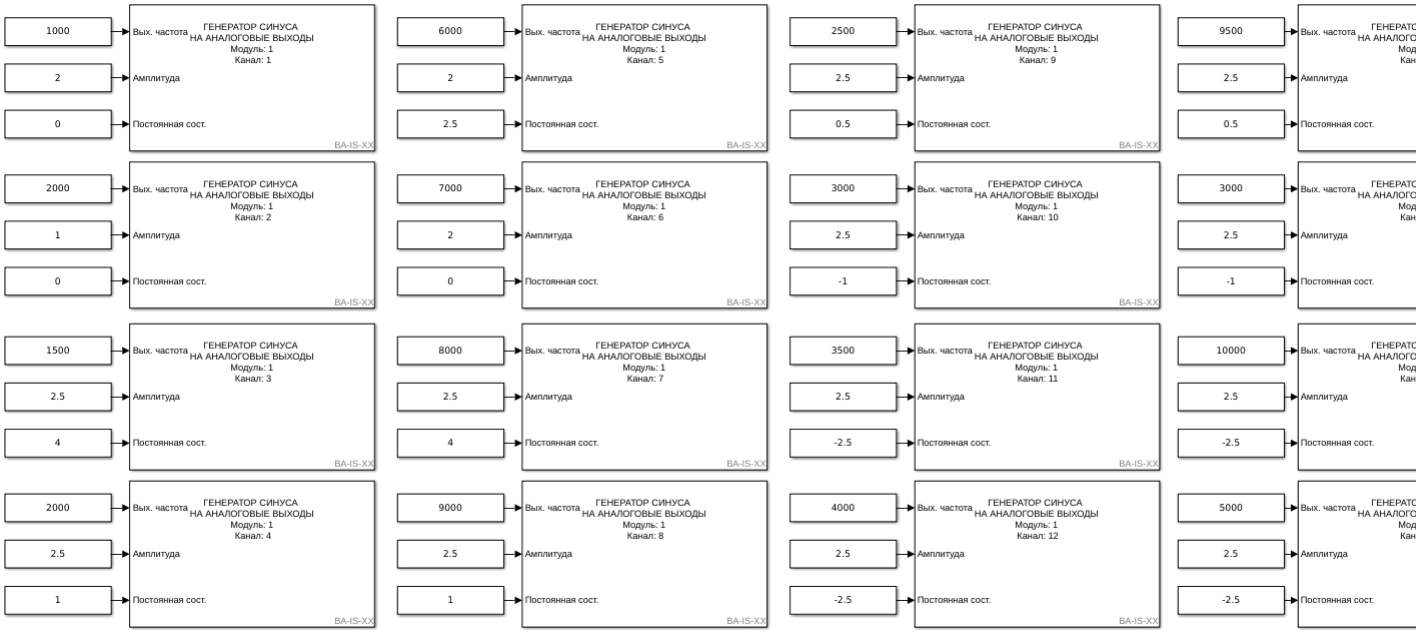
Входы блока

- **Вых. частота**: Выходная частота на канале. Тип данных *double*, от 0 до 10000 Гц (10 кГц). Максимальная выходная частота зависит от одновременно включённых каналов.
- **Амплитуда**: Задаёт амплитуду синусоидального сигнала. Тип данных *double*, может быть +- 10 В.
- **Постоянная сост.**: Смещение синусоидального сигнала. Тип данных *double*, может быть +- 10 В.

Примеры и демонстрации BA-IS-GEN-SINE-DAC

Пример работы с генератором синуса на ЦАП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_gen_sine.slx')
```



Модуль усреднения с аналоговых входов

Содержание

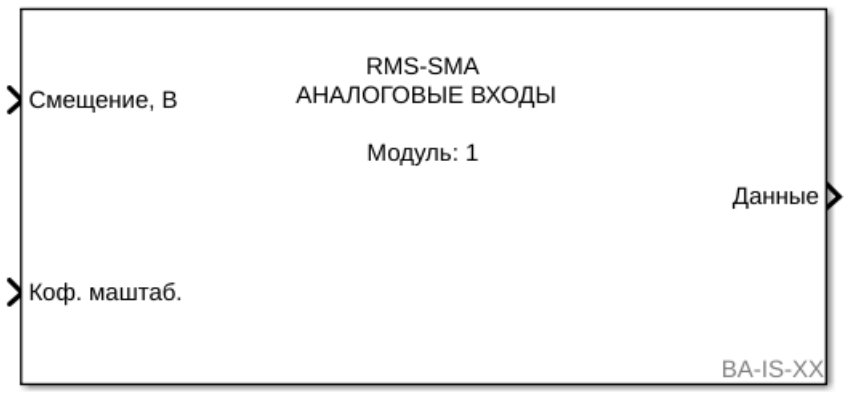
- Описание
- Библиотека блоков
- Блок BA-IS-ADC-RMS-SMA

Описание

Настраиваемый модуль усреднения RMS-SMA аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

- RMS - среднеквадратичное значение;
- SMA - простое среднее значение;
- 16 независимых входов;
- Входной диапазон напряжений: биполярный ±12.5 В, ±10 В, ±6.25 В, ±5 В, ±2.5 В униполярный 0 В до 12.5 В, 0 В до 10 В, 0 В до 5 В и биполярный дифференциальный ±20 В, ±12.5 В, ±10 В, ±5 В;
- Разрядность АЦП 16 бит;
- Одновременное считывание значений по всем каналам;
- Входное сопротивление каждого канала 1 МОм;
- Две полосы на каждый канал 25 кГц и 220 кГц;
- Частота дискретизации 1 МГц;
- Временное окно от 0.1 до 100 мс.

Библиотека блоков

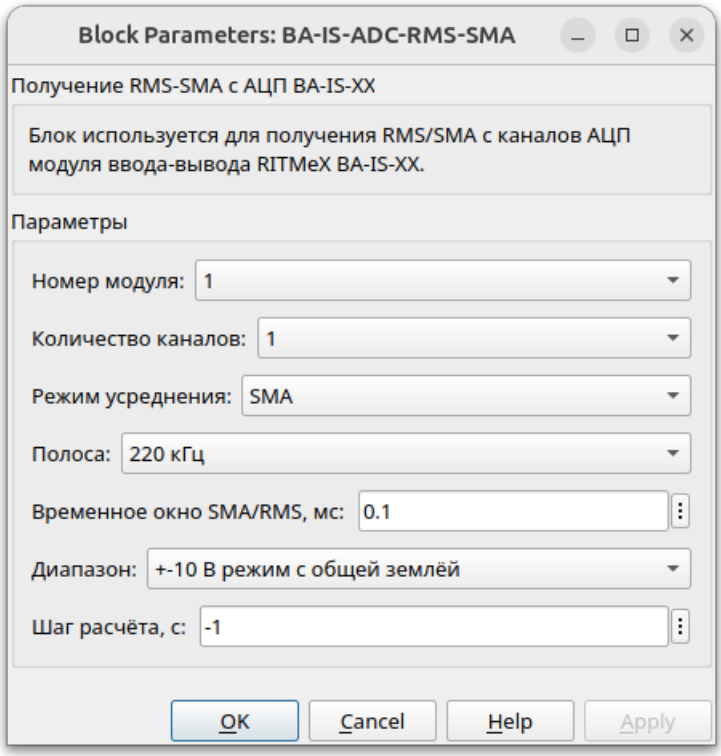


BA-IS-ADC-RMS-SMA

Блок BA-IS-ADC-RMS-SMA

Блок используется для получения RMS/SMA с каналов АЦП модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество каналов***: Задает количество каналов аналогового входа.
- ***Режим усреднения***: Задает режим усреднения SMA или RMS.
- ***Полоса***: Полоса пропускания 25 или 220 кГц.
- ***Временное окно SMA/RMS***: Задаёт временно окно для усреднения в мс.
- ***Диапазон***: Диапазон входного напряжения для каждого канала.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит получение данных с аналоговых входов.

Входы блока

- **Смещение**: Параметр для задания смещения сигнала на входе АЦП, для каждого канала индивидуально.
- **Коф. маштаб.**: Коэффициент масштабирования, безразмерная величина. Умножает канал на данный коэффициент, для каждого канала индивидуально.

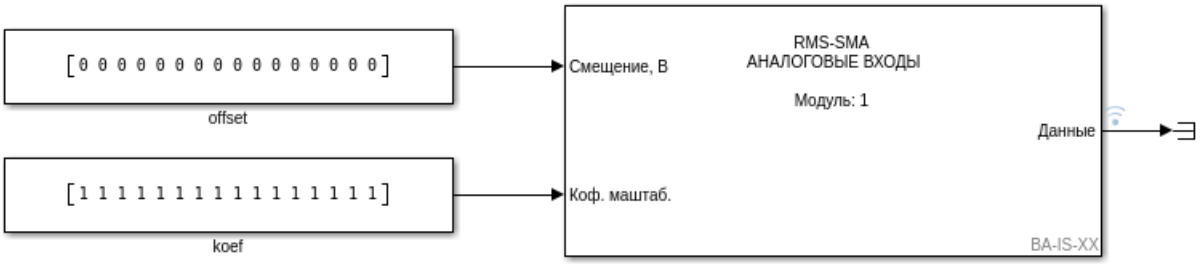
Выходы блока

- **Данные**: Данные, принятые с аналоговых входов и усреднённые за временное окно в зависимости от типа усреднения. Тип данных *double*, может быть массивом в зависимости от значения параметра *Количество каналов*.

Примеры и демонстрации BA-IS-ADC-RMS-SMA

Пример работы с каналами RMS/SMA АЦП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_rms_sma.slx')
```



Назначение пинов BA-IS-ADC-RMS-SMA

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модули на основе Fibre Channel

Содержание

- Модуль аналоговых выходов через протокол Аврора
- Модуль аналоговых входов-выходов через протокол Аврора
- Модули приемопередачи через интерфейс Aurora 8b10b
- Модуль Fibre Channel стандарта FC-AE-ASM

Модуль аналоговых выходов через протокол Аврора

Настраиваемый модуль аналоговых выходов на базе ПЛИС через протокол Аврора, поддерживающий до 12 аналоговых каналов на каждом из 4 каналов Fibre Channel.

Модуль аналоговых входов-выходов через протокол Аврора

Настраиваемый модуль аналоговых входов-выходов на базе ПЛИС через протокол Аврора, поддерживающий до 3 аналоговых каналов на каждом из 4 каналов Fibre Channel.

Модули приемопередачи через интерфейс Aurora 8b10b

Настраиваемые модули приемопередачи данных через интерфейс Aurora 8b10b на базе ПЛИС, поддерживающие 4 канала Fibre Channel.

Модуль Fibre Channel стандарта FC-AE-ASM

Программно-аппаратная реализация стандарта FC-AE-ASM для интерфейса Fibre Channel модуля ввода-вывода RITMeX BA-IS-XX с RITMeX FC-IS-FMC.

Модуль аналоговых выходов через протокол Аврора

Содержание

- Описание
- Требования к оборудованию
- Библиотека блоков

Описание

Настраиваемый модуль аналоговых выходов на базе ПЛИС для машины реального времени РИТМ. Модуль поддерживает работу с ЦАП внешнего устройства, взаимодействие с которым происходит по электроэнергетическому протоколу Аврора. Модуль протестирован с внешним устройством *Ponovo PAV DIA02*.

- 12 каналов ЦАП на один канал Fiber Channel;
- Диапазон напряжений -10..+10 В;
- Одновременная установка значений по всем каналам;
- Возможность использовать до четырёх внешних устройств, по одному на каждый канал Fibre Channel.

Требования к оборудованию

- Модуль RITMeX FC-IS-FMC;
- Базовая плата RITMeX BA-IS-45 или RITMeX BA-IS-55.

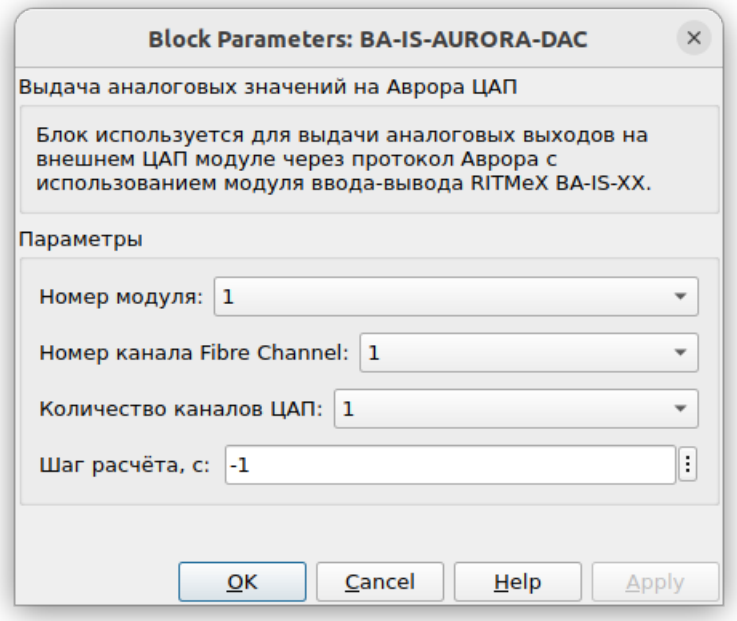
Библиотека блоков

BA-IS-AURORA-DAC

Блок Аврора ЦАП

Блок используется для выдачи аналоговых выходов на внешнем ЦАП модуле через протокол Аврора с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Требования к блоку

- При работе с устройством *Ponovo PAV DIA02* шаг расчёта блока должен быть меньше 1 мс.

Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала Fibre Channel***: Задает номер канала Fibre Channel модуля RITMeX FC-IS-FMC.
- ***Количество каналов ЦАП***: Задает количество каналов аналогового выхода.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

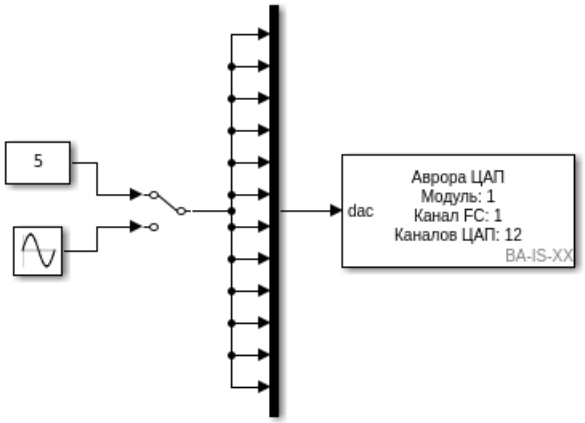
Входы блока

- **dac**: Данные для отправки на аналоговые выходы Аврора ЦАП. Тип данных *double*, может быть массивом в зависимости от значения параметра *Количество каналов ЦАП*.

Примеры и демонстрации BA-IS-AURORA-DAC

Пример работы с каналами Аврора ЦАП модуля BA-IS-XX с использованием поставляемого модуля RITMeX FC-IS-FMC.

```
open_system('ritmdemo_block_bais_aurora_dac.slx')
```

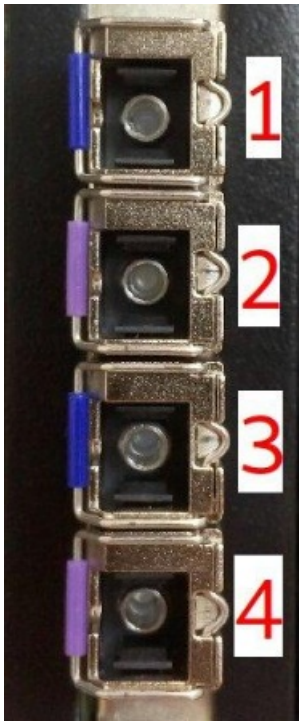


Назначение пинов BA-IS-AURORA-DAC

Модуль BA-IS-XX поставляется с RITMeX FC-IS-FMC.

В RITMeX FC-IS-FMC каждый разъем представляет собой отдельный канал Fibre Channel, который может использоваться для передачи данных Аврора ЦАП.

Цвет разъёма указывает на длину волны, которую использует оптический приёмопередатчик. Нужно учитывать длину волны для корректной работы оптического приёмопередатчика. Если в RITMeX FC-IS-FMC установлен оптический приёмопередатчик с длиной волны передачи 1550nm (фиолетовый цвет), то он должен быть соединён оптическим кабелем с приёмопередатчиком, имеющим длину волны приёма 1550nm (синий цвет).



Модули приемопередачи через интерфейс Aurora 8b10b

Содержание

- Описание
- Требования к оборудованию
- Библиотека блоков
  - Блок Aurora 8b10b TX
  - Блок Aurora 8b10b RX

Описание

Настраиваемые модули приемопередачи пакетов данных на базе ПЛИС для машины реального времени РИТМ.

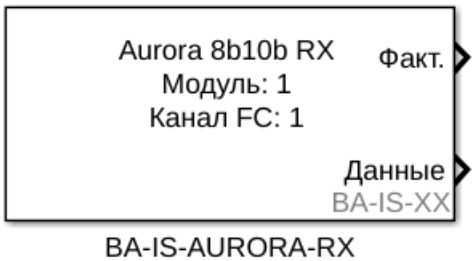
- Максимальное количество приемопередачи данных 2048 32-битных слов, за шаг расчёта;
- Минимальное количество приемопередачи данных 8 32-битных слов, за шаг расчёта;
- Поддержка 4 независимых каналов Aurora 8b10b;
- Line Rate Aurora 8b10b TX/RX 2Gbps;
- Line Width Aurora 8b10b TX/RX 4 Bytes;
- Режим работы Duplex, интерфейс Framing.

Требования к оборудованию

- Модуль RITMeX FC-IS-FMC;
- Базовая плата RITMeX BA-IS-45 или RITMeX BA-IS-55.

Библиотека блоков

BA-IS-AURORA-TX

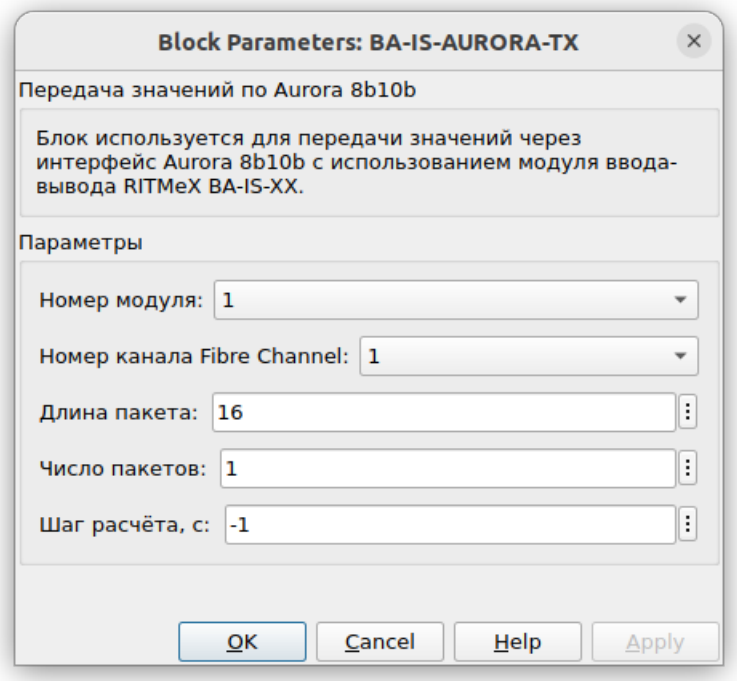


BA-IS-AURORA-RX

Блок Aurora 8b10b TX

Блок используется для передачи значений через интерфейс Aurora 8b10b с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала Fibre Channel***: Задает номер канала Fibre Channel модуля RITMeX FC-IS-FMC.
- ***Длина пакета***: Задает количество слов в пакете данных.
- ***Число пакетов***: Задает количество пакетов данных для передачи.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача данных.

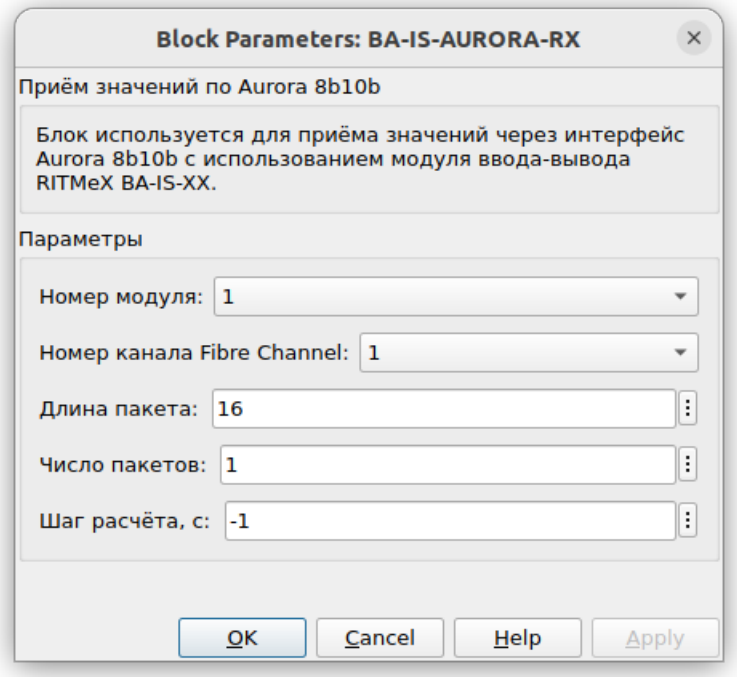
Вход блока

- **Данные**: Данные, для передачи через Aurora 8b10b. Тип данных *uint64*, может быть массивом в зависимости от значения параметра *Длина пакета*, из них могут быть значащие только первые 32 бита в каждом слове.

Блок Aurora 8b10b RX

Блок используется для приёма значений через интерфейс Aurora 8b10b с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала Fibre Channel***: Задает номер канала Fibre Channel модуля RITMeX FC-IS-FMC.
- ***Длина пакета***: Задает количество слов в пакете данных.
- ***Число пакетов***: Задает количество пакетов данных для приёма.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит приём данных.

Выход блока

- **Данные**: Данные, принятые Aurora 8b10b. Тип данных *uint64*, может быть массивом в зависимости от значения параметра *Длина пакета*, из них могут быть значащие только первые 32 бита в каждом слове.
- **Факт.**: Количество принятых пакетов данных на текущем шаге расчёта. Тип данных *uint32*.

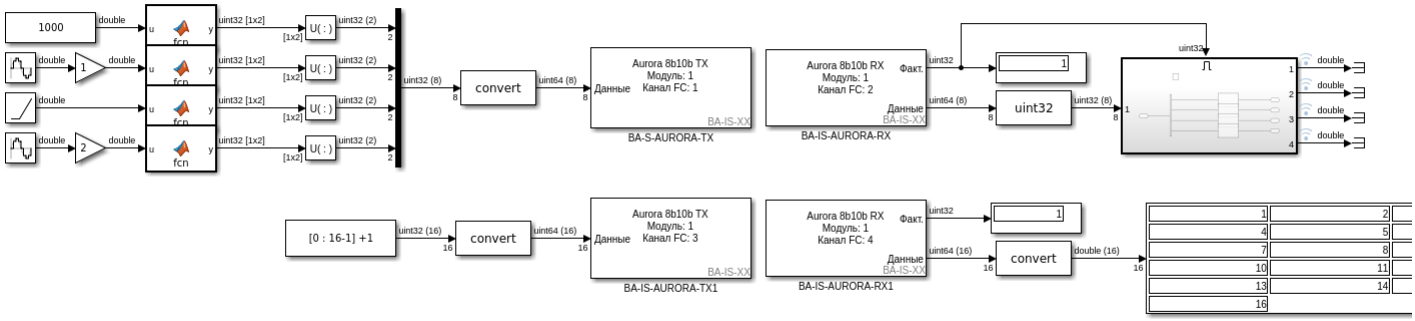
Примечание:

- Важно понимать значение выхода *Факт.* блока Aurora 8b10b RX. Если на текущем шаге расчёта по интерфейсу Aurora не были приняты данные, то на выходе *Факт.* будет значение *0*. В этом случае данные на выходе блока не определены (могут содержать мусор). Пользователю требуется обработать данную ситуацию - например, используя выход *Факт.* как условие для выполнения Enabled Subsystem в модели.
- Возможна ситуация, когда выход *Факт.* показывает большее число пакетов, чем выставлено в параметре *Число пакетов* в блоке Aurora 8b10b RX. Это значит, что данных по интерфейсу Aurora приходит больше, чем блок Aurora 8b10b RX готов принять. В этом случае требуется увеличить значение параметра *Число пакетов*, чтобы не пропускать данные.

Примеры и демонстрации BA-IS-AURORA-TX и BA-IS-AURORA-RX

Пример работы с каналами Aurora 8b10b TX и Aurora 8b10b RX модуля BA-IS-XX с использованием поставляемого модуля RITMeX FC-IS-FMC. На модуле RITMeX FC-IS-FMC нужно соединить оптическим кабелем канал 1 с каналом 2 и канал 3 с 4.

```
open_system('ritmdemo_block_bais_aurora_tx_rx.slx')
```

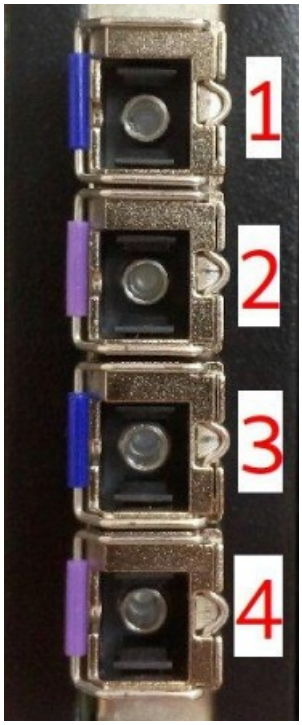


Назначение пинов BA-IS-AURORA-TX и BA-IS-AURORA-RX

Модуль BA-IS-XX поставляется с поставляется с модулем расширения RITMeX FC-IS-FMC.

В RITMeX FC-IS-FMC каждый разъем представляет собой отдельный канал Fibre Channel, который может использоваться для приемопередачи данных Aurora 8b10b.

Цвет разъёма указывает на длину волны, которую использует оптический приёмопередатчик. Нужно учитывать длину волны для корректной работы оптического приёмопередатчика. Если в RITMeX FC-IS-FMC установлен оптический приёмопередатчик с длиной волны передачи 1550nm (фиолетовый цвет), то он должен быть соединён оптическим кабелем с приёмопередатчиком, имеющим длину волны приёма 1550nm (синий цвет).



Модуль Fibre Channel стандарта FC-AE-ASM

Содержание

- Описание

Описание

Программно-аппаратная реализация стандарта FC-AE-ASM для интерфейса Fibre Channel модуля ввода-вывода RITMeX BA-IS-XX с RITMeX FC-IS-FMC.

- Количество каналов Fibre Channel: 4;
- Скорость передачи информации: 2,125 Гбит/с.

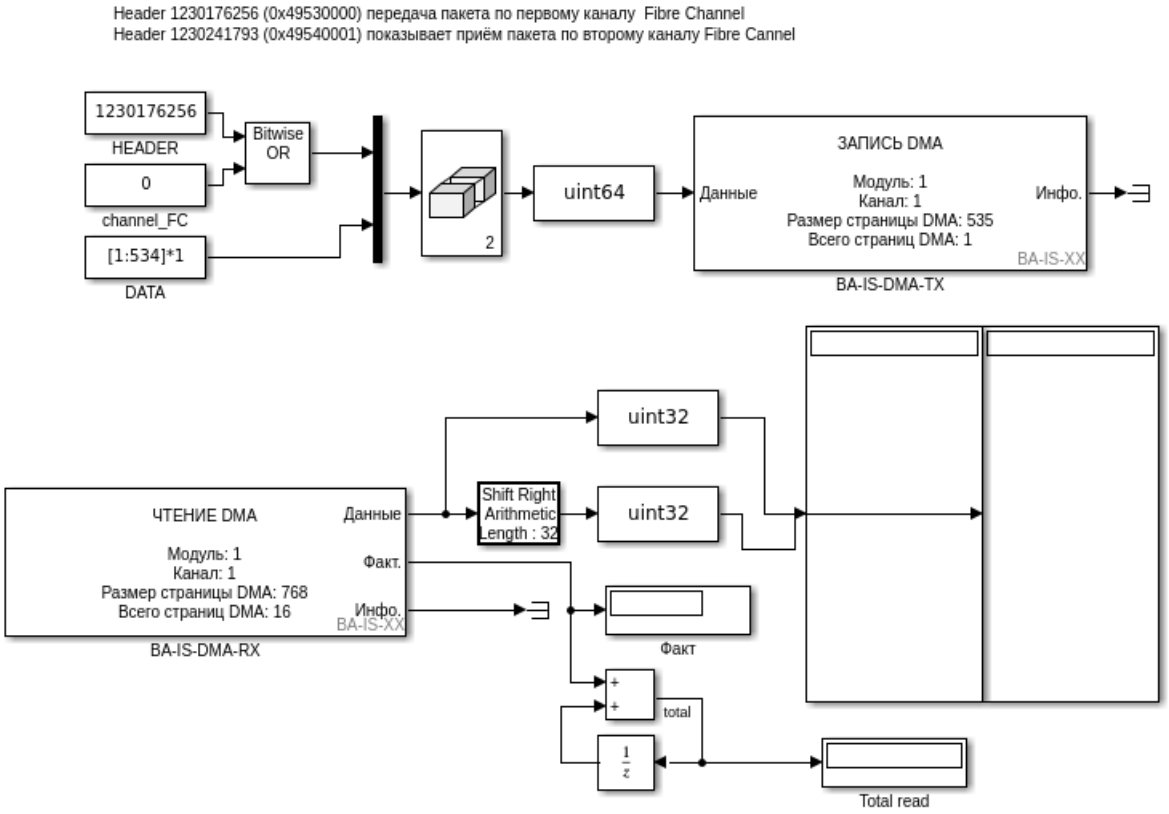
Подробное описание протокола в стандарте FC-AE-ASM.

Модуль встроен в специализированную прошивку RITMeX BA-IS-XX и доступен через Модули прямого доступа к памяти DMA.

Примеры и демонстрации BA-IS-DMA-TX/BA-IS-DMA-RX

Пример работы с DMA и задействованным Fibre Channel. Для работы требуется специализированная прошивка модуля BA-IS-XX. На модуле RITMeX FC-IS-FMC первый и второй канал должны быть соединены оптическим кабелем.

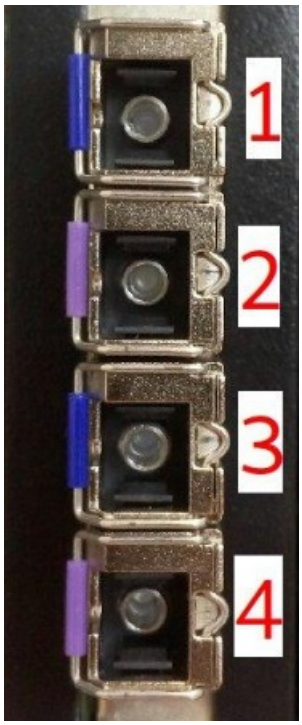
```
open_system('ritmdemo_block_bais_dma_fc.slx')
```



Назначение пинов Fibre Channel

Модуль BA-IS-XX поставляется с RITMeX FC-IS-FMC и специализированной прошивкой. В RITMeX FC-IS-FMC каждый разъем представляет собой отдельный канал Fibre Channel.

Цвет разъёма указывает на длину волны, которую использует оптический приёмопередатчик. Нужно учитывать длину волны для корректной работы оптического приёмопередатчика. Если в RITMeX FC-IS-FMC установлен оптический приёмопередатчик с длиной волны передачи 1550nm (фиолетовый цвет), то он должен быть соединён оптическим кабелем с приёмопередатчиком, имеющим длину волны приёма 1550nm (синий цвет).



Модули прямого доступа к памяти DMA

Содержание

- DMA для записи данных в ПЛИС
- DMA для чтения данных из ПЛИС

DMA для записи данных в ПЛИС

Настраиваемый модуль DMA для записи данных между ПЛИС и процессором, поддерживающий до 4 независимых каналов записи.

DMA для чтения данных из ПЛИС

Настраиваемый модуль DMA для чтения данных между ПЛИС и процессором, поддерживающий до 4 независимых каналов чтения.

DMA для записи данных в ПЛИС

Содержание

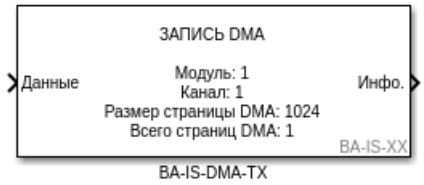
- Описание
- Библиотека блоков

Описание

Блок используется для записи DMA с использованием модуля ввода-вывода RITMeX BA-IS-XX. Блок настраивает канал DMA между *DUT* и *SW* частью модели, выделяя память для прямого доступа между процессором и ПЛИС. Память разбивается на страницы, которые объединены друг с другом в кольцевой буфер через дескрипторы. Общий размер выделенной памяти зависит от параметров *Число страниц DMA для записи* и *Число слов DMA для записи*. В зависимости от природы передаваемых данных пользователь может настраивать параметры DMA под конкретную задачу. Данные записываются блоком страницами дескрипторов в зависимости от выбранного параметра *Число страниц DMA для записи*.

- 4 независимых канала записи;
- 64-битный размер слова.

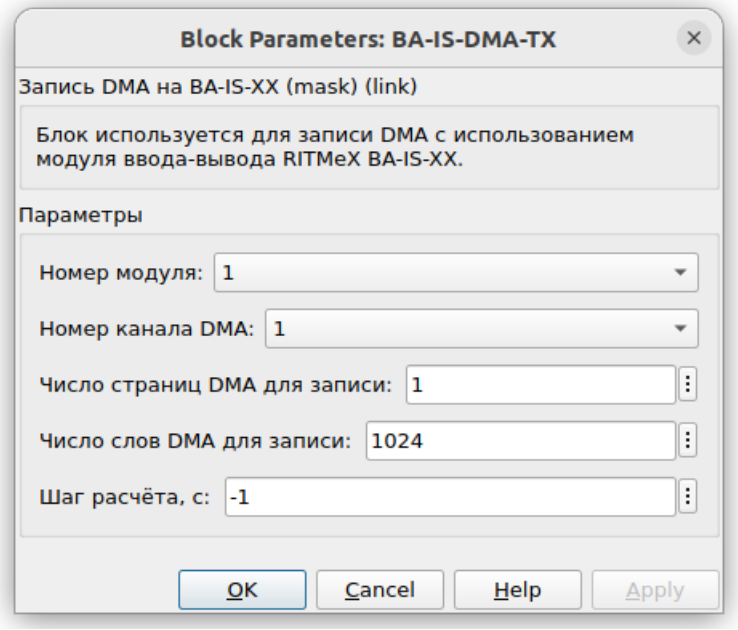
Библиотека блоков



Блок запись DMA

Блок используется для записи DMA с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Требования к блоку

Важно подобрать правильное сочетание параметров *Число страниц DMA для записи*, *Число слов DMA для записи* и *Шага расчёта*, чтобы гарантировать отсутствие пропусков DMA страниц при записи.

Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала DMA***: Задает номер канала DMA модуля RITMeX BA-IS-XX.
- ***Число страниц DMA для записи***: Задает количество желаемых страниц записи.
- ***Число слов DMA для записи***: Задает размер страниц дескрипторов записи.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача страниц дескрипторов.

Входы блока

- **Данные**: Данные переданные блоком запись DMA. Тип данных *uint64*, может быть матрицей в зависимости от значения параметра *Число страниц DMA для записи* и *Число слов DMA для записи*.
- **Инфо.**: Служебная информация о канале DMA. Не документировано.

DMA для чтения данных из ПЛИС

Содержание

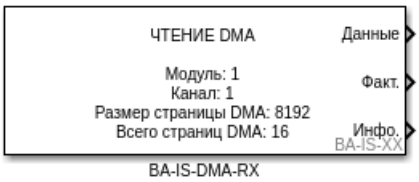
- Описание
- Библиотека блоков

Описание

Блок используется для чтения DMA с использованием модуля ввода-вывода RITMeX BA-IS-XX. Блок настраивает канал DMA между *DUT* и *SW* частью модели, выделяя память для прямого доступа между ПЛИС и процессором. Память разбивается на страницы, которые объединены друг с другом в кольцевой буфер через дескрипторы. Общий размер выделенной памяти зависит от параметров *Всего страниц DMA* и *Размер страницы DMA*. В зависимости от природы передаваемых данных пользователь может настраивать параметры DMA под конкретную задачу. Данные считаются блоком страницами дескрипторов в зависимости от выбранного параметра *Число страниц DMA для чтения*.

- 4 независимых канала чтения;
- 64-битный размер слова.

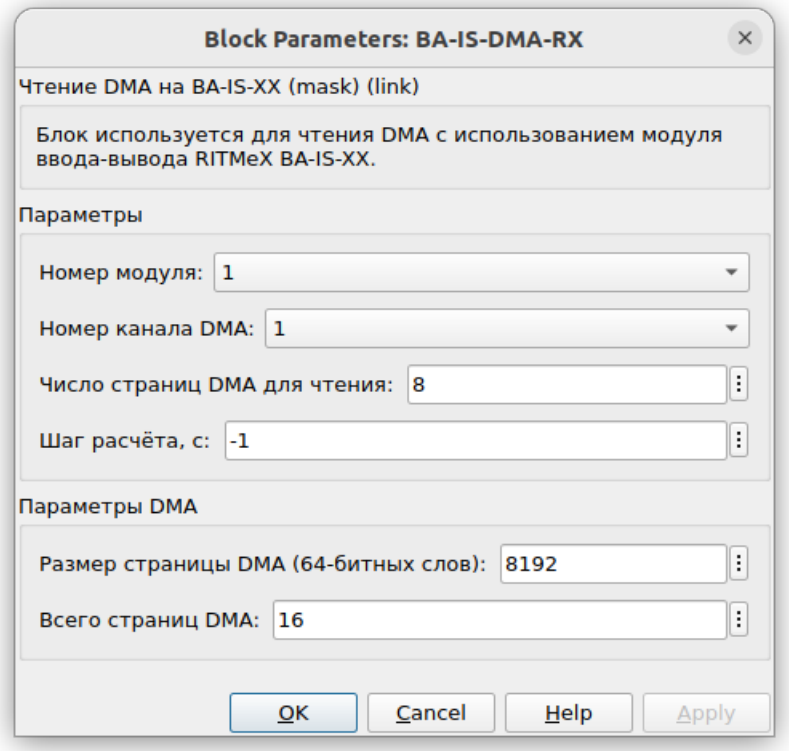
Библиотека блоков



Блок чтение DMA

Блок используется для чтения DMA с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Требования к блоку

Важно подобрать правильное сочетание параметров *Число страниц DMA для чтения*, *Всего страниц DMA* и *Шага расчёта*, чтобы гарантировать отсутствие пропусков DMA страниц при чтении. Выход *Факт* помогает подобрать такое сочетание.

Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала DMA***: Задает номер канала DMA модуля RITMeX BA-IS-XX.
- ***Число страниц DMA для чтения***: Задает количество желаемых страниц чтения.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача страниц дескрипторов.
- ***Размер страницы DMA (64-битных слов)***: Задает размер страниц дескрипторов.
- ***Всего страниц DMA:***: Задает количество страниц дескрипторов.

Входы блока

- **Данные**: Данные принятые блоком чтение DMA. Тип данных *uint64*, может быть матрицей в зависимости от значения параметра *Число страниц DMA для чтения* и *Размер страницы DMA (64-битных слов)*.
- **Факт.**: Показывает количество страниц DMA, фактически доступных (содержащих новые данные) с момента предыдущего вызова блока. Если выход Факт больше, чем параметр Число страниц DMA для чтения, то следует увеличить параметр Число страниц DMA для чтения или уменьшить шаг расчёта блока.
- **Инфо.**: Служебная информация о канале DMA. Не документировано.

ПРИМЕЧАНИЕ 1:

Важно подобрать правильное сочетание параметров *Число страниц DMA для чтения* и

Шаг расчёта, чтобы гарантировать отсутствие пропусков DMA страниц при чтении. Выход *Факт.* помогает подобрать такое сочетание.
Например, пусть в канал DMA поступают данные, формирующие равномерный поток в 5000 DMA страниц/секунду. Если задать параметр *Шаг расчёта* как 0.01 с, то за один шаг расчёта блок должен прочитать 50 страниц DMA. Если задать при этом параметр *Число страниц DMA* для чтения как 20, то на каждом шаге расчёта блока будет происходить пропуск 30 страниц DMA и будут прочитаны только 20 страниц. При этом выход *Факт.* будет показывать число реально доступных страниц DMA как 50.
Вывод: поэтому общее правило такое, если выход *Факт.* систематически выдает значение, превышающее параметр *Число страниц DMA для чтения*, то следует либо увеличить значение параметра *Число страниц DMA для чтения*, либо уменьшить значение параметра *Шаг расчёта*.

ПРИМЕЧАНИЕ 2:
Важно правильно обрабатывать в модели массив страниц DMA, выдаваемый на выходе блока Чтение DMA. * Выход *Факт.* определяет количество DMA страниц, прочитанных на данном шаге расчета. В модели следует обработать именно такое количество кадров. Страницы за пределом этого значения не определены (мусор). * В зависимости от природы передаваемых данных, в одной странице DMA может содержаться меньше данных, чем размер страницы. В протоколе обмена на стороне ПЛИС следует учесть это и передавать фактический размер данных в пределах одной страницы DMA. В модели следует обработать именно такое количество данных этой страницы. Данные за пределом этого значения не определены (мусор).

Модуль Цифровых входов-выходов

Содержание

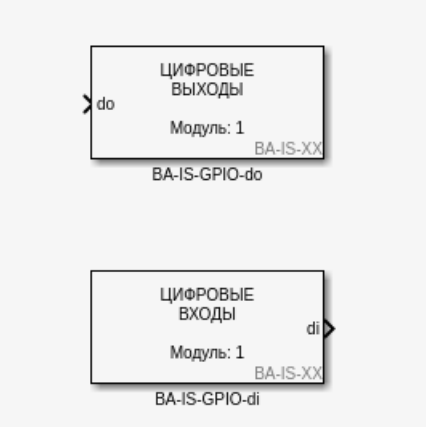
- Описание
- Библиотека блоков
  - Блок Цифровых входов
  - Блок Цифровых выходов

Описание

Настраиваемый модуль цифровых входов и выходов на базе ПЛИС, поддерживающий до 128 цифровых каналов в машине реального времени РИТМ.

- Количество входов: до 128 (в зависимости от конфигурации ПЛИС).
- Уровень напряжения: 3.3 В LVTTL или 5 В TTL (выбирается программно).
- Максимальная частота ввода/вывода: 5 МГц.

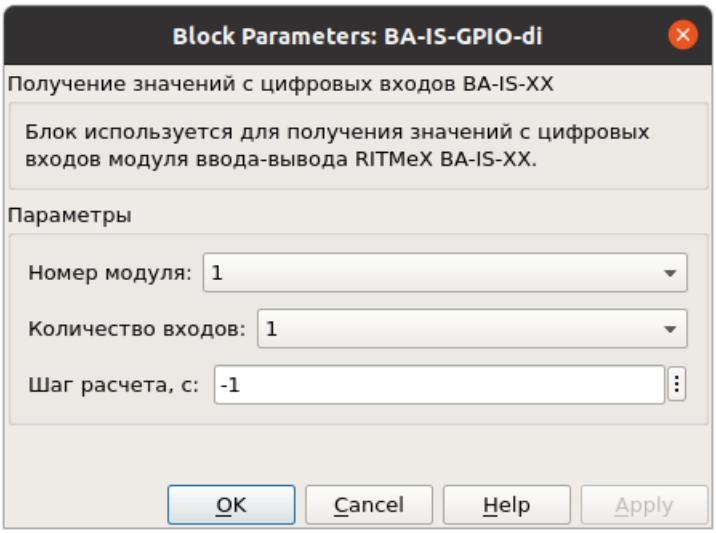
Библиотека блоков



Блок Цифровых входов

Блок используется для получения значений с цифровых входов модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество входов***: Позволяет выбрать количество цифровых входов.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах для получения отсчетов с цифровых каналов входа в приложение реального времени.

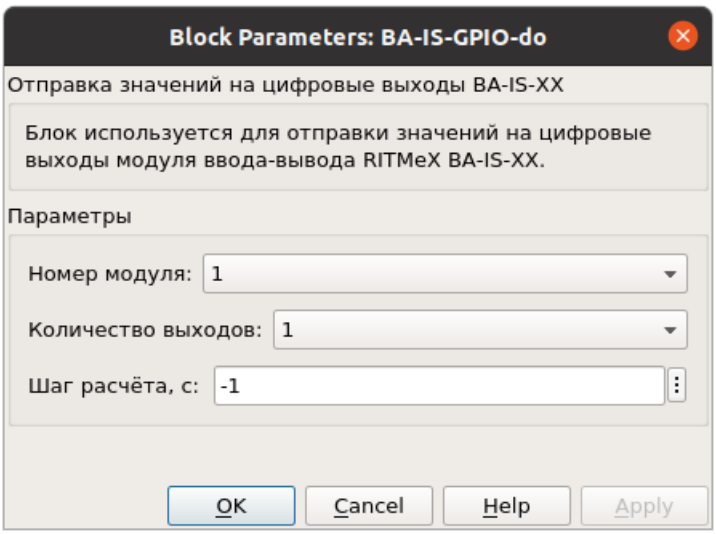
Выходы блока

- **di**: На выходе блока формируется массив отсчетов типа double, принятых с цифровых входов. Размер массива соответствует параметру “Количество входов” в блоке.

Блок Цифровых выходов

Блок используется для отправки значений на цифровые выходы модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Количество выходов***: Позволяет выбрать количество цифровых выходов.
- ***Шаг расчёта***: Позволяет задать шаг расчёта секундах для отправки отсчетов на цифровые каналы выхода из приложения реального времени.

Входы блока

- **do**: На вход блока поступает массив отсчетов типа double для передачи на цифровые выходы. Размер массива соответствует параметру “Количество выходов” в блоке.

Примеры и демонстрации BA-IS-GPIO

Пример работы с цифровыми входами и выходами модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_gpio.slx')
```

Назначение пинов BA-IS-GPIO

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль Генерации ШИМ

Содержание

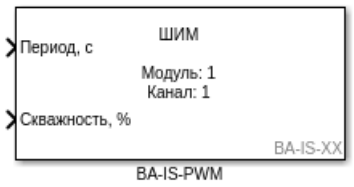
- Описание
- Библиотека блоков
  - Блок Генерации ШИМ

Описание

Настраиваемый модуль генерации ШИМ на базе ПЛИС, поддерживающий до 128 каналов генерации ШИМ в машине реального времени РИТМ.

- Количество каналов: до 128 (в зависимости от конфигурации ПЛИС).
- Частота вывода: 5 МГц.

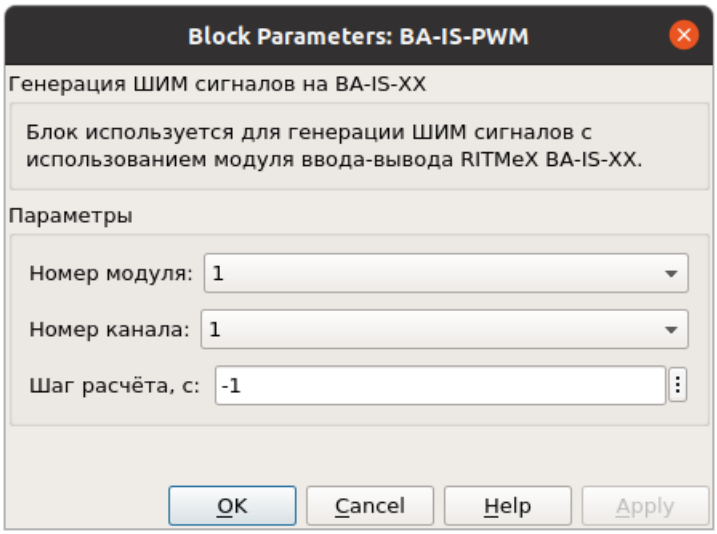
Библиотека блоков



Блок Генерации ШИМ

Блок используется для генерации ШИМ сигналов с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала ШИМ.

- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым значения периода и скважности ШИМ меняются из приложения реального времени.

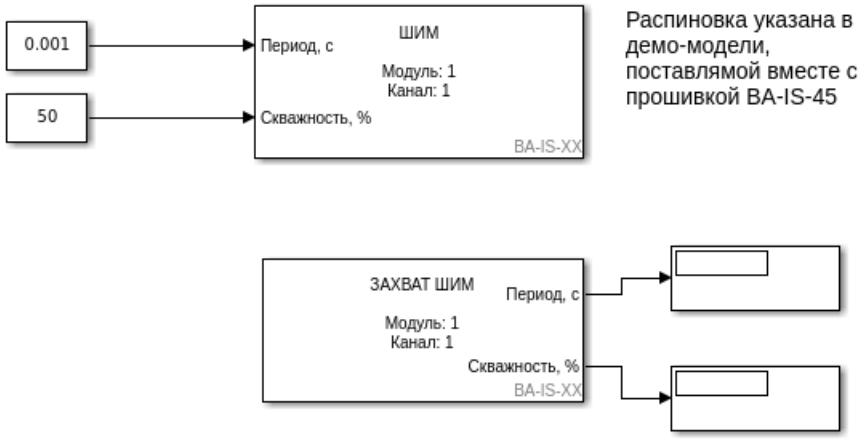
Входы блока

- **Период**: Значение периода ШИМ в секундах.
- **Скважность**: Значение скважности ШИМ в процентах (от 0 до 100 %).

Примеры и демонстрации BA-IS-PWM/BA-IS-CAP

Пример работы с каналами генерации и захвата ШИМ модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_pwmcap.slx')
```



Назначение пинов BA-IS-PWM

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль Захвата ШИМ

Содержание

- Описание
- Библиотека блоков
  - Блок Захвата ШИМ

Описание

Настраиваемый модуль захвата ШИМ на базе ПЛИС, поддерживающий до 128 каналов захвата ШИМ в машине реального времени РИТМ.

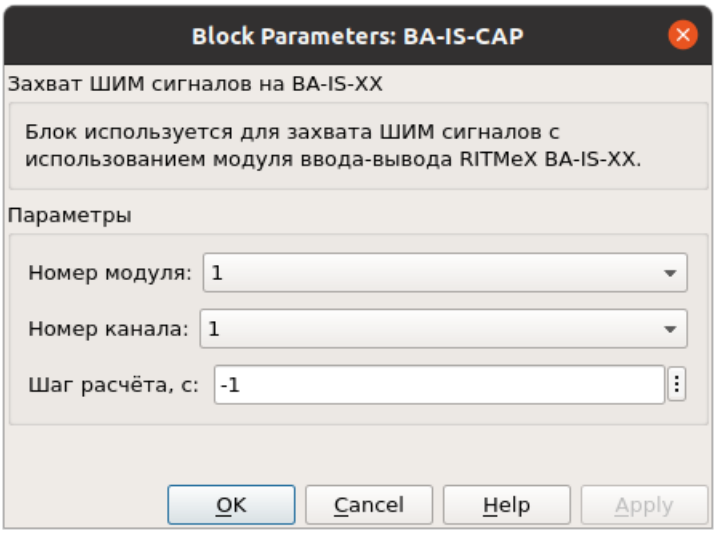
- Количество каналов: до 128 (в зависимости от конфигурации ПЛИС).
- Частота ввода: 5 МГц.

Библиотека блоков

Блок Захвата ШИМ

Блок используется для захвата ШИМ сигналов с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала захвата ШИМ.

- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым значения периода и скважности ШИМ доступны в приложении реального времени.

Выходы блока

- **Период**: Значение периода ШИМ в секундах. Период захватываемого сигнала ШИМ должен быть не больше 0.05 с (то есть не меньше 20 Гц).
- **Скважность**: Значение скважности ШИМ в процентах (от 0 до 100 %).

Назначение пинов BA-IS-CAP

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль SPI

Содержание

- Описание
- Библиотека блоков
  - Блок Работа с SPI

Описание

Настраиваемый модуль SPI на базе ПЛИС, поддерживающий SPI Master и SPI Slave в машине реального времени РИТМ.

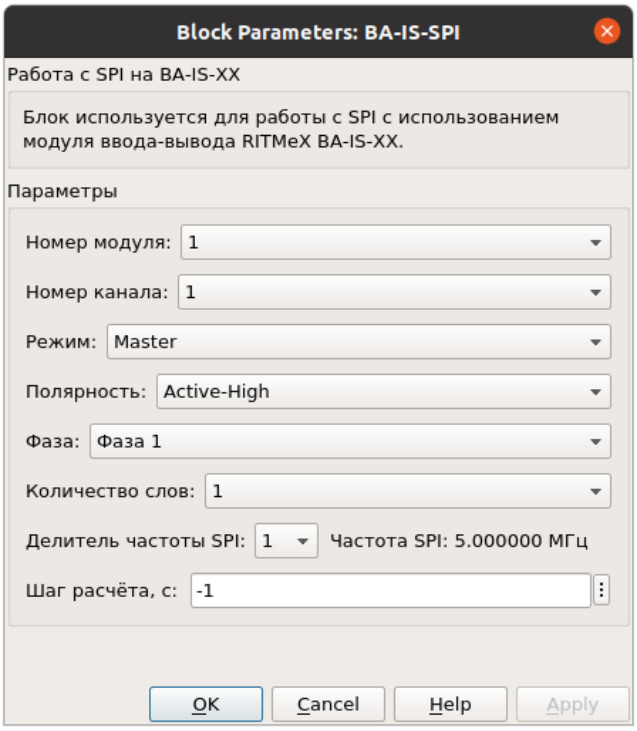
- Поддерживаемые режимы: SPI Master и SPI Slave.
- Частота SPI: до 5 МГц.

Библиотека блоков

Блок Работа с SPI

Блок используется для работы с SPI с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала SPI.
- ***Режим*** Режим работы SPI - Master или Slave.
- ***Полярность***: Задает полярность линии SCK. Возможные значения Active High и Active Low.

- ***Фаза***: Задает значение фазы для передачи данных. Возможные значение Фаза 1 и Фаза 2.
- ***Количество слов***: Количество слов для отправки и получения по SPI.
- ***Делитель частоты SPI***: Задает значение частоты сигнала SCK в режиме Master (от 100 кГц до 5 МГц).
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит обмен данными по SPI.

Входы блока

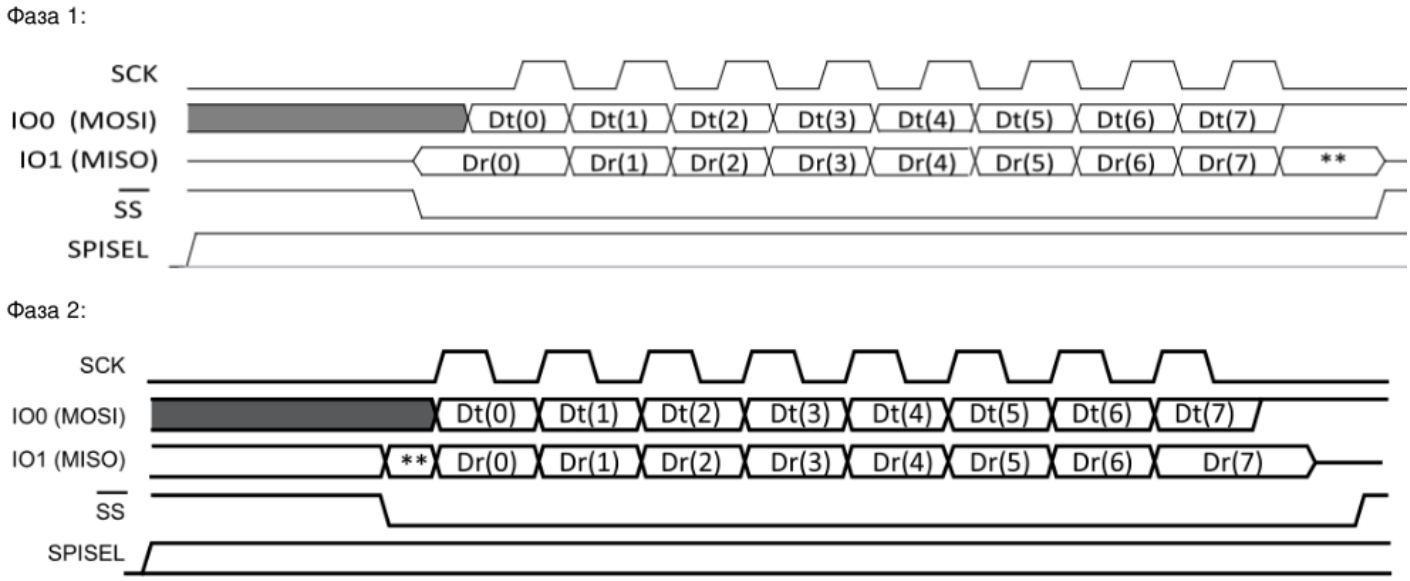
- **TX**: Данные для отправки по SPI. Тип данных uint8, может быть массивом в зависимости от значения параметра *Количество слов*.

Выходы блока

- **RX**: Данные, полученные по SPI. Тип данных uint8, может быть массивом в зависимости от значения параметра *Количество слов*.

Фаза и полярность

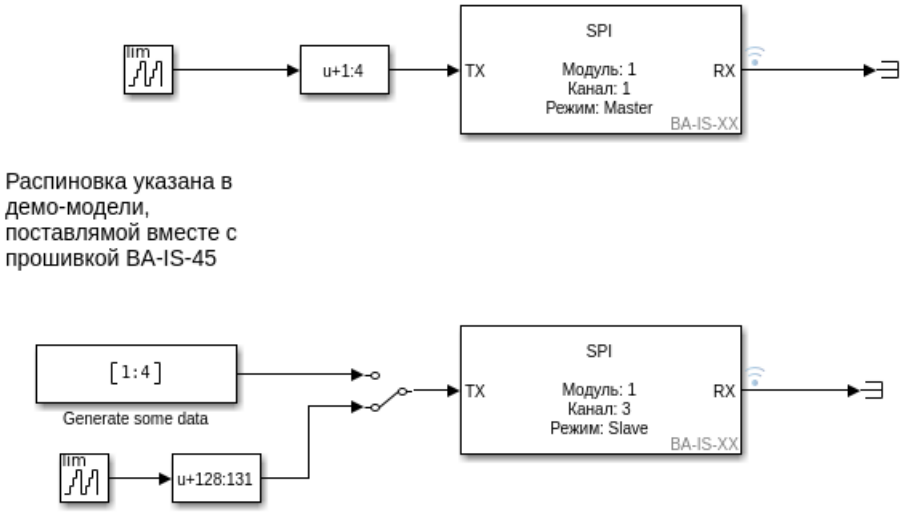
Ниже приводятся диаграмма, демонстрирующая режимы работы SPI с разными фазами.



Примеры и демонстрации BA-IS-SPI

Пример работы с каналами SPI Master и SPI Slave модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_spi.slx')
```



Назначение пинов BA-IS-SPI

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль I2C

Содержание

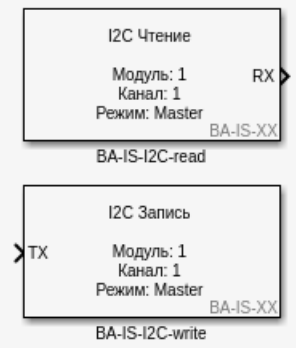
- Описание
- Библиотека блоков
  - Блок Запись в I2C
  - Блок Чтение с I2C

Описание

Настраиваемый модуль I2C на базе ПЛИС, поддерживающий I2C Master и I2C Slave в машине реального времени РИТМ.

- Поддерживаемые режимы: I2C Master и I2C Slave.
- Частота I2C: 100 кГц.

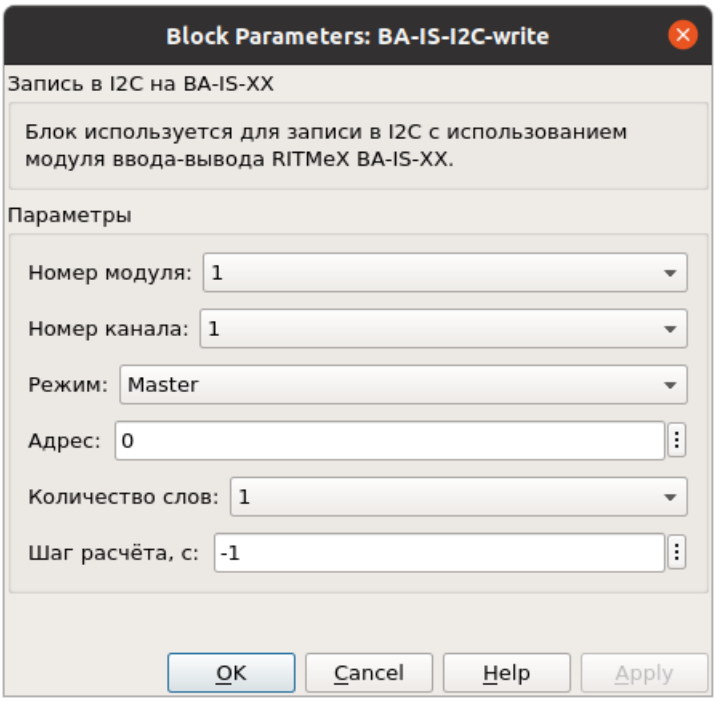
Библиотека блоков



Блок Запись в I2C

Блок используется для записи в I2C с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска Блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала I2C.
- ***Режим***: Режим работы I2C - Master или Slave.
- ***Адрес***: Задает адрес устройства I2C Slave на шине.
- ***Количество слов***: Количество слов для отправки и получения по I2C.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит обмен данными по I2C.

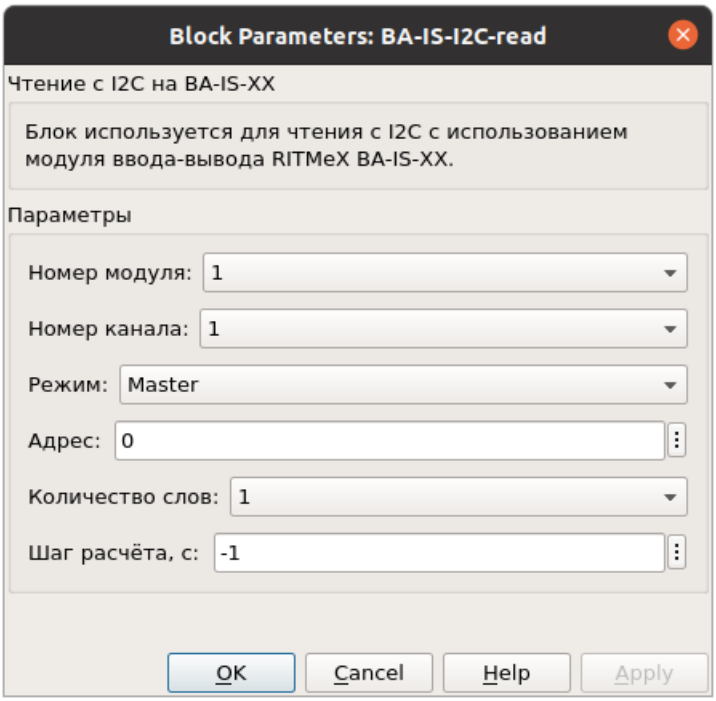
Входы блока

- **TX**: Данные для записи в I2C. Тип данных *uint8*, может быть массивом в зависимости от значения параметра *Количество слов*.

Блок Чтение с I2C

Блок используется для чтения с I2C с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска Блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала I2C.
- ***Режим***: Режим работы I2C - Master или Slave.
- ***Адрес***: Задает адрес устройства I2C Slave на шине.
- ***Количество слов***: Количество слов для отправки и получения по I2C.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит обмен данными по I2C.

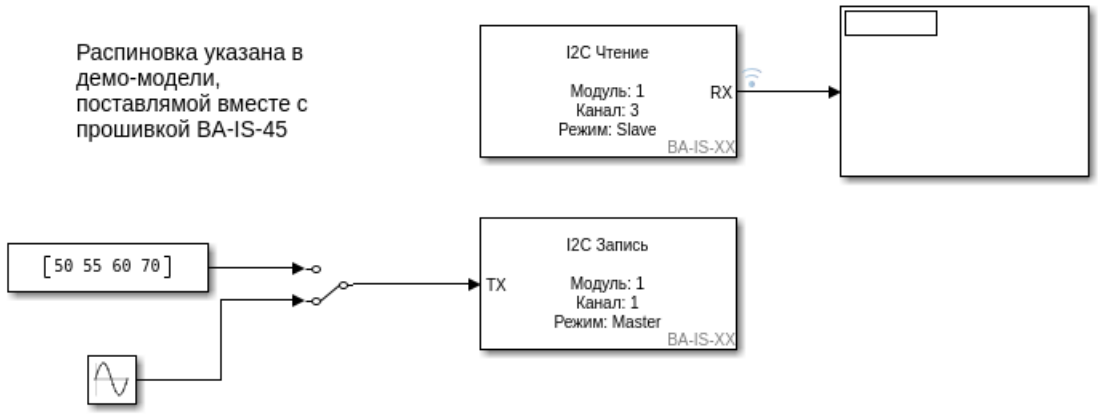
Выходы блока

- **RX**: Данные, прочитанные с I2C. Тип данных *uint8*, может быть массивом в зависимости от значения параметра *Количество слов*.

Примеры и демонстрации BA-IS-I2C

Пример работы с каналами I2C модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_i2c.slx')
```



Назначение пинов BA-IS-I2C

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль Квадратурного декодера и Квадратурного энкодера

Содержание

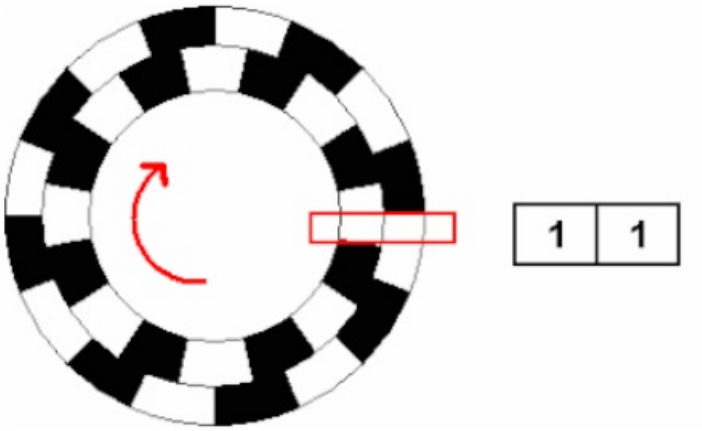
- Описание
- Заметки об использовании
- Библиотека блоков
  - Блок Квадратурный декодер
  - Блок Квадратурный энкодер

Описание

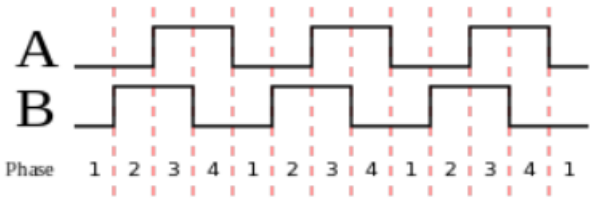
Настраиваемый модуль захвата и симуляции сигналов квадратурного энкодера в машине реального времени РИТМ.

Заметки об использовании

Блоки Квадратурный декодер и Квадратурный энкодер оперируют понятием “счетчик” для описания входов и выходов в терминах сегментов квадратурного энкодера. Физический квадратурный энкодер генерирует два сигнала, A и B, которые могут быть равны нулю или единице:



Временная диаграмма сигналов A и B выглядит следующим образом в случае вращения против часовой стрелки (когда фаза сигнала B опережает фазу сигнала A):



Сегментом называется каждая отдельная фаза квадратурного энкодера, т.е. сочетание значений сигналов A и B.

Индексной меткой называется условная граница на окружности, проход которой обозначет прямоугольный импульс на линии C. Индескная метка указывает на начальное положение. К примеру, если произвести поворот на 90 градусов против часовой стрелки,

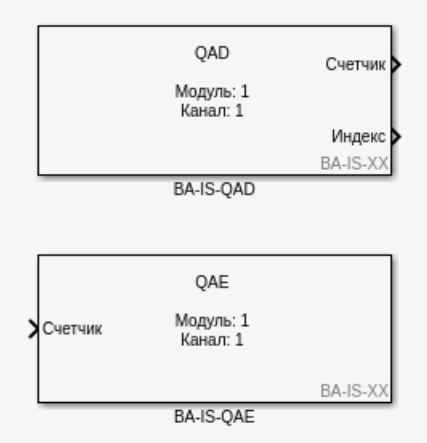
затем на 90 градусов по часовой стрелке, сгенерируется прямоугольный импульс на линии C.

Для блока Квадратурный декодер выход “Счетчик” означает количество сегментов, которые были получены (посчитаны) по линиям A и B за текущий шаг расчета модели. Чтобы преобразовать значение счетчика в угол поворота, необходимо узнать количество сегментов энкодера на один оборот (из даташита энкодера). Например, для 16-сегментного энкодера, значение счетчика “24” означает полтора оборота по часовой стрелке. А значение счетчика “-8” означает полоборота против часовой стрелки. “Индекс” обозначает количество полученных импульсов на линии C за текущий шаг расчета модели.

Для блока Квадратурный энкодер вход “Счетчик” означает количество сегментов, которые требуется выдать (сгенерировать) по линиям A и B за следующий шаг расчета модели. Например, если требуется повернуть виртуальный 16-сегментный энкодер на три четверти по часовой стрелке, значение счетчика должно равняться “12”. А если требуется повернуть энкодер на два оборота против часовой стрелки, значение счетчика должно равняться “-24”.

- Количество каналов: до 128 каналов, в зависимости от конфигурации ПЛИС.
- Частота сигналов A/B/C: до 1 МГц.
- Частота ввода/вывода: 5 МГц.

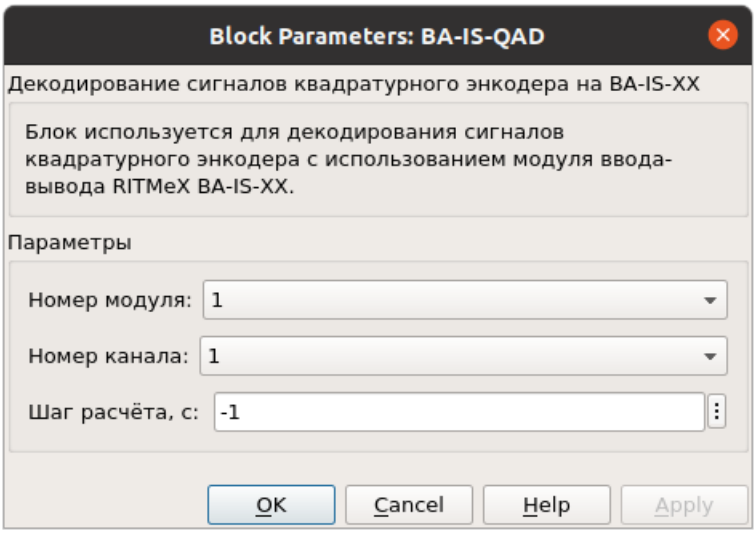
Библиотека блоков



Блок Квадратурный декодер

Блок используется для захвата сигналов квадратурного энкодера с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала квадратурного декодера.
- ***Шаг расчёта***: Позволяет задать шаг расчета в секундах, по которому значения счетчика и индекса выдаются в приложение реального времени.

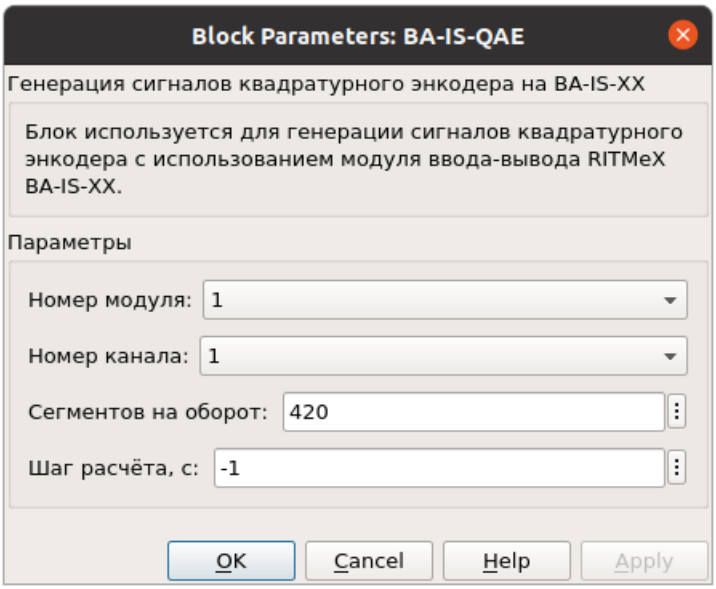
Выходы блока

- **Счетчик**: Количество сегментов энкодера, соответствующее углу поворота. Дополнительную информацию см. в разделе Заметки об использовании.
- **Индекс**: Количество импульсов прохода индексной метки за текущий шаг расчета модели.

Блок Квадратурный энкодер

Блок используется для симуляции сигналов квадратурного энкодера с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задает номер канала квадратурного декодера.
- ***Сегментов на оборот***: Задает количество сегментов на один оборот энкодера.
- ***Шаг расчёта***: Позволяет задать шаг расчета в секундах, по которому значения счетчика считываются из приложения реального времени.

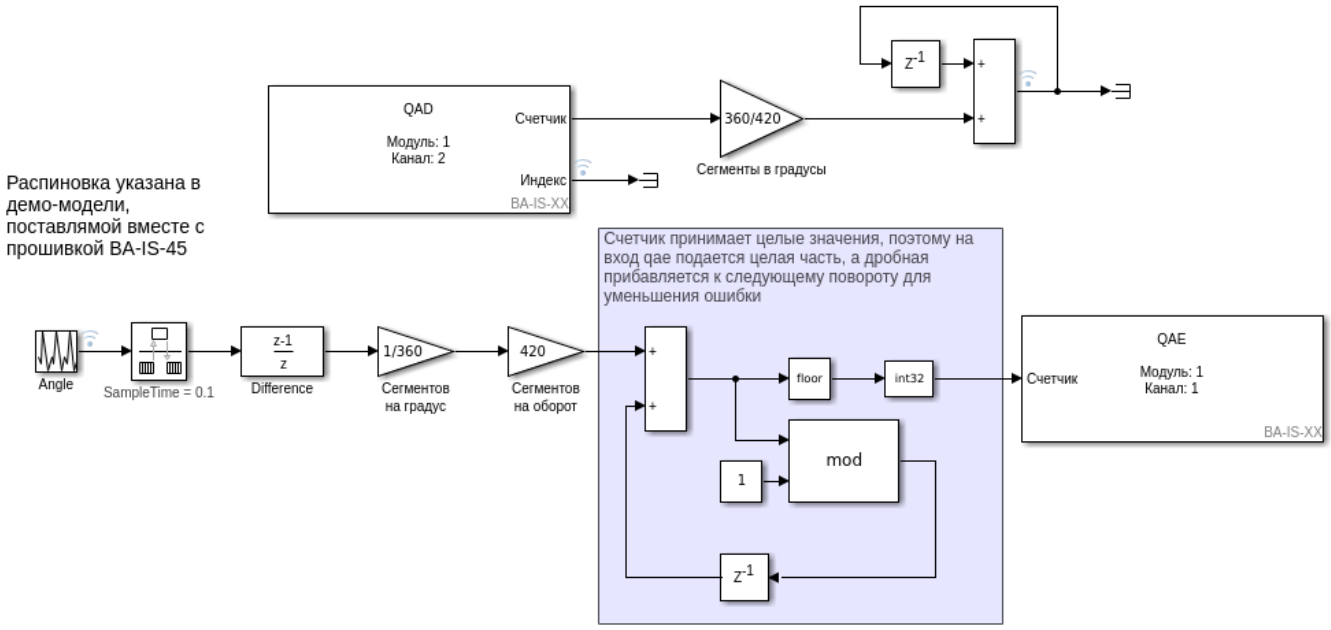
Входы блока

- **Счетчик**: Количество сегментов энкодера, соответствующее углу поворота. Дополнительную информацию см. в разделе Заметки об использовании.

Примеры и демонстрации BA-IS-QAD/BA-IS-QAE

Пример работы с квадратурным энкодером и декодером модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_qaeqad.slx')
```



Назначение пинов BA-IS-QAD/BA-IS-QAE

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль приемопередатчика UART

Содержание

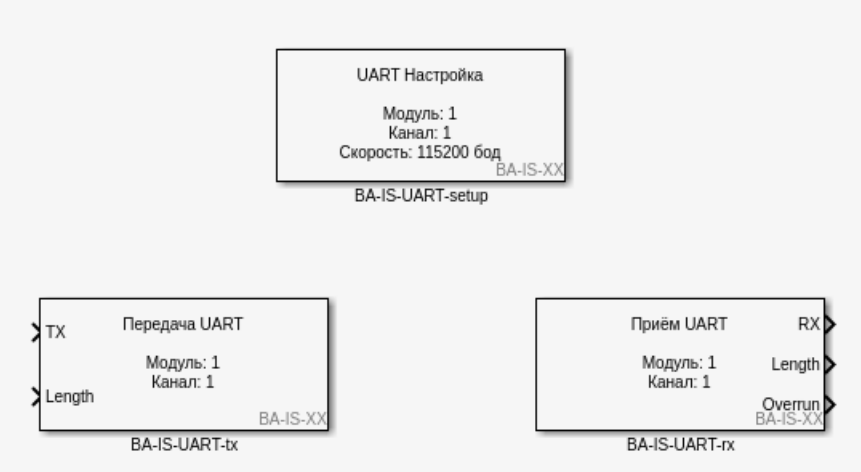
- Описание
- Библиотека блоков
  - Блок Настройка UART
  - Блок Передача UART
  - Блок Приём UART
  - Замечания об использовании и рекомендации

Описание

Настраиваемый модуль приемопередатчика UART базе ПЛИС, поддерживающий до 64 каналов UART в машине реального времени РИТМ.

- Размер буфера на приём и отправку от 16 до 1024 байт (зависит от конфигурации ПЛИС).
- Количество интерфейсов: до 64 портов UART (выбирается при конфигурации ПЛИС).
- Максимальная скорость: 921600 бод.
- Уровень напряжения: 3.3 В LVTTL или 5 В TTL (выбирается при конфигурации ПЛИС).

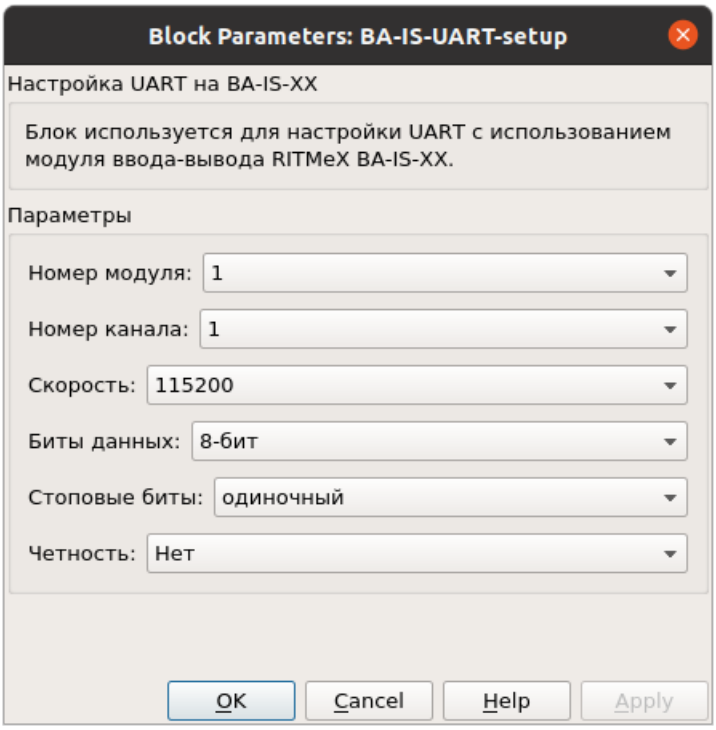
Библиотека блоков



Блок Настройка UART

Блок используется для настройки UART с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



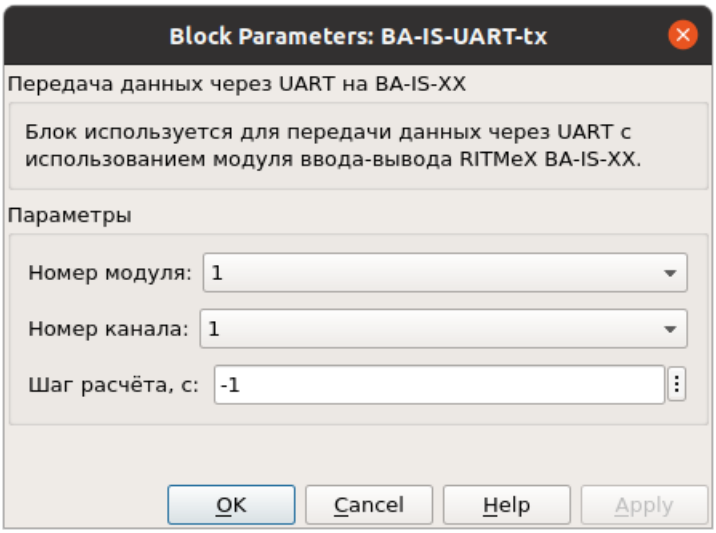
Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Позволяет выбрать канал UART для настройки.
- ***Скорость***: Задает скорость приемопередачи.
- ***Биты данных***: Задает количество битов данных приемопередачи. Возможные значения: *5-бит, 6-бит, 7-бит, 8-бит*.
- ***Стоповые биты***: Задает количество стоповых битов приемопередачи. Возможные значения: *одиночный, двойной*.
- ***Чётность***: Задает метод определения ошибок приемопередачи. Возможные значения: *Нет, Odd, Even*.

Блок Передача UART

Блок используется для передачи данных через UART с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Позволяет выбрать канал UART для передачи.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, по которому происходит передача данных по UART из приложения реального времени.

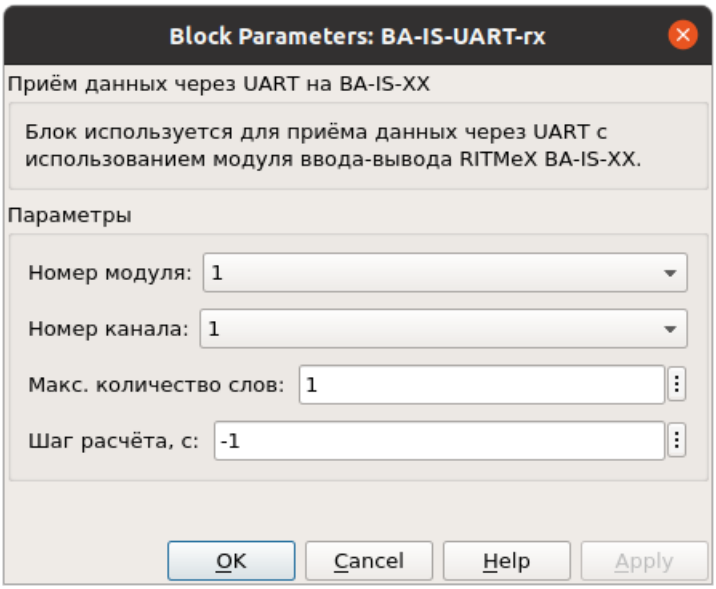
Входы блока

- **TX**: На вход TX блока поступает массив в типе данных uint8 для передачи в линию UART.
- **Length**: На вход Length блока поступает число в типе данных uint32, указывающее, сколько элементов массива TX надо передать в линию UART за шаг расчета.

Блок Приём UART

Блок используется для приёма данных через UART с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Позволяет выбрать канал UART для приёма.
- ***Макс. количество слов***: Позволяет задать количество слов считываемое из буфера UART за шаг расчёта.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, по которому происходит приём данных по UART в приложение реального времени.

Выходы блока

- **RX**: Содержит массив в типе данных uint8, принятый по UART. Размер массива фиксирован и составляет 16. См. также Замечания об использовании.
- **Length**: Содержит информацию о количестве байт, принятых по UART за шаг расчета.
- **Overrun**: Ненулевое значение на этом выходе означает, что данные извне приходят быстрее, чем их вычитывает блок. См. также Замечания об использовании.

Замечания об использовании и рекомендации

Модуль UART на ПЛИС содержит два аппаратных буфера: на приём и отправку. Минимальный размер каждого буфера - 16 байт, максимальный 1024 байт. В связи с этим нужно хорошо понимать характеристики конкретного протокола взаимодействия по UART, чтобы избежать переполнения этих буферов или использовать по максимуму весь буфер. Также гарантировать отправку предыдущего сообщения перед тем, как пытаться отправить следующее, если скорость физического канала выбрана очень низкая.

Первое, что надо учитывать - это скорость передачи данных. Очевидно, что данные по UART не передаются мгновенно. Пусть, например, требуется отправить массив из 16 символов (каждый символ состоит из 10 бит - это 8 бит данных, стартовый и стоповый биты) со скоростью 115200 бод. Для этого потребуется (10/115200)*16=0.0014 секунд. Если блок **Отправка UART** работает с шагом 0.001 с, то данные не успеют отправиться до следующего шага расчета блока, потому что 0.001 < 0.0014. Поэтому следующий

массив символов не может быть отправлен и в такой ситуации выполнение модели будет остановлено с ошибкой. Требуется подобрать такую комбинацию шага расчета блока, скорости передачи и размера массива, чтобы эти данные физически было возможно отправить за один шаг расчета блока.

Второе, что надо учитывать - это размер аппаратного буфера на приём. Если данные извне приходят быстрее, чем блок **Приём UART** считывает их из аппаратного буфера, то новые данные, не умещающиеся в буфер приёма, отбрасываются. Таким образом, надо учитывать и рассчитывать ожидаемое время для заполнения буфера чтения внешним потоком данных.

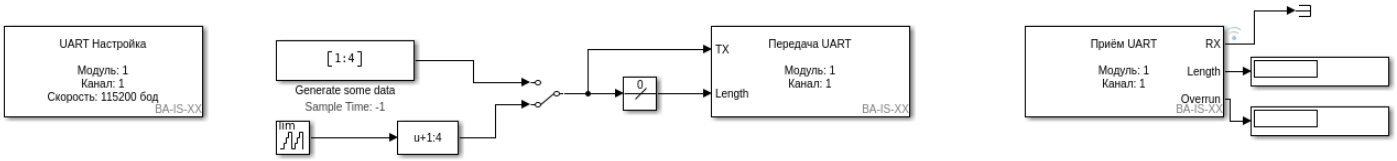
Для этого можно руководствоваться значениями на выходах **Length** и **Overrun** блока **Приём UART**. В расчетах внутри модели следует учитывать ровно столько элементов на выходе RX блока, сколько было фактически получено (выход **Length**) на текущем шаге расчёта. Значения всех остальных элементов массива **RX** не определены.

Если на выходе **Overrun** блока выдается ненулевое значение, то это означает, что аппаратный буфер переполнился и данные в аппаратном буфере приёма были потеряны. В этом случае можно уменьшить шаг расчета блока **Приём UART**, чтобы считывать данные чаще и, таким образом, избегать переполнения буфера приёма. Слишком маленький шаг расчета для блока **Приём UART**, также не следует выставлять, а надо руководствоваться конкретным протоколом обмена. Например, если известно, что внешнее устройство отправляет команды из 8 байт, можно подобрать такой шаг расчета блока, чтобы значение выхода **Length** примерно равнялось 8 от шагу к шагу расчета.

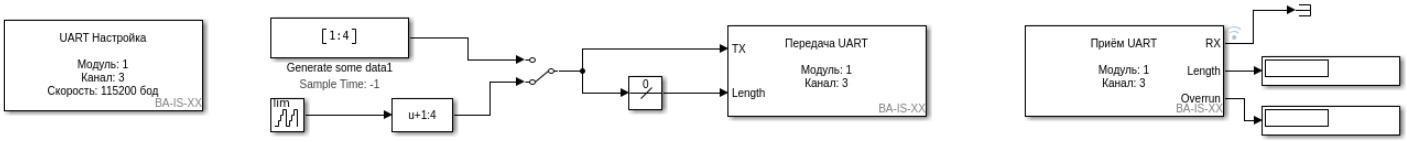
Примеры и демонстрации BA-IS-UART

Пример работы с UART модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_uart.slx')
```



Распиновка указана в
демо-модели,
поставлямой вместе с
прошивкой BA-IS-45



Назначение пинов BA-IS-UART

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль захвата ШИМ через UART

Содержание

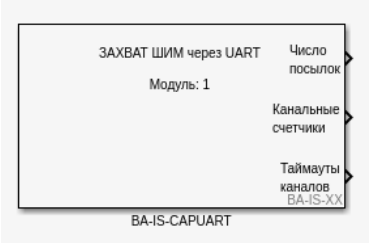
- Описание
- Библиотека блоков
  - Блок Захват ШИМ через UART
  - Детали реализации

Описание

Настраиваемый модуль захвата ШИМ на базе ПЛИС, поддерживающий 18 каналов захвата ШИМ через UART в КПМ РИТМ.

- Количество каналов: 18 каналов захвата ШИМ.
- Скорость обмена по UART: 5 МБод.
- Частота следования пакетов по UART: 100 кГц.

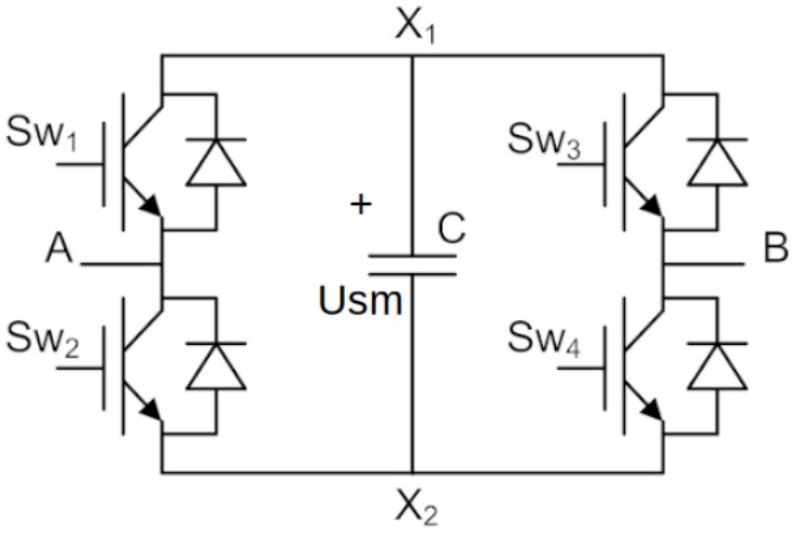
Библиотека блоков



Блок Захват ШИМ через UART

Блок используется для захвата ШИМ сигналов через UART с использованием модуля ввода-вывода RITMeX BA-IS-XX.

Каждый канал UART управляет отдельным силовым блоком. Силовой блок состоит из четырех ключей. Ключи объединены попарно в левый полумост и правый полумост.



Силовые ключи могут находиться в одном из четырех состояний для управления выходным напряжением силового блока:

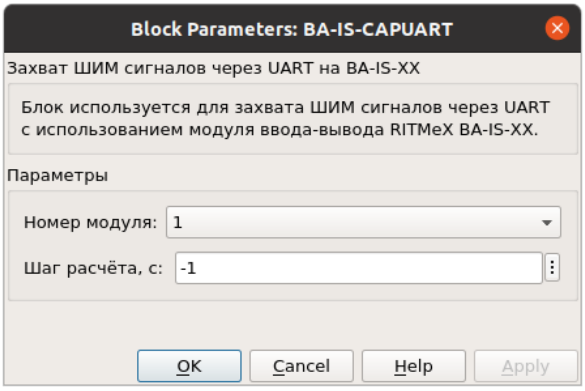
Включенные ключи	Выходное напряжение силового блока	Обозначение состояния
$Sw_1 + Sw_4$	$+U_{sm}$	ON_1
$Sw_2 + Sw_3$	$-U_{sm}$	ON_2
$Sw_1 + Sw_3$	0 (выключен)	OFF_1
$Sw_2 + Sw_4$	0 (выключен)	OFF_2

Модуль, установленный в КПМ РИТМ, получает сигналы через UART от внешней системы управления. Эти сигналы кодируют режимы работы силовых блоков. Формат одной посылки через UART следующий:

Бит	Назначение
0	Общий флаг разрешения работы силового блока (0 – отключить все ключи)
1	Флаг управления левым полумостом (1 – включить верхний ключ *и* отключить нижний; 0 – включить нижний ключ *и* отключить верхний)
2	Флаг управления правым полумостом (1 – включить верхний ключ *и* отключить нижний; 0 – включить нижний ключ *и* отключить верхний)
3	резерв
4	резерв
5	резерв
6	резерв
7	резерв

Задача блока “Захват ШИМ” заключается в том, чтобы принять данные через UART, правильно их интерпретировать (декодировать) и выдать в модель Simulink интегральное состояние ключей с момента предыдущего выполнения блока. На уровне модели Simulink эта информация может использоваться, например, для того, чтобы посчитать действующее значение напряжения и выдать его в модель двигателя в модели Simulink.

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым выдаются значения канальных счетчиков в приложение реального времени.

Выходы блока

- **Число посылок**: Количество посылок, полученных по UART с момента предыдущего расчета блока. Является массивом из 18 элементов в типе данных uint32, которые соответствуют 18 каналам захвата ШИМ.
- **Канальные счетчики**: Интегральный счетчик, показывающий состояние ключей с момента предыдущего расчета блока. В случае получения пакета типа ON_1 счетчик увеличивается, в случае получения пакета типа ON_2 счетчик уменьшается. Является массивом из 18 элементов в типе данных int32, которые соответствуют 18 каналам захвата ШИМ.
- **Таймауты каналов**: Содержит информацию о таймауте получения данных через UART (данные через UART не приходят). Выставляется в 1, если в течение 50 мс по каналу не приходят посылки. Является массивом из 18 элементов в типе данных uint32, которые соответствуют 18 каналам захвата ШИМ.

Детали реализации

В основе блока “Захват ШИМ” находится плата ввода-вывода на базе ПЛИС. Сигналы UART непрерывно захватываются и анализируются в ПЛИС.

ПЛИС поддерживает счетчик по каждому каналу, который позволяет усреднить состояние ключей за последние XXX пакетов (количество пакетов зависит от шага расчета блока Simulink). Счетчик увеличивается, если приходит состояние ON_1. Счетчик уменьшается, если приходит состояние ON_2. Если приходит одно из состояний OFF_1 или OFF_2, то счетчик не меняется. При чтении счетчиков из ПЛИС программным драйвером (блоком Simulink), значения счетчиков по всем 18 каналам выдаются в модель. При этом счетчики внутри ПЛИС сбрасываются, и ПЛИС продолжает получать и обрабатывать (считать) пакеты.

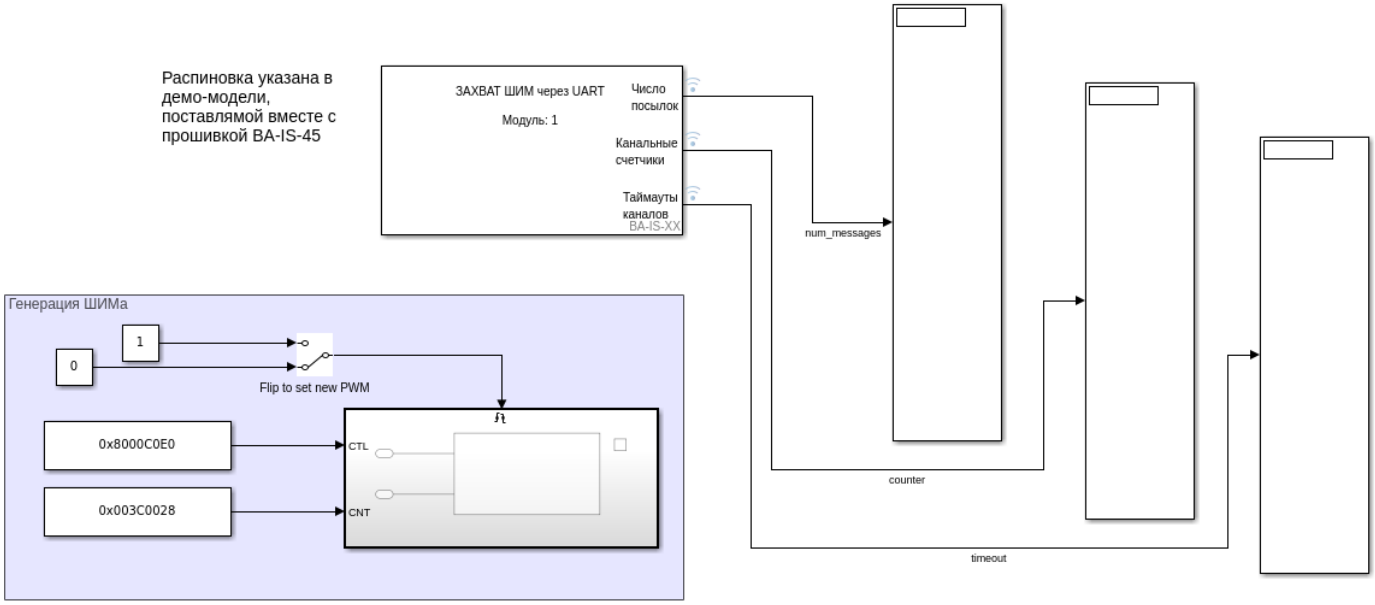
Количеством получаемых пакетов можно управлять, меняя настройку “Шаг расчёта” в блоке Simulink. Например, при шаге расчёта блока 0.001 с и частоте следования пакетов по UART 100 кГц за один шаг расчета блок будет в среднем получать 100 пакетов.

Таким образом, на каждом шаге расчета блока Simulink получается “срез” счетчиков (интеграторов) и количества полученных пакетов по всем каналам (и одновременно эти счетчики сбрасываются внутри ПЛИС).

Примеры и демонстрации BA-IS-CAPAURT

Пример работы с каналами генерации и захвата ШИМ модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_uartpwmcap.slx')
```



Назначение пинов BA-IS-CAPAURT

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модули FMC-RF1 аналоговых входов/выходов

Содержание

- Модуль FMC-RF1 АЦП в коде
- Модуль FMC-RF1 АЦП в вольтах
- Модуль FMC-RF1 DUT АЦП инициализация
- Модуль FMC-RF1 ЦАП одноканальный
- Модуль FMC-RF1 ЦАП двухканальный
- Модуль FMC-RF1 DUT ЦАП инициализация

Модуль FMC-RF1 АЦП в коде

Настраиваемый модуль FMC-RF1 АЦП в коде аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 АЦП в вольтах

Настраиваемый модуль FMC-RF1 АЦП в вольтах аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 DUT АЦП инициализация

Настраиваемый модуль FMC-RF1 DUT инициализация каналов АЦП на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 ЦАП одноканальный

Настраиваемый модуль FMC-RF1 ЦАП одноканальный аналоговых выходов на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 ЦАП двухканальный

Настраиваемый модуль FMC-RF1 ЦАП двухканальный аналоговых выходов на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 DUT ЦАП инициализация

Настраиваемый модуль FMC-RF1 DUT инициализация каналов ЦАП на базе ПЛИС для машины реального времени РИТМ.

Модуль FMC-RF1 АЦП в коде

Содержание

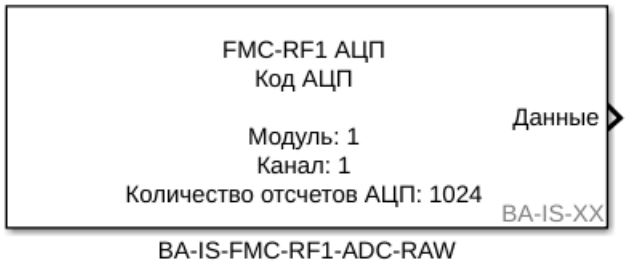
- Описание
- Библиотека блоков

Описание

Настраиваемый модуль FMC-RF1 АЦП в коде аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

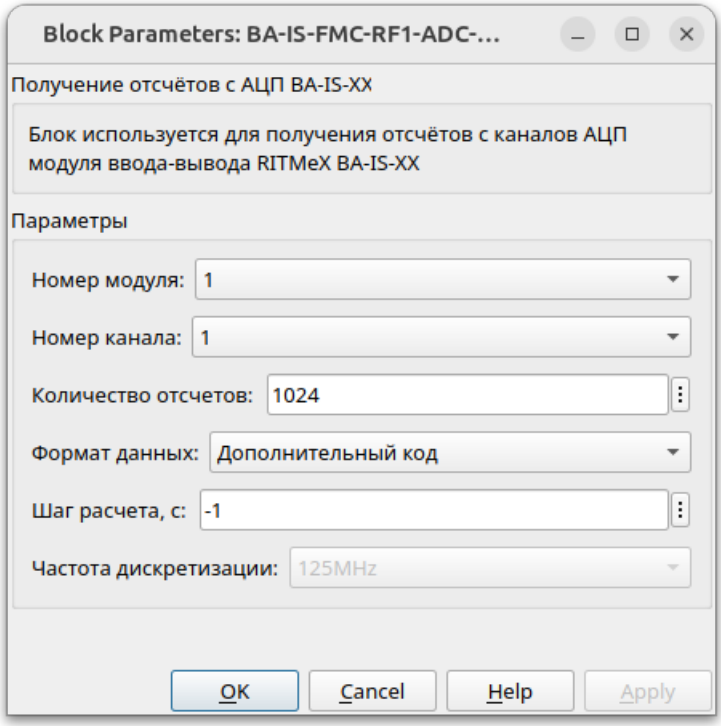
- 2 независимых входа;
- Разрядность АЦП 16 бит;
- Частота дискретизации 125 МГц.

Библиотека блоков



BA-IS-FMC-RF1-ADC-RAW

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого

же типа.

- ***Номер канала***: Задаёт номер каналов аналогового входа.
- ***Количество отсчётов***: Задаёт количество отсчётов для выхода.
- ***Формат данных***: Формат выдачи отсчётов.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит получение данных с аналоговых входов.
- ***Частота дискретизации***: Частота дискретизации, не активная, всегда 125 МГц.

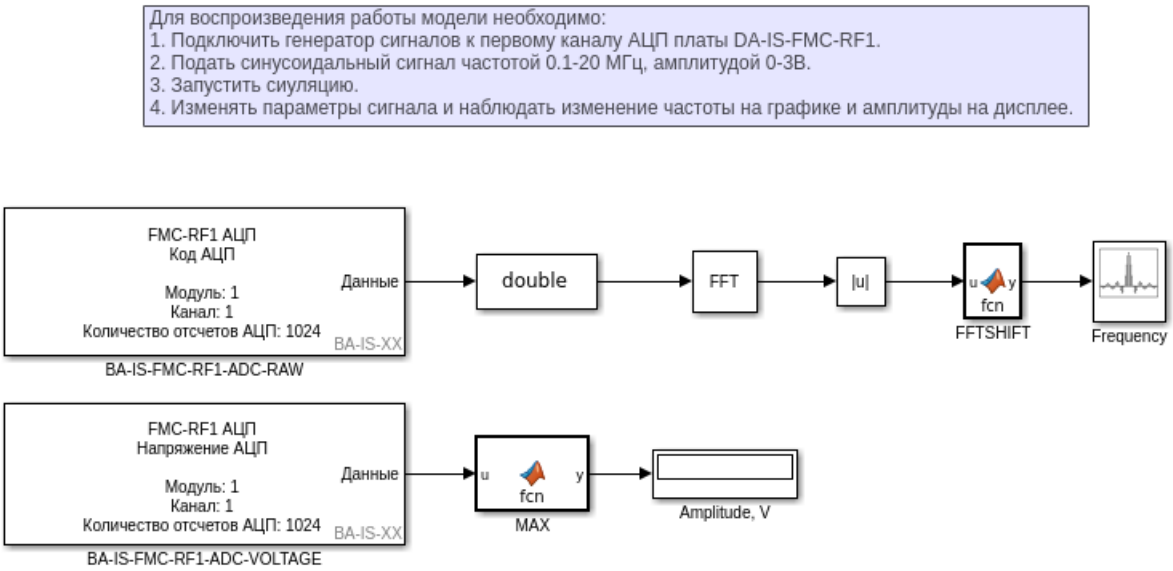
Выходы блока

- **Данные**: Данные, принятые с аналоговых входов. Тип данных *int16*, массив в зависимости от значения параметра *Количество отчётов*.

Примеры и демонстрации FMC-RF1 АЦП

Пример работы с АЦП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_rf1_adc.slx')
```



Назначение пинов FMC-RF1 АЦП

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль FMC-RF1 АЦП в вольтах

Содержание

- Описание
- Библиотека блоков

Описание

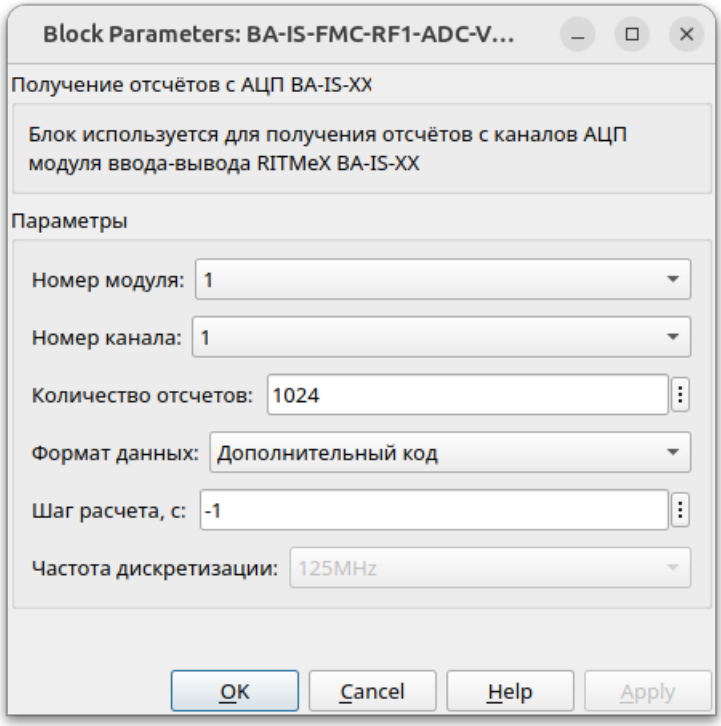
Настраиваемый модуль FMC-RF1 АЦП в вольтах аналоговых входов на базе ПЛИС для машины реального времени РИТМ.

- 2 независимых входа;
- Разрядность АЦП 16 бит;
- Частота дискретизации 125 МГц.

Библиотека блоков

BA-IS-FMC-RF1-ADC-VOLTAGE

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого

же типа.

- ***Номер канала***: Задаёт номер каналов аналогового входа.
- ***Количество отсчётов***: Задаёт количество отсчётов для выхода.
- ***Формат данных***: Формат выдачи отсчётов.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит получение данных с аналоговых входов.
- ***Частота дискретизации***: Частота дискретизации, не активная, всегда 125 МГц.

Выходы блока

- **Данные**: Данные, принятые с аналоговых входов. Тип данных *double*, массив в зависимости от значения параметра *Количество отчётов*.

Модуль FMC-RF1 DUT АЦП инициализация

Содержание

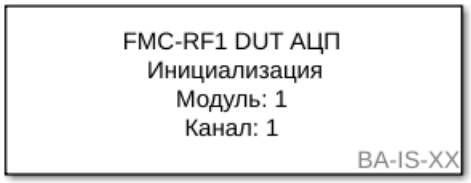
- Описание
- Библиотека блоков

Описание

Настраиваемый модуль FMC-RF1 DUT инициализация каналов АЦП на базе ПЛИС для машины реального времени РИТМ.

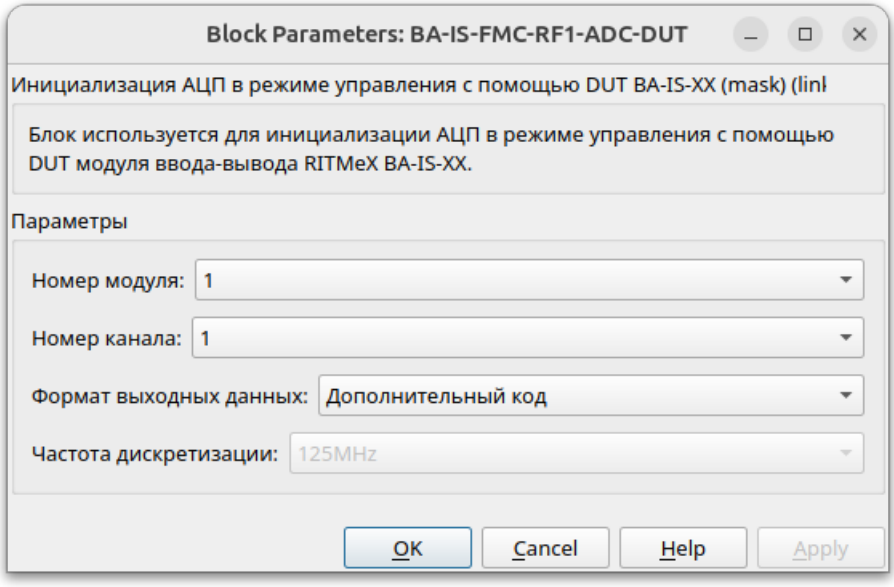
- 2 независимых входа;
- Разрядность АЦП 16 бит;
- Частота дискретизации 125 МГц.

Библиотека блоков



BA-IS-FMC-RF1-ADC-DUT

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Номер канала***: Задаёт номер каналов аналогового входа.
- ***Формат данных***: Формат выдачи отсчётов.
- ***Частота дискретизации***: Частота дискретизации, не активная, всегда 125 МГц.

Модуль FMC-RF1 ЦАП одноканальный

Содержание

- Описание
- Библиотека блоков

Описание

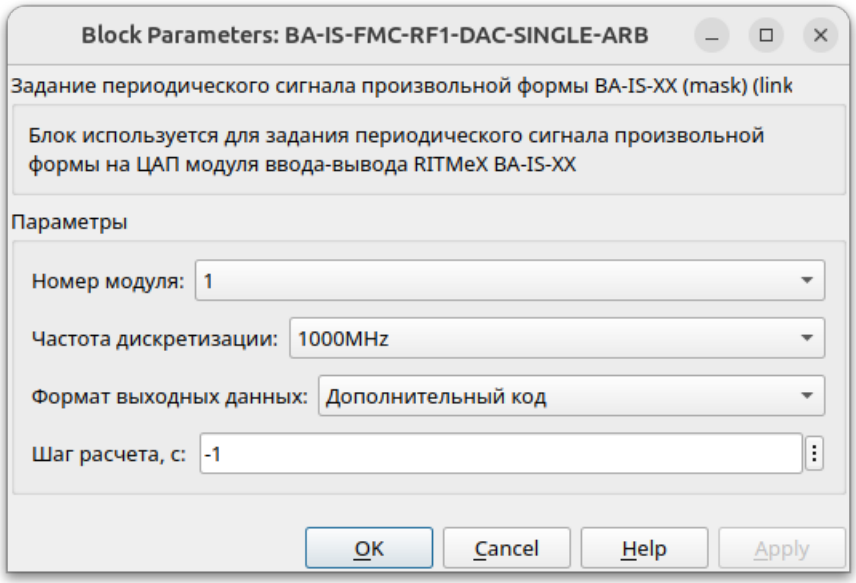
Настраиваемый модуль FMC-RF1 ЦАП одноканальный аналоговых выходов на базе ПЛИС для машины реального времени РИТМ.

- Разрядность ЦАП 16 бит;
- Частота дискретизации 125/500/1000 МГц.

Библиотека блоков

BA-IS-FMC-RF1-DAC-SINGLE-ARB

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Частота дискретизации***: Частота дискретизации в МГц.
- ***Количество каналов***: Задает количество каналов аналогового выхода.
- ***Формат выходных данных***: Формат данных для ЦАП.

- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

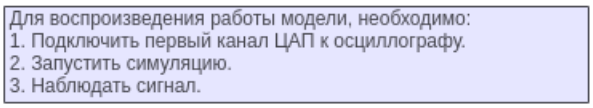
Входы блока

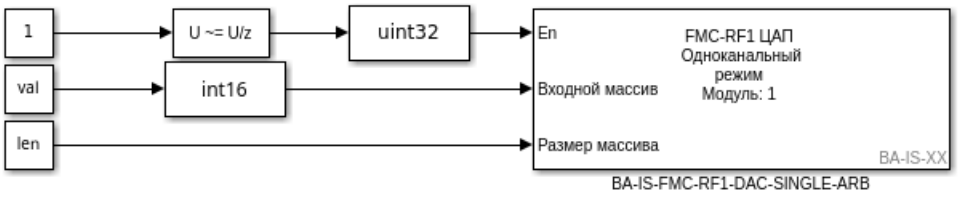
- **EN**: Вход включения блока. Тип данных *uint32*.
- **Входной массив**: Данные для отправки на аналоговые выходы. Тип данных *int16*, может быть массивом в зависимости от значения входа *Размер массива*.
- **Размер массива**: Размер входного массива. Тип данных *uint32*.

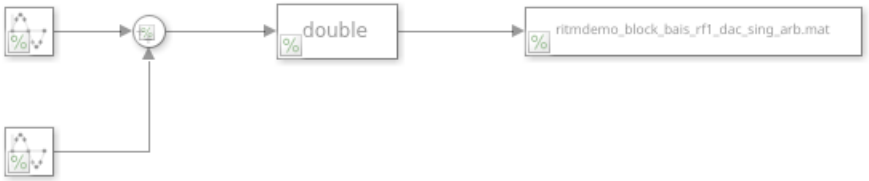
Примеры и демонстрации FMC-RF1 ЦАП одноканальный

Пример работы с каналами ЦАП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_rf1_dac_arb.slx')
```







Назначение пинов FMC-RF1 ЦАП

Модуль BA-IS-XX поставляется с клеммной платой.

Распиновка зависит от конкретной прошивки ПЛИС. Распиновка указана в демо-модели, поставляемой вместе с прошивкой BA-IS-XX.

Модуль FMC-RF1 ЦАП двухканальный

Содержание

- Описание
- Библиотека блоков

Описание

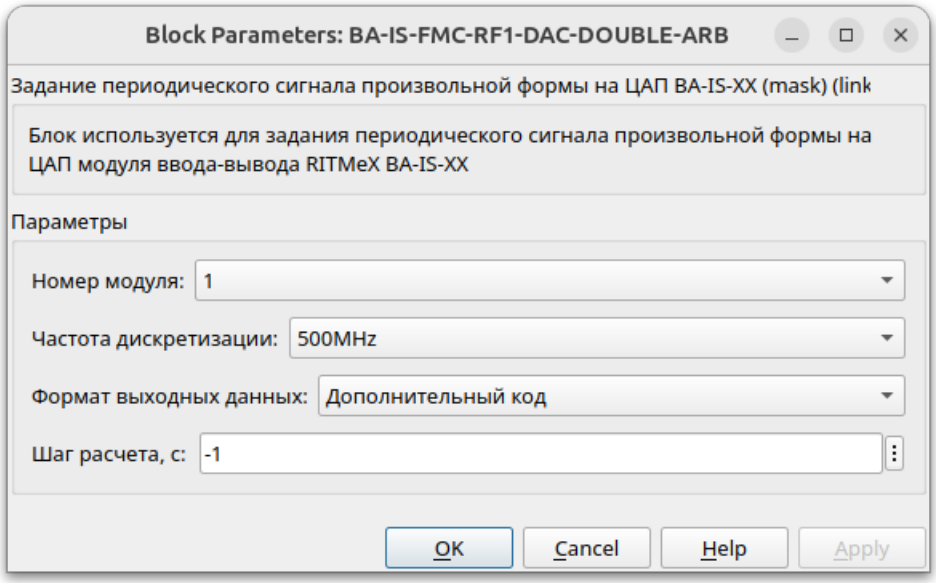
Настраиваемый модуль FMC-RF1 ЦАП двухканальный аналоговых выходов на базе ПЛИС для машины реального времени РИТМ.

- Разрядность ЦАП 16 бит;
- Частота дискретизации 62,5/250/500 МГц.

Библиотека блоков

BA-IS-FMC-RF1-DAC-DOUBLE-ARB

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Частота дискретизации***: Частота дискретизации в МГц.
- ***Количество каналов***: Задает количество каналов аналогового выхода.

- ***Формат выходных данных***: Формат данных для ЦАП.
- ***Шаг расчёта***: Позволяет задать шаг расчёта в секундах, с которым происходит выдача отсчётов на аналоговые выходы.

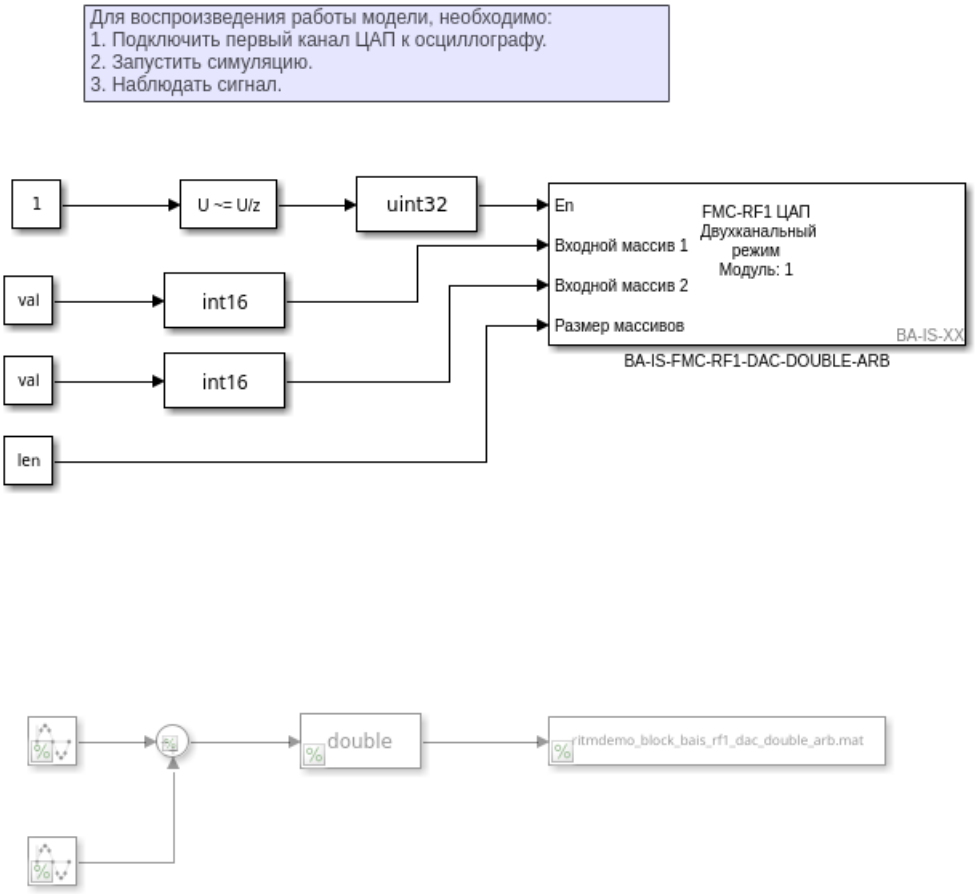
Входы блока

- **EN**: Вход включения блока. Тип данных *uint32*.
- **Входной массив 1**: Данные для отправки на аналоговые выходы канал 1. Тип данных *int16*, может быть массивом в зависимости от значения входа *Размер массива*.
- - **Входной массив 2**: Данные для отправки на аналоговые выходы канал 2. Тип данных *int16*, может быть массивом в зависимости от значения входа *Размер массива*.
- **Размер массива**: Размер входного массива. Тип данных *uint32*.

Примеры и демонстрации FMC-RF1 ЦАП двухканальный

Пример работы с каналами ЦАП модуля BA-IS-XX с использованием поставляемой клеммной платы.

```
open_system('ritmdemo_block_bais_rf1_double_dac_arb.slx')
```



Модуль FMC-RF1 DUT ЦАП инициализация

Содержание

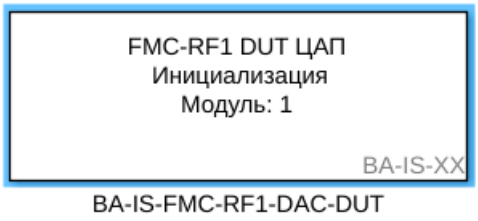
- Описание
- Библиотека блоков

Описание

Настраиваемый модуль FMC-RF1 DUT инициализация каналов ЦАП на базе ПЛИС для машины реального времени РИТМ.

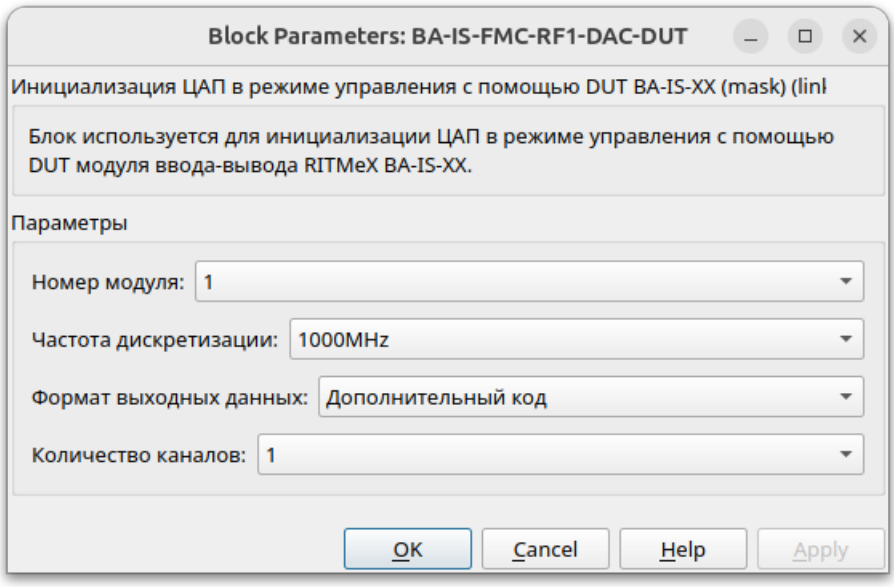
- Разрядность ЦАП 16 бит;
- Частота дискретизации в одноканальном режиме 125/500/1000 МГц;
- Частота дискретизации в двухканальном режиме 62.5/250/500 МГц.

Библиотека блоков



BA-IS-FMC-RF1-DAC-DUT

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Частота дискретизации***: Частота дискретизации в МГц.
- ***Формат выходных данных***: Формат данных для ЦАП.
- ***Количество каналов***: Задает количество каналов аналогового выхода.

Настройка модуля ввода-вывода BA-IS-XX

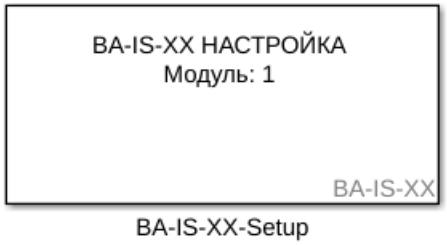
Содержание

- Описание
- Библиотека блока
- Маска блока
- Параметры блока

Описание

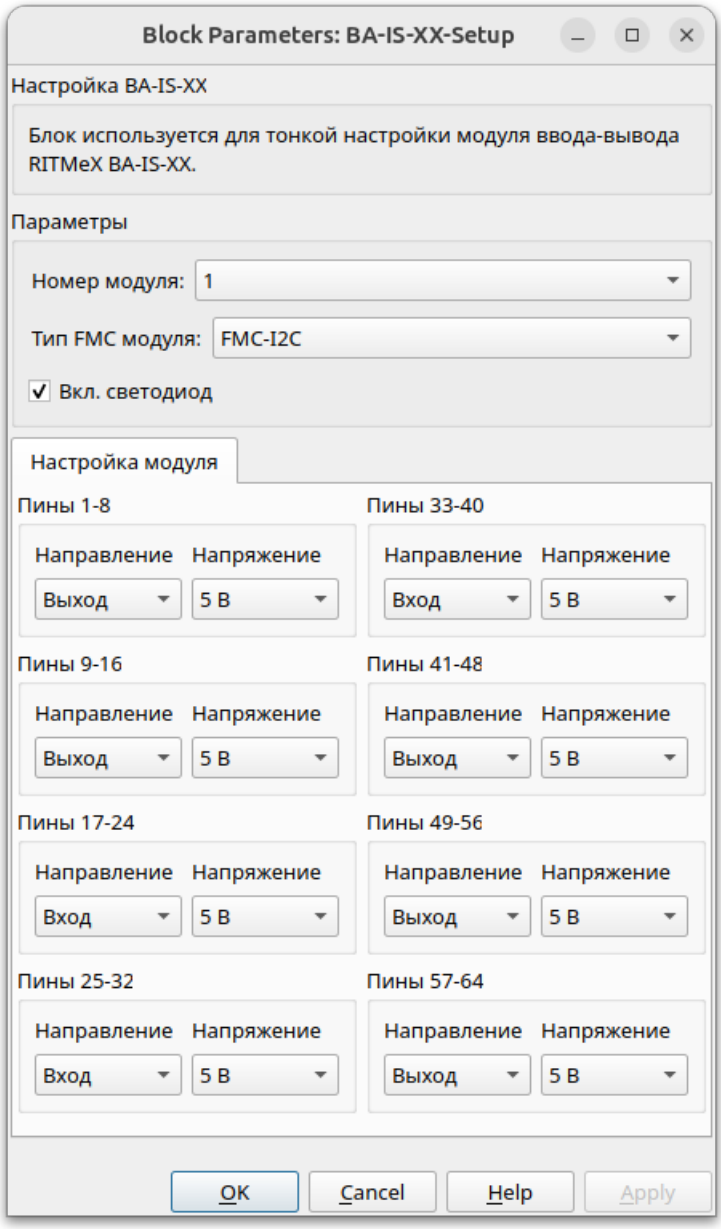
Блок используется для тонкой настройки модуля ввода-вывода RITMeX BA-IS-XX.

Библиотека блока



BA-IS-XX-Setup

Маска блока



Параметры блока

- ***Номер модуля***: Используется для уникальной идентификации модуля, когда в машине реального времени установлено больше одного модуля ввода-вывода этого же типа.
- ***Тип FMC модуля***: Используется для уникальной идентификации FMC модуля.
- ***Вкл. светодиод***: Позволяет включить или выключить светодиод на модуле.
- ***Пины 1-64 Направление Напряжение***: Позволяет задать направление пинов Вход или Выход и уровень сигналов 3.3 В или 5 В.

Электроэнергетические системы

Содержание

- Блоки поддержки IEC 61850
- Блоки поддержки PRP
- Блоки поддержки C37.118
- Блоки для моделирования электрических машин
- Вспомогательные функции

Блоки поддержки IEC 61850

Библиотека блоков для отправки и приёма сообщений по протоколам IEC 61850.

Блоки поддержки PRP

Библиотека блоков для формирования кадров с RCT и отбрасывания дубликатов PRP-кадров в соответствии с IEC 62439-3-2016.

Блоки поддержки C37.118

Библиотека блоков для отправки и приема синхронизированных векторных измерений.

Блоки для моделирования электрических машин

В данном разделе приводится информация о дополнительных блоках для физического моделирования электрических машин средствами Simscape Electrical.

Вспомогательные функции

В данном разделе приводится информация о различных вспомогательных функциях.

Блоки поддержки IEC 61850

Содержание

- Библиотека блоков для работы с SV по IEC 61850
- Библиотека блоков для работы с GOOSE по IEC 61850

Библиотека блоков для работы с SV по IEC 61850

Библиотека блоков для отправки и приёма сообщений по протоколу SV IEC 61850.

Библиотека блоков для работы с GOOSE по IEC 61850

Библиотека блоков для отправки и приёма сообщений по протоколу GOOSE IEC 61850.

Библиотека блоков для работы с SV по IEC 61850

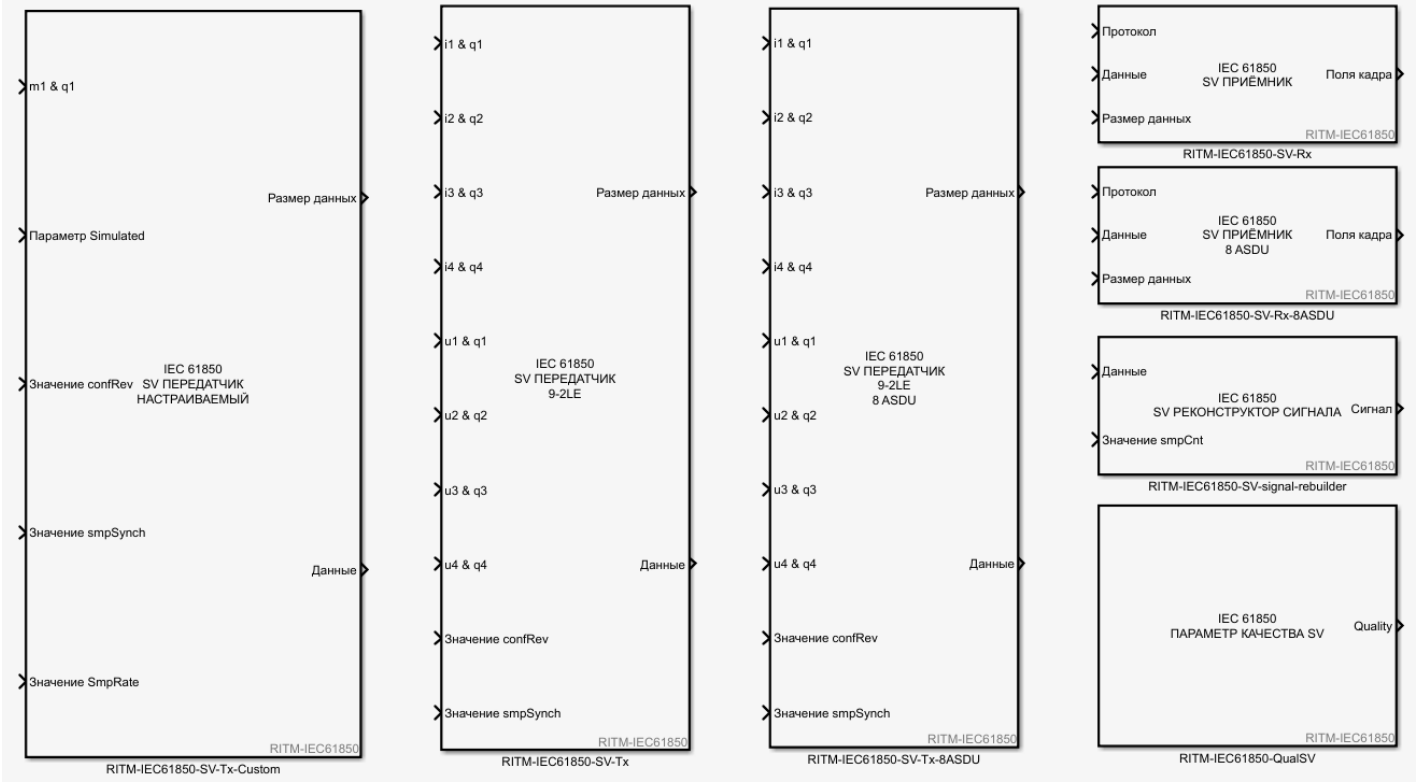
Содержание

- Описание
- Библиотека блоков
- Блок IEC 61850 SV Передатчик
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок IEC 61850 SV Передатчик 8 ASDU
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок IEC 61850 SV Передатчик настраиваемый
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок IEC 61850 SV Приёмник
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выход блока
- Блок IEC 61850 SV Приёмник 8 ASDU
  - Маска блока
  - Параметры блока
  - Входы блока
  - Выход блока
- Блок IEC 61850 SV Реконструктор сигнала
  - Маска блока
  - Входы блока
  - Выход блока
  - Параметры блока
- Блок IEC 61850 Параметр качества SV
  - Маска блока
  - Входы блока
  - Выход блока
  - Параметры блока

Описание

Библиотека блоков для отправки и приёма SV сообщений по Ethernet.
Перед использованием необходимо произвести настройку Ethernet портов КПМ РИТМ, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

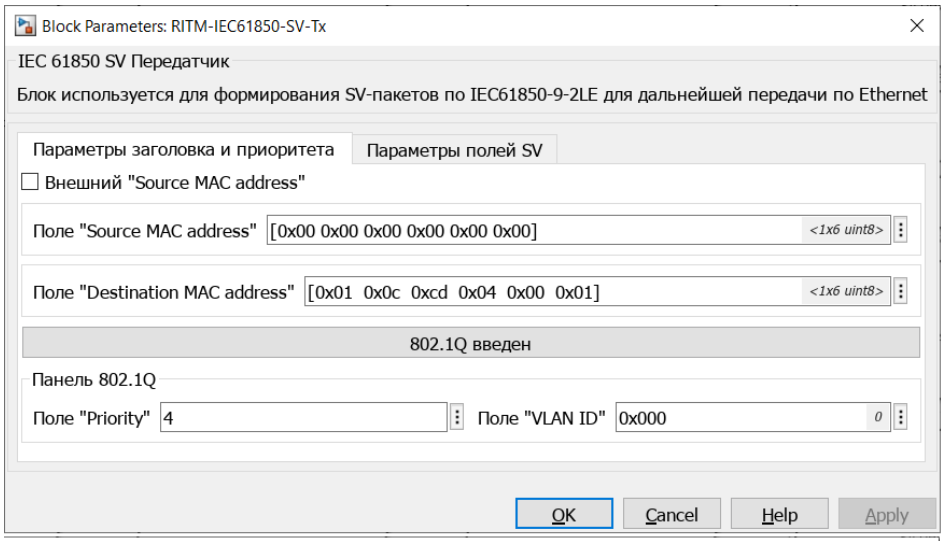
Библиотека блоков

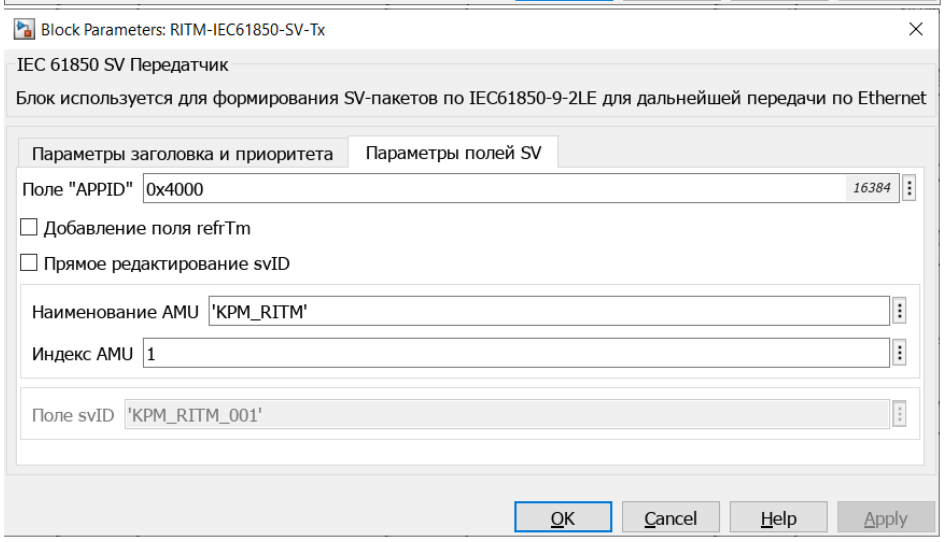


Блок IEC 61850 SV Передатчик

Блок используется для формирования SV-кадров из набора полей по IEC61850-9-2LE для дальнейшей передачи по Ethernet.

Маска блока





Входы блока

- **i1 & q1 - u4 & q4**: Вектор из 2 элементов, содержащий значение сигнала тока или напряжения, а также параметр качества этого сигнала от блока IEC 61850 Параметр качества SV, составленный при помощи блока Mux, представленного в библиотеке Simulink.
- **Значение confRev**: Версия ревизии конфигурации, тип данных *uint32*.
- **Значение smpSynch**: Статус синхронизации, тип данных *uint8*.
- **MAC отправителя**: Значение MAC-адреса в форме массива из 6 байт, вход используется в случае включения опции **Внешний “Source MAC address”** в параметрах блока.
- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.

Выходы блока

- **Размер данных**: Размер SV-пакета, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные**: Сформированный SV-пакет, как набор байт, тип данных *uint8*.

Параметры блока

Панель “Параметры заголовка и приоритета”

- **Внешний “Source MAC address”**: Позволяет включить/отключить возможность задать MAC-адрес через вход **MAC отправителя** блока.
- **Поле “Source MAC address”**: Массив из 6 байт - MAC-адрес источника передачи SV-потока.
- **Поле “Destination MAC address”**: Массив из 6 байт - MAC-адрес приемника SV-потока, для Multicast рассылки потокам присваиваются MAC-адреса в диапазоне 01:0C:CD:04:00:00 - 01:0C:CD:04:01:FF.
- **802.1Q введено/выведено**: Кнопка включения/отключения формирования поля VLAN.
- **Поле “Priority”**: Позволяет задать тег приоритета передачи трафика, формат *uint8*.
- **Поле “VLAN ID”**: Идентификатор виртуальной сети, которой принадлежит поток (12 бит), тип данных *uint16*.

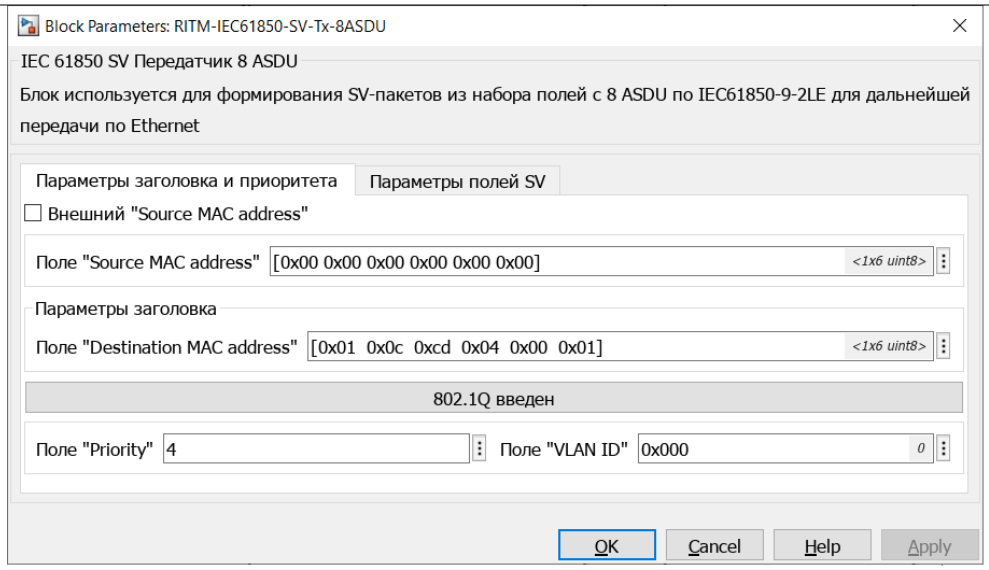
Панель “Параметры полей SV”

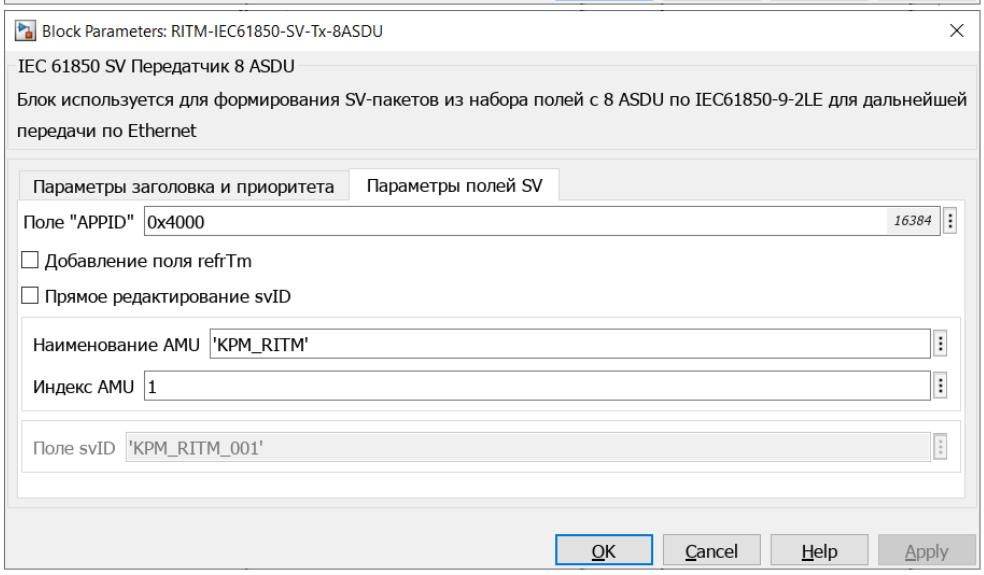
- **Поле “APPID”**: Позволяет задать значение идентификационной метки сообщения (стандарт предусматривает диапазон 0x4000 - 0x7FFF, однако возможно задать и иное значение).
- **Добавление поля refrTm**: Добавляет в SV-кадр поле refrTm.
- **Прямое редактирование svID**: Позволяет переключаться между видами задания поля svID в SV-кадре: через поля **Наименование AMU** и **Индекс AMU** или поле **Поле “svID”**.
- **Наименование AMU**: Позволяет задать имя AMU, тип данных *string*.
- **Индекс AMU**: Позволяет задать номер AMU, тип данных *uint8*.
- **Поле “svID”**: Позволяет задать полное наименование AMU, тип данных *string*.

Блок IEC 61850 SV Передатчик 8 ASDU

Блок используется для формирования SV-кадров из набора полей с **8 ASDU** по IEC61850-9-2LE для дальнейшей передачи по Ethernet.

Маска блока





Входы блока

- **i1 & q1 - u4 & q4**: Вектор из 2 элементов, содержащий значение сигнала тока или напряжения, а также параметр качества этого сигнала от блока IEC 61850 Параметр качества SV, составленный при помощи блока Mux, представленного в библиотеке Simulink.
- **Значение confRev**: Версия ревизии конфигурации, тип данных *uint32*.
- **Значение smpSynch**: Статус синхронизации, тип данных *uint8*.
- **MAC отправителя**: Значение MAC-адреса в форме массива из 6 байт, вход используется в случае включения опции **Внешний “Source MAC address”** в параметрах блока.
- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.

Выходы блока

- **Размер данных**: Размер SV-пакета, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные**: Сформированный SV-пакет, как набор байт, тип данных *uint8*.

Параметры блока

Панель “Параметры заголовка и приоритета”

- **Внешний “Source MAC address”**: Позволяет включить/отключить возможность задать MAC-адрес через вход **MAC отправителя** блока.
- **Поле “Source MAC address”**: Массив из 6 байт - MAC-адрес источника передачи SV-потока.
- **Поле “Destination MAC address”**: Массив из 6 байт - MAC-адрес приемника SV-потока, для Multicast рассылки потокам присваиваются MAC-адреса в диапазоне 01:0C:CD:04:00:00 - 01:0C:CD:04:01:FF.
- **802.1Q введено/выведено**: Кнопка включения/отключения формирования поля VLAN.
- **Поле “Priority”**: Позволяет задать тег приоритета передачи трафика, формат *uint8*.
- **Поле “VLAN ID”**: Идентификатор виртуальной сети, которой принадлежит поток (12 бит), тип данных *uint16*.

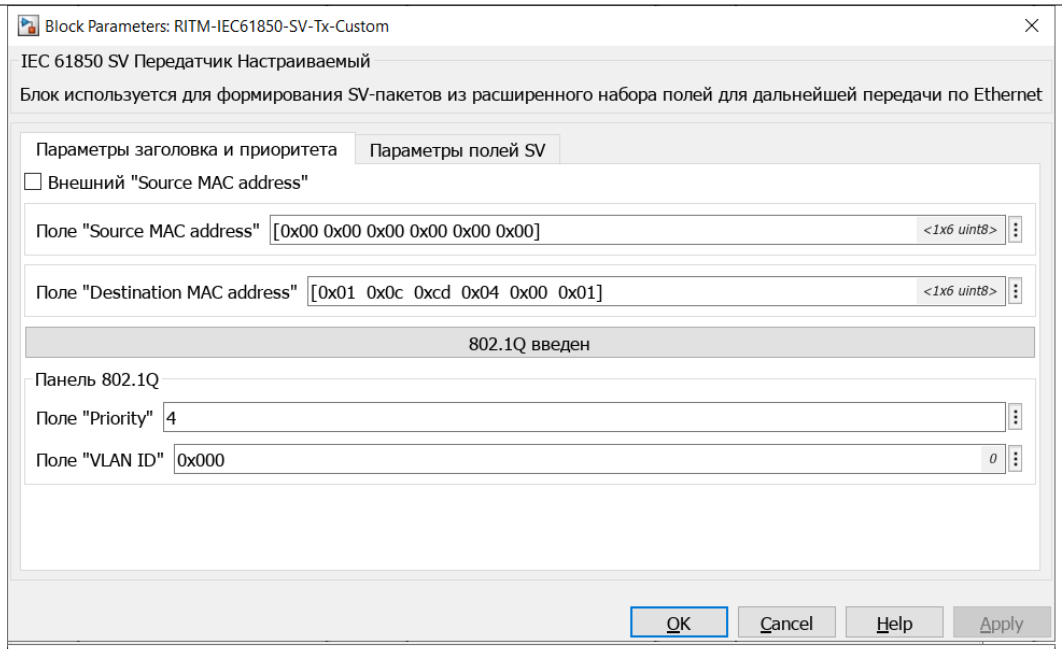
Панель “Параметры полей SV”

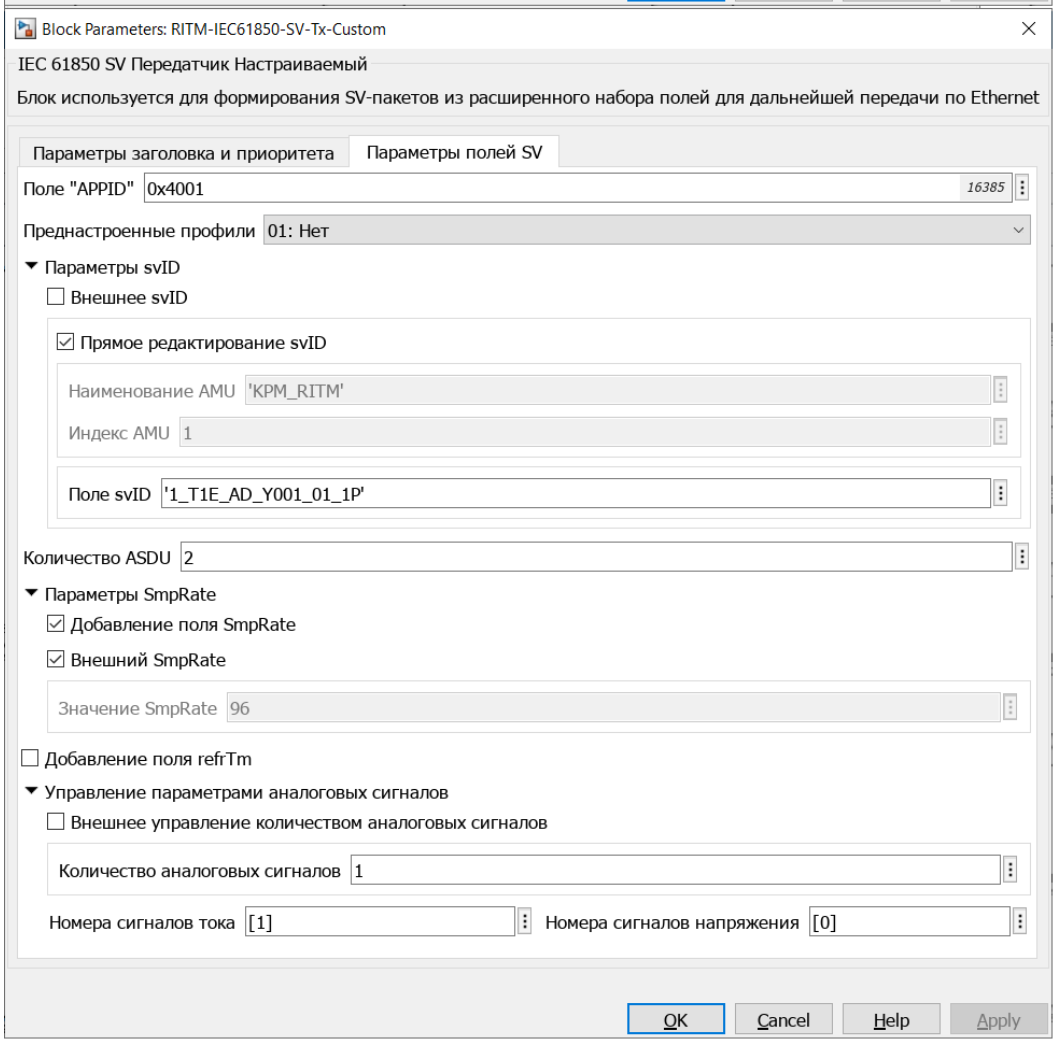
- **Поле “APPID”**: Позволяет задать значение идентификационной метки сообщения (стандарт предусматривает диапазон 0x4000 - 0x7FFF, однако возможно задать и иное значение).
- **Добавление поля refrTm**: Добавляет в SV-кадр поле refrTm.
- **Прямое редактирование svID**: Позволяет переключаться между видами задания поля svID в SV-кадре: через поля **Наименование AMU** и **Индекс AMU** или поле **Поле “svID”**.
- **Наименование AMU**: Позволяет задать имя AMU, тип данных *string*.
- **Индекс AMU**: Позволяет задать номер AMU, тип данных *uint8*.
- **Поле “svID”**: Позволяет задать полное наименование AMU, тип данных *string*.

Блок IEC 61850 SV Передатчик настраиваемый

Блок используется для формирования SV-кадров из расширенного набора полей для дальнейшей передачи по Ethernet.

Маска блока





Входы блока

- **m1 & q1 - m8 & q8**: Вектор из 2 элементов, содержащий значение сигнала тока или напряжения, а также параметр качества этого сигнала от блока IEC 61850 Параметр качества SV, составленный при помощи блока Mux, представленного в библиотеке Simulink.
- **Параметр Simulated**: Значение старшего бита в поле Reserved 1 для перевода потока в режим тестирования, тип данных *uint8*.
- **Значение confRev**: Версия ревизии конфигурации, тип данных *uint32*.
- **Значение smpSynch**: Статус синхронизации, тип данных *uint8*.
- **Количество входных аналогов**: количество аналоговых сигналов для передаче в SV-потоке, тип данных *uint8*.
- **Значение smpRate**: Значение скорости формирования пакетов, тип данных *uint32*.
- **Значение “svID”**: Значение поля идентификатора мгновенных значений, тип данных *uint8*.
- **MAC отправителя**: Значение MAC-адреса в форме массива из 6 байт, вход используется в случае включения опции **Внешний**

“Source MAC address” в параметрах блока.

- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*, вход используется в случае включения опции **Добавление поля refrTm** в параметрах блока.

Выходы блока

- **Размер данных**: Размер SV-пакета, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные**: Сформированный SV-пакет, как набор байт, тип данных *uint8*.

Параметры блока

Панель “Параметры заголовка и приоритета”

- **Внешний “Source MAC address”**: Позволяет включить/отключить возможность задать MAC-адрес через вход **MAC отправителя** блока.
- **Поле “Source MAC address”**: Массив из 6 байт - MAC-адрес источника передачи SV-потока.
- **Поле “Destination MAC address”**: Массив из 6 байт - MAC-адрес приемника SV-потока, для Multicast рассылки потокам присваиваются MAC-адреса в диапазоне 01:0C:CD:04:00:00 - 01:0C:CD:04:01:FF.
- **802.1Q введено/выведено**: Кнопка включения/отключения формирования поля VLAN.
- **Поле “Priority”**: Позволяет задать тег приоритета передачи трафика, формат *uint8*.
- **Поле “VLAN ID”**: Идентификатор виртуальной сети, которой принадлежит поток (12 бит), тип данных *uint16*.

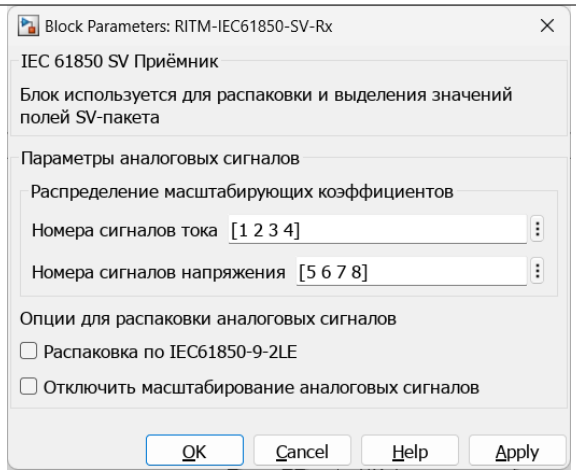
Панель “Параметры полей SV”

- **Поле “APPID”**: Позволяет задать значение идентификационной метки сообщения (стандарт предусматривает диапазон 0x4000 - 0x7FFF, однако возможно задать и иное значение).
- **Преднастроенные профили**: Выпадающий список, позволяющий выбрать заранее настроенные параметры блока для передачи по профилям организаций (например, профиль ФСК ЕЭС) или активировать поля для возможности ввода собственных настроек.
- **Параметры svID**: Выпадающий список с настройками svID.
- **Внешнее svID**: Позволяет задавать значение поля svID через вход **Значение “svID”** блока.
- **Прямое редактирование svID**: Позволяет переключаться между видами задания поля svID в SV-кадре: через поля **Наименование AMU** и **Индекс AMU** или поле **Поле “svID”**.
- **Наименование AMU**: Позволяет задать имя AMU, тип данных *string*.
- **Индекс AMU**: Позволяет задать номер AMU, тип данных *uint8*.
- **Поле “svID”**: Позволяет задать полное наименование AMU, тип данных *string*.
- **Количество ASDU**: Поле для определения количества блоков ASDU в SV-кадр.
- **Параметры SmpRate**: Выпадающий список с настройками SmpRate.
- **Добавление поля SmpRate**: Добавляет в SV-кадр поле SmpRate.
- **Внешний SmpRate**: Позволяет задавать значение SmpRate через вход **Значение SmpRate** блока.
- **Значение SmpRate**: Поле позволяет задать количество выборок аналоговых сигналов на период промышленной частоты SmpRate, тип данных *uint32*.
- **Добавление поля refrTm**: Добавляет в SV-кадр поле refrTm.
- **Внешнее управление количеством аналоговых сигналов**: Включает режим динамического изменения количества аналоговых сигналов в SV-пакете через вход **Количество входных аналогов** блока, обратите внимание, что в данном случае данные всех аналоговых сигналов и параметров качества следует подавать на вход m1 & q1 в виде вектора.
- **Количество аналоговых сигналов**: Позволяет задать количество аналоговых сигналов на ASDU, тип данных *uint8*.
- **Номера сигналов тока**: Вектор, содержащий порядковые номера аналоговых сигналов тока в ASDU, тип данных *uint8*.
- **Номера сигналов напряжения**: Вектор, содержащий порядковые номера аналоговых сигналов напряжения в ASDU, тип данных *uint8*.

Блок IEC 61850 SV Приёмник

Блок используется для распаковки и выделения значений полей SV-пакета.

Маска блока



Параметры блока

Параметры аналоговых сигналов

Распределение масштабирующих коэффициентов

- **Номера сигналов тока**: Вектор, содержащий порядковые номера аналоговых сигналов тока в ASDU для определения масштабирующих коэффициентов.
- **Номера сигналов напряжения**: Вектор, содержащий порядковые номера аналоговых сигналов напряжения в ASDU для определения масштабирующих коэффициентов.

Опции для распаковки аналоговых сигналов

- **Распаковка по IEC61850-9-2LE**: Блок выставляет масштабирующие коэффициенты и распаковывает SV-кадр по IEC61850-9-2LE.
- **Отключить масштабирование аналоговых сигналов**: Блок отключает масштабирование аналоговых сигналов в ходе распаковки кадра.

Входы блока

- **Протокол**: Значение поля метки протокола, которую содержит пакет, полученное с блока Raw Ethernet Распаковка пакета.
- **Данные**: Сегмент данных пакета, полученный с блока Raw Ethernet Распаковка пакета.
- **Размер данных**: Размер сегмента данных, полученный с блока Raw Ethernet Распаковка пакета.

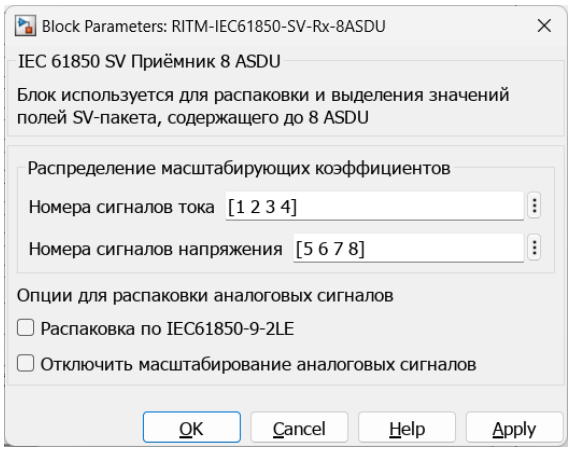
Выход блока

- **Поля кадра**: Шина, содержащая все поля SV-кадра. Для выделения значения определенного поля или набора полей необходимо воспользоваться блоком **Bus Selector** библиотеки блоков Simulink.

Блок IEC 61850 SV Приёмник 8 ASDU

Блок используется для распаковки и выделения значений полей SV-пакета, содержащего до **8 ASDU**.

Маска блока



Параметры блока

Параметры аналоговых сигналов

Распределение масштабирующих коэффициентов

- **Номера сигналов тока**: Вектор, содержащий порядковые номера аналоговых сигналов тока в ASDU для определения масштабирующих коэффициентов.
- **Номера сигналов напряжения**: Вектор, содержащий порядковые номера аналоговых сигналов напряжения в ASDU для определения масштабирующих коэффициентов.

Опции для распаковки аналоговых сигналов

- **Распаковка по IEC61850-9-2LE**: Блок выставляет масштабирующие коэффициенты и распаковывает SV-кадр по IEC61850-9-2LE.
- **Отключить масштабирование аналоговых сигналов**: Блок отключает масштабирование аналоговых сигналов в ходе распаковки кадра.

Входы блока

- **Протокол**: Значение поля метки протокола, которую содержит пакет, полученное с блока Raw Ethernet Распаковка пакета.
- **Данные**: Сегмент данных пакета, полученный с блока Raw Ethernet Распаковка пакета.
- **Размер данных**: Размер сегмента данных, полученный с блока Raw Ethernet Распаковка пакета.

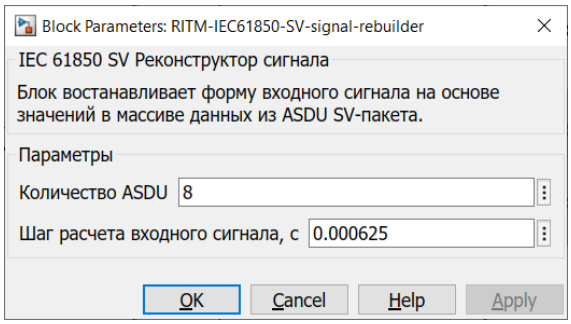
Выход блока

- **Поля кадра**: Шина, содержащая все поля SV-кадра. Для выделения значения определенного поля или набора полей необходимо воспользоваться блоком **Bus Selector** библиотеки блоков Simulink.

Блок IEC 61850 SV Реконструктор сигнала

Блок используется для восстановления формы сигнала тока и напряжения в случае передачи больше одного ASDU в SV-кадре.

Маска блока



Входы блока

- **Данные**: Значения токов и напряжений от блока IEC 61850 SV Передатчик или IEC 61850 SV Передатчик 8 ASDU.
- **Значение smpCnt**: Значение поля smpCnt, которую содержит пакет, полученное с блока IEC 61850 SV Передатчик или IEC 61850 SV Передатчик 8 ASDU.

Выход блока

- **Сигнал**: Вектор из 4 выборок токов и 4 выборок напряжений, упорядоченных в соответствии с smpCnt.

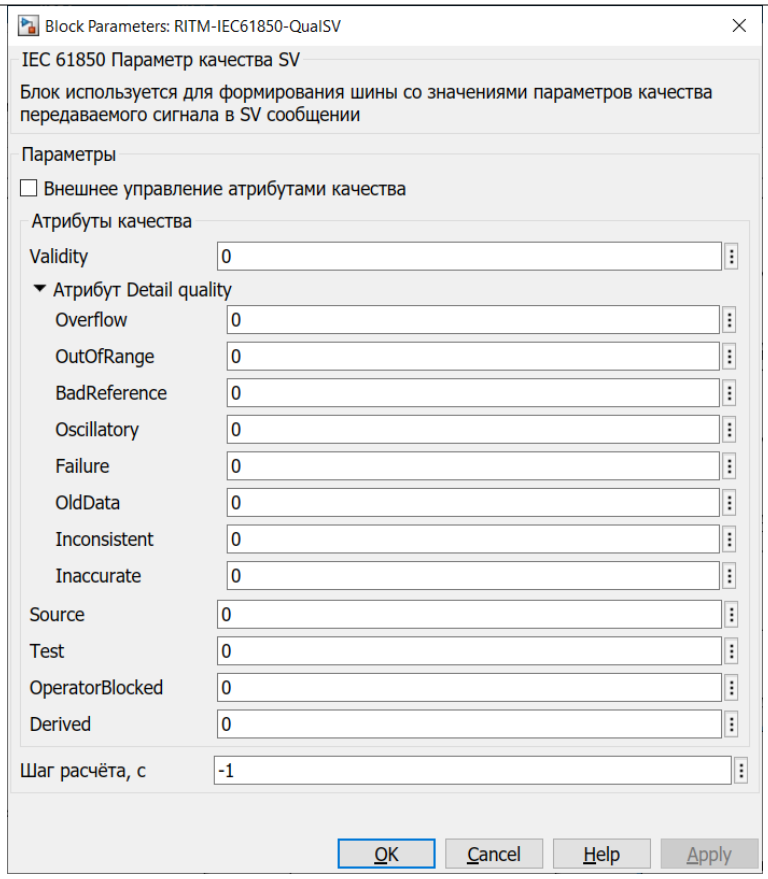
Параметры блока

- **Количество ASDU**: Количество ASDU в SV-кадре форму токов и напряжений, которых необходимо восстановить.
- **Шаг расчета входного сигнала, с**: Время между приемом двух SV-кадров (250 мкс для 80 выборок и 1 ASDU, 625 мкс для 256 выборок и 8 ASDU для частоты 50 Гц).

Блок IEC 61850 Параметр качества SV

Блок используется для формирования значения качества передаваемого сигнала в SV-сообщении.

Маска блока



Входы блока

Входы блока становятся активными в случае активации опции **“Внешнее управление атрибутами качества”** в параметрах блока. В таком случае становится возможным задать через соответствующие входы параметры: **Validity**, **Overflow**, **OutOfRange**, **BadReference**, **Oscillatory**, **Failure**, **oldData**, **inconsistency**, **inaccurate**, **Source**, **Test**, **operatorBlocked**, **Derived**.

Выход блока

На выходе блока формируется значение параметра качества, соответствующее 4 байтам поля Quality сигнала тока или напряжения SV-пакета в зависимости от значения атрибутов качества, предназначенное для подачи на вход блоков IEC 61850 SV Передатчик и IEC 61850 SV Передатчик 8 ASDU совместно с током или напряжением.

Параметры блока

- **Внешнее управление атрибутами качества**: Позволяет включить/отключить возможность задать атрибуты качества через соответствующие входы блока.
- **Шаг расчета**: Шаг модельного времени, с.

Атрибуты качества

- **Validity**: Атрибут качества Validity. Данный атрибут может принимать одно из трех значений:
  значение Good сообщает, что неисправностей не обнаружено. Тип данных *uint8*;
  значение Invalid – что получаемая информация неверна и ее нельзя использовать. Тип данных *uint8*;
  значение Questionable сообщает о проблемах с сервером, однако передаваемая информация может оставаться актуальной. Тип данных *uint8*.
- **Source**: Атрибут, предоставляющий информацию об источнике данных. Значение данных может быть истинным, исходящим от первичного процесса или вычисленным на основе данных первичного процесса (process), и установленным пользователем (substituted). Тип данных *uint8*.
- **Test**: Идентификатор нахождения устройства в режиме тестирования. Тип данных *boolean*.
- **operatorBlocked**: Идентификатор блокирования обновления данных оператором. Тип данных *boolean*.
- **Derived**: Идентификатор, определяющий, являются данные расчетными или измеренными. Тип данных *boolean*.

Атрибут Detail quality

Атрибуту Validity значения invalid и questionable присваиваются в зависимости от состояния дополнительных идентификаторов качества атрибута Detail quality. Атрибут Detail quality иметь следующие идентификаторы:

- **Overflow**: Идентификатор отсутствия правильного представления соответствующих значений данных. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **OutOfRange**: Идентификатор превышения значением измеренной величины предопределенного допустимого диапазона значений. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **BadReference**: Идентификатор потери калибровки устройства. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **Oscillatory**: Идентификатор фиксации дребезга для предотвращения излишней перегрузки каналов передачи данных. Тип

данных *boolean*.

- **Failure**: Идентификатор обнаружения системой самодиагностики устройства внутренней или внешней неисправности. Тип данных *boolean*.
- **oldData**: Идентификатор отсутствия обновления соответствующих данных в течение определенного времени. Тип данных *boolean*.
- **inconsistency**: Идентификатор обнаружения системой самодиагностики устройства несоответствия данных действительности. Тип данных *logical*.
- **inaccurate**: Идентификатор несоответствия значений данных требуемым точностным характеристикам. Тип данных *boolean*.

Примеры и демонстрации RITM IEC 61850 SV

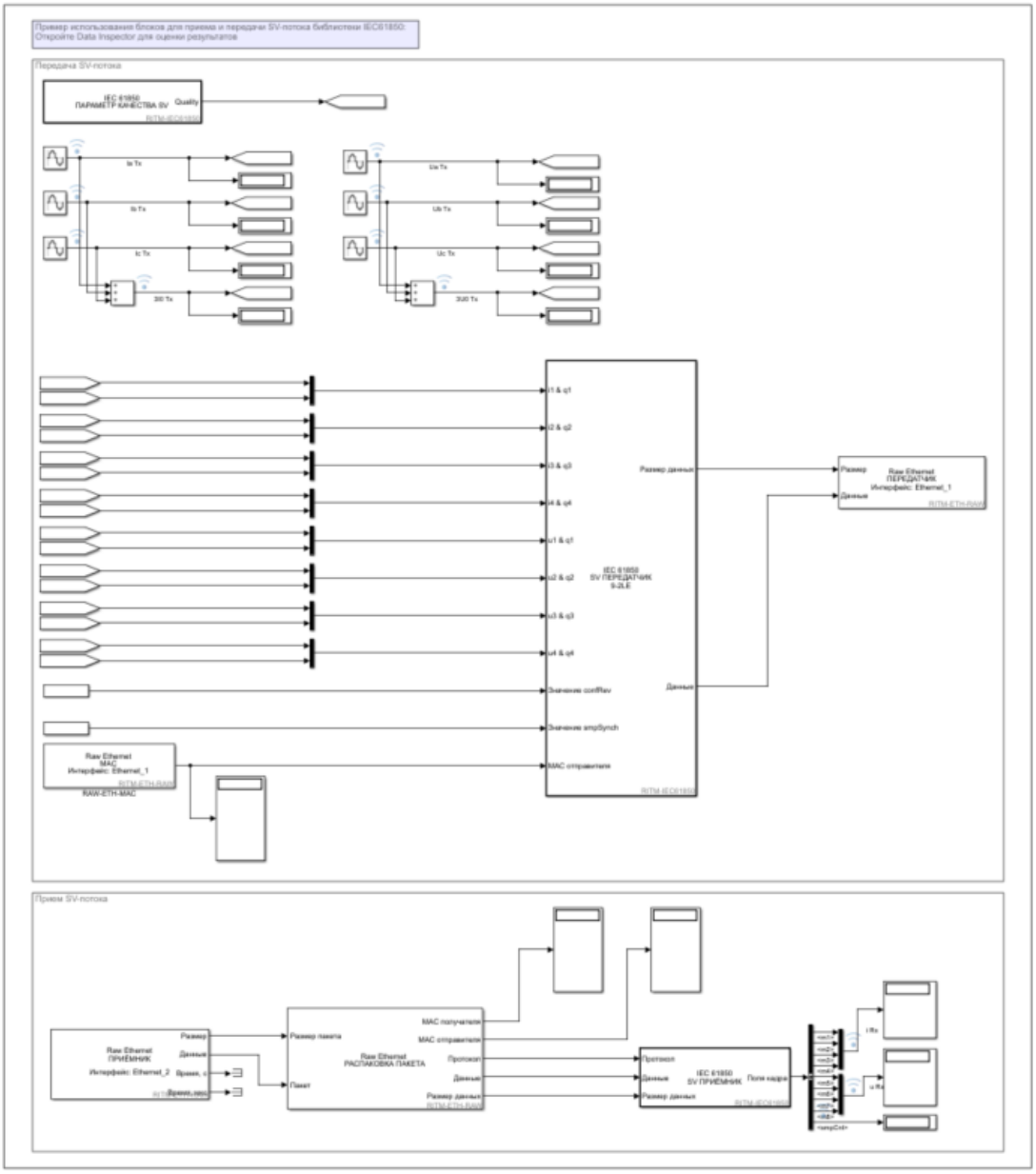
Содержание

- SV-поток с 80 выборками на период
- SV-поток с 256 выборками на период
- Настраиваемый SV-поток

SV-поток с 80 выборками на период

Пример использования блоков IEC 61850 для передачи и приема SV-потока с 80 выборками на период промышленной частоты.

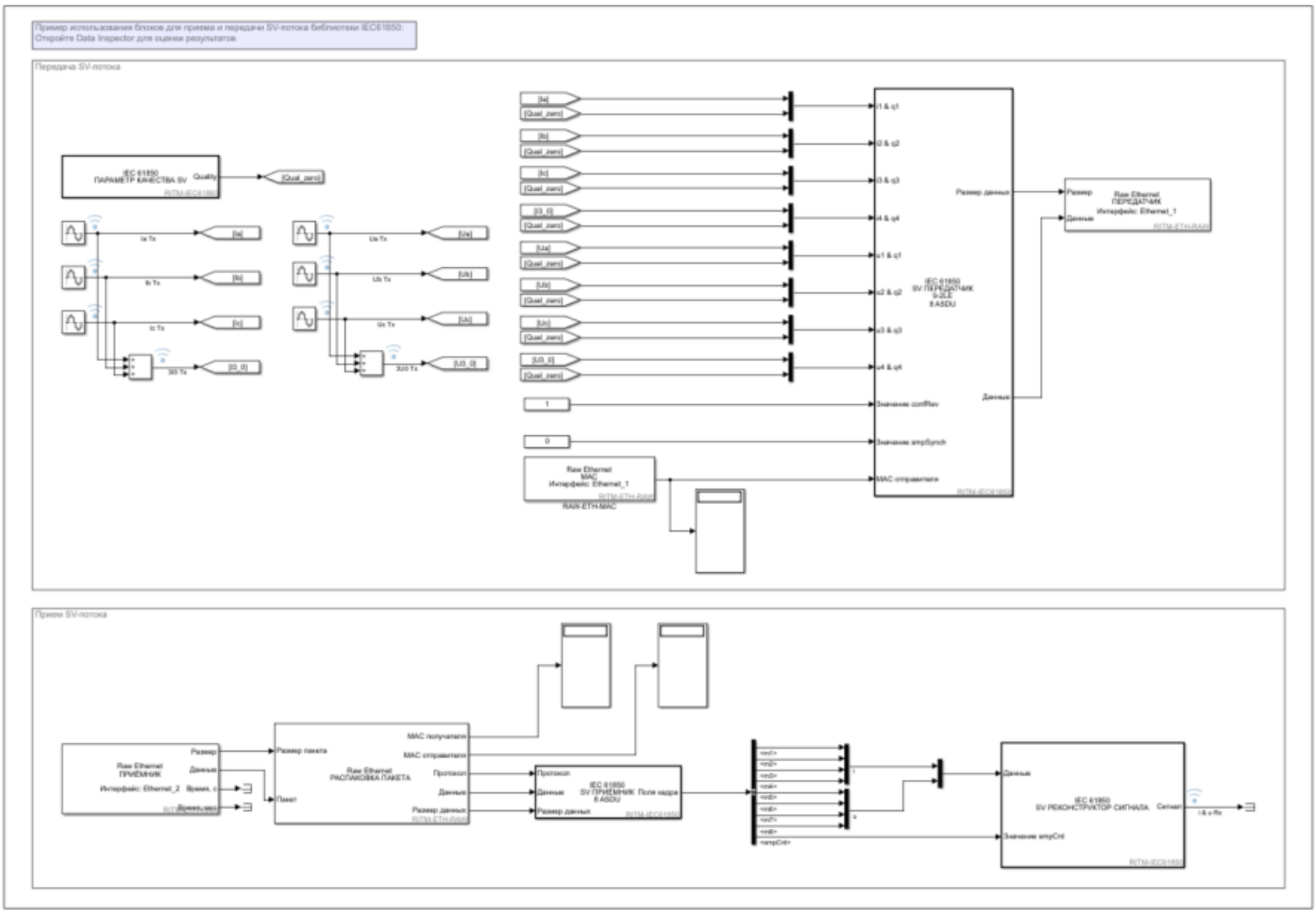
```
open_system('ritmdemo_block_iec61850_sv80.slx')
```



SV-поток с 256 выборками на период

Пример использования блоков IEC 61850 для передачи и приема SV-потока с 256 выборками на период промышленной частоты.

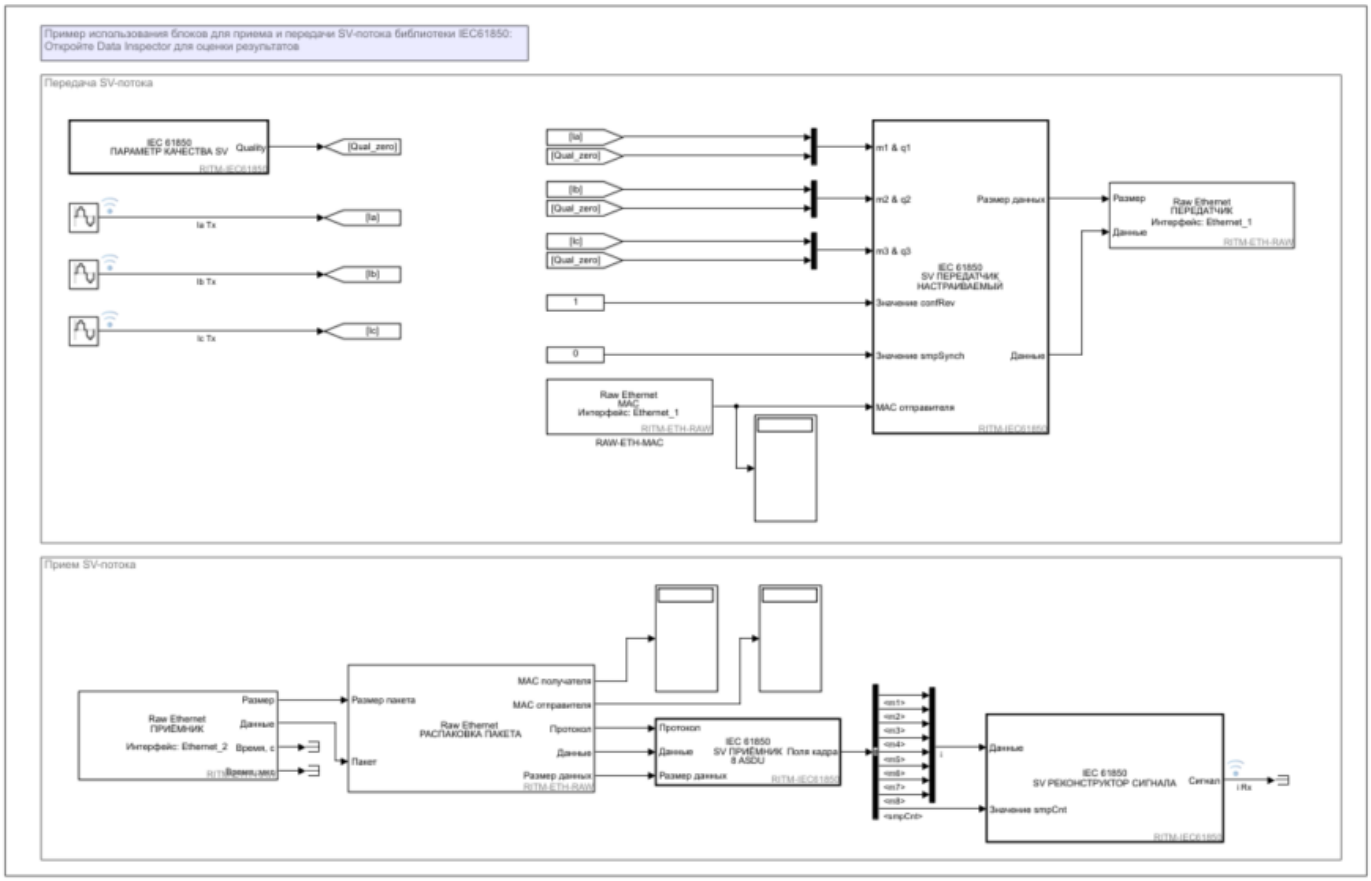
```
open_system('ritmdemo_block_iec61850_sv256.slx')
```



Настраиваемый SV-поток

Пример использования блоков IEC 61850 для передачи и приема SV-потока с 96 выборками на период промышленной частоты по корпоративному профилю ФСК ЕЭС.

```
open_system('ritmdemo_block_iec61850_sv_custom.slx')
```


Пример использования блоков для приема и передачи SV-потока библиотеки IEC61850:
Откройте Data Inspector для оценки результатов
Передача SV-потока
IEC 61850
ПАРАМЕТР КАЧЕСТВА SV
Quality
[Qual_zero]
Ia Tx
[Ia]
Ib Tx
[Ib]
Ic Tx
[Ic]
[Ia]
[Qual_zero]
[Ib]
[Qual_zero]
[Ic]
[Qual_zero]
1
0
Raw Ethernet
MAC
Интерфейс: Ethernet_1
RAW-ETH-MAC
m1 & q1
m2 & q2
m3 & q3
Значение confRev
Значение smpSynch
MAC отправителя
IEC 61850
SV ПЕРЕДАТЧИК
НАСТРАИВАЕМЫЙ
Размер данных
Данные
Размер
Данные
Raw Ethernet
ПЕРЕДАТЧИК
Интерфейс: Ethernet_1
Прием SV-потока
Raw Ethernet
ПРИЁМНИК
Интерфейс: Ethernet_2
Размер
Данные
Время, с
Время,мкс
Размер пакета
Пакет
Raw Ethernet
РАСПАКОВКА ПАКЕТА
MAC получателя
MAC отправителя
Протокол
Данные
Размер данных
Протокол
Данные
Размер данных
IEC 61850
SV ПРИЁМНИК Поля кадра
8 ASDU
<smpCnt>
Данные
Значение smpCnt
IEC 61850
SV РЕКОНСТРУКТОР СИГНАЛА
Сигнал
I Rx

Библиотека блоков для работы с GOOSE по IEC 61850

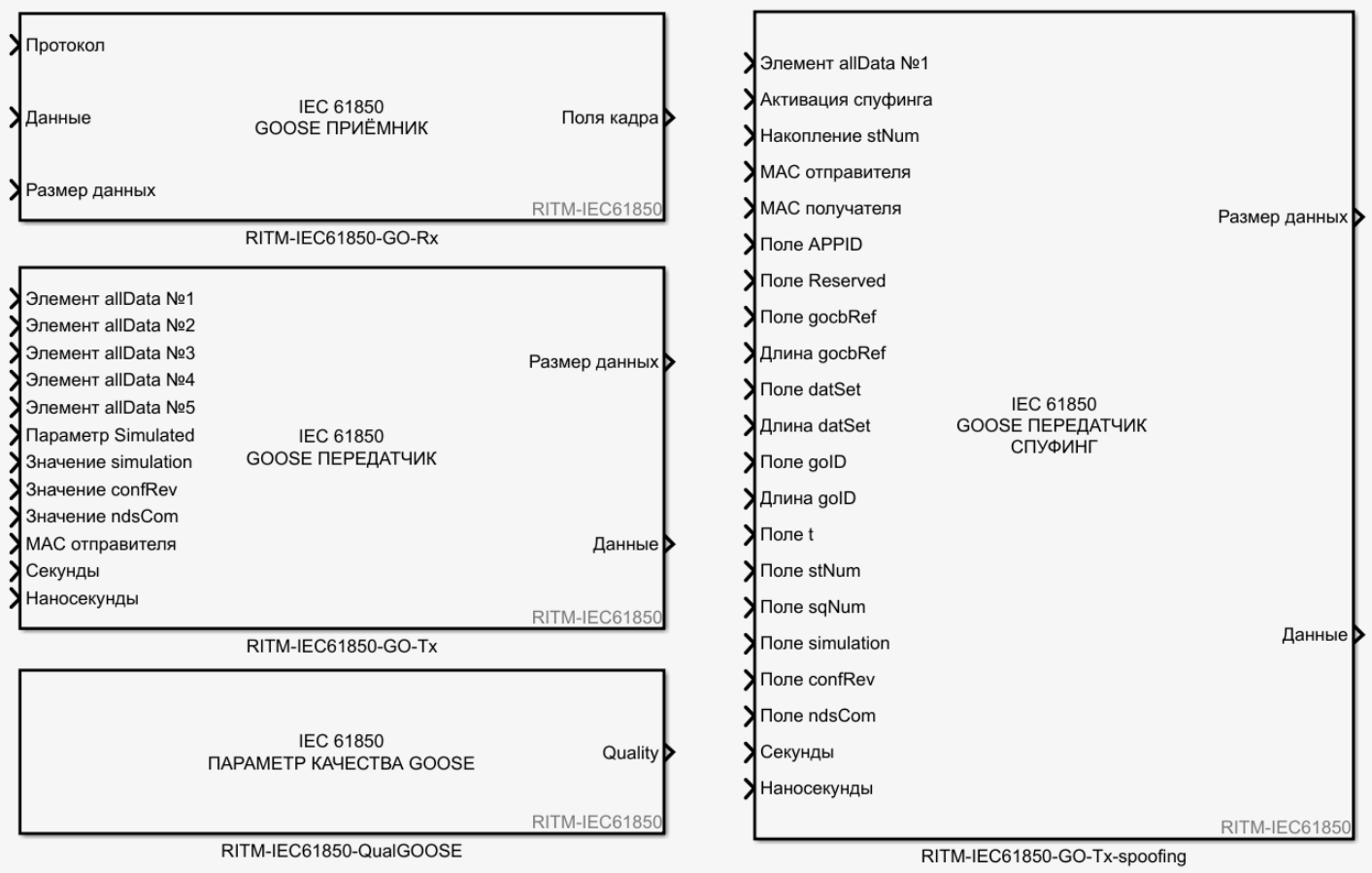
Содержание

- Описание
- Библиотека блоков
- Блок IEC 61850 GOOSE Передатчик
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок IEC 61850 GOOSE Передатчик Спуфинг
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок IEC 61850 GOOSE Приёмник
  - Маска блока
  - Параметры блока
  - Вход блока
  - Выход блока
- Блок IEC 61850 Параметр качества GOOSE
  - Маска блока
  - Входы блока
  - Выход блока
  - Параметры блока

Описание

Библиотека блоков для отправки и приёма GOOSE сообщений по Ethernet.
Перед использованием необходимо произвести настройку Ethernet портов КПМ РИТМ, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

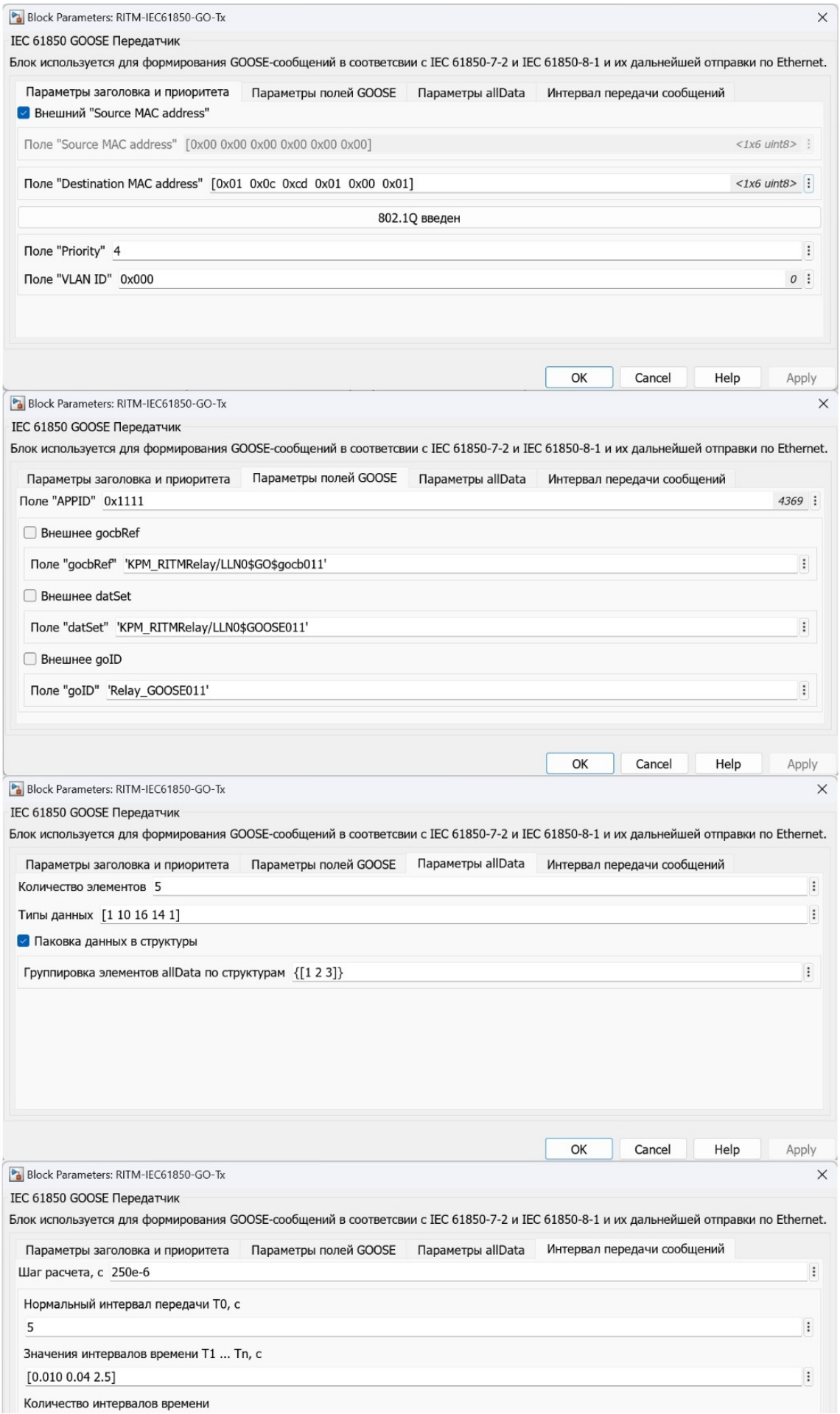
Библиотека блоков



Блок IEC 61850 GOOSE Передатчик

Блок используется для формирования GOOSE-сообщения из набора полей для дальнейшей передачи всех типов данных по Ethernet по IEC 61850-7-2 и IEC 61850-8-1.

Маска блока



Входы блока

- **Элемент allData №1 - Элемент allData №60**: Сигнал любого типа данных в соответствии с таблицей A.2 IEC61850-8-1 для паковки и отправки GOOSE-сообщением. Количество входов задается параметром **Количество элементов** маски блока.
- **Параметр Simulated**: Старший бит поля *Reserved 1*, сигнализирующий о том, что устройство находится в режиме теста, тип данных *boolean*.
- **Значение simulation**: Значение поля *simulation*. Если оно истинно, сообщение и, следовательно, его значение были выданы модулем моделирования и не являются реальными значениями, тип данных *boolean*.
- **Значение confRev**: Позволяет задать версию ревизии конфигурации, тип данных *uint32*.
- **Значение ndsCom**: Указывает в сообщении, что требуется пуско-наладка, тип данных *boolean*.
- **Значение gocbRef**: Позволяет напрямую задать значение поля **gocbRef** GOOSE-сообщения, тип данных *uint8*.
- **Значение datSet**: Позволяет напрямую задать значение поля **datSet** GOOSE-сообщения, тип данных *uint8*.
- **Значение goID**: Позволяет напрямую задать значение поля **goID** GOOSE-сообщения, тип данных *uint8*.
- **MAC отправителя**: Значение MAC-адреса в форме массива из 6 байт, вход используется в случае включения опции **Внешний “Source MAC address”** в параметрах блока, тип данных *uint8*.
- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*.

Выходы блока

- **Размер данных**: Размер GOOSE-сообщения, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные**: Сформированное GOOSE-сообщение, как набор байт, тип данных *uint8*.

Параметры блока

Панель “Параметры заголовка и приоритета”

- **Внешний “Source MAC address”**: Позволяет включить/отключить возможность задать MAC-адрес через вход **Source MAC** блока.
- **Поле “Source MAC address”**: Массив из 6 байт - MAC-адрес источника передачи GOOSE-сообщений.
- **Поле “Destination MAC address”**: Массив из 6 байт - MAC-адрес приемника GOOSE-сообщений, например для Multicast рассылке потокам присваиваются MAC-адреса в диапазоне 01:0C:CD:01:00:00 - 01:0C:CD:01:01:FF.
- **802.1Q введен/выведен**: Кнопка включения/отключения формирования поля VLAN.
- **Поле “Priority”**: Позволяет задать тег приоритета передачи трафика, формат *uint8*.
- **Поле “VLAN ID”**: Позволяет задать идентификатор, какой сети принадлежит поток (12 бит), тип данных *uint16*.

Панель “Параметры полей GOOSE”

- **Поле “APPID”**: Позволяет задать значение идентификационной метки сообщения APPID (стандарт предусматривает 0x8000 - 0xBFFF для класса сообщения 1A, 0x0000 - 0x3FFF для класса сообщения 1, однако возможно задать и иное значение).
- **Внешнее gocbRef**: Позволяет задать значение поля **gocbRef** GOOSE-сообщения через вход **Значение gocbRef** блока.
- **Поле “gocbRef”**: Позволяет напрямую задать значение поля **gocbRef** GOOSE-сообщения, тип данных *string*.
- **Внешнее datSet**: Позволяет задать значение поля **datSet** GOOSE-сообщения через вход **Значение datSet** блока.
- **Поле “datSet”**: Позволяет напрямую задать значение поля **datSet** GOOSE-сообщения, тип данных *string*.
- **Внешнее goID**: Позволяет задать значение поля **goID** GOOSE-сообщения через вход **Значение goID** блока.
- **Поле “goID”**: Позволяет напрямую задать значение поля **goID** GOOSE-сообщения, тип данных *string*.

Панель “Параметры allData”

- **Количество элементов**: Позволяет задать количество передаваемых элементов **Data** в поле **allData**, а также задает количество входов блока для приема значимой информации для передачи в GOOSE-сообщении, максимальное значение - 60. Для каждого входного сигнала должен быть указан тип данных в поле **Типы данных**.
- **Типы данных**: позволяет задать типы данных элементов **Data** поля **allData** в виде вектора, где порядковый номер элемента вектора соответствует номеру входа блока **Элемент allData №...**. В зависимости от значения элемента вектора сигналу присуждается тип данных в соответствии с таблицей:

Значение элемента вектора Типы данных	Тип данных	Тэг поля Data	Длина поля Data
1	Boolean	0x83	1
2	INT8	0x85	2

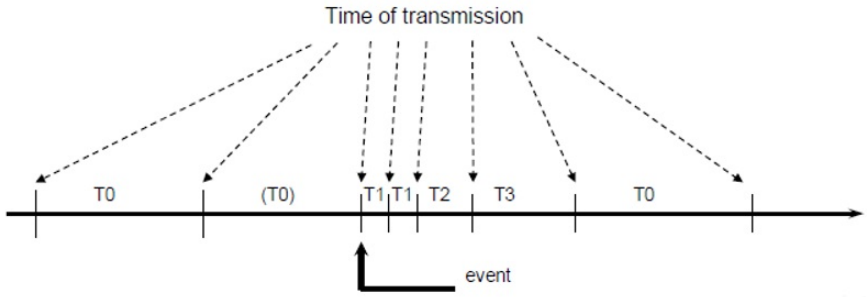
Значение элемента вектора Типы данных	Тип данных	Тэг поля Data	Длина поля Data
3	INT16	0x85	3
4	INT32	0x85	5
5	INT64	0x85	9
6	INT8U	0x86	2
7	INT16U	0x86	3
8	INT24U	-	-
9	INT32U	0x86	5
10	FLOAT32	0x87	4
11	ENUMERATED	0x85	2
12	CODED ENUM	0x84	2
13	OCTET STRING	0x89	20
14	VISIBLE STRING	0x8a	35
15	TimeStamp	0x91	8
16	Quality	0x84	3

Пример: Значение параметра **Количество элементов** равно 5, значение параметра **Типы данных** равно [1 10 16 4 1], тогда блок будет иметь 5 входов **Элемент allData №...**, первый вход будет кодировать входной сигнал для передачи как *Boolean*, второй вход как *float32*, третий как *TimeStamp*, четвертый как *int32*, пятый как *Boolean*.

- **Паковка данных в структуры**: Позволяет включить возможность паковки данных или части данных GOOSE-сообщения в формат структуры.
- **Группировка элементов allData по структурам**: Массив типа *cell-array*, который содержит в себе векторы с номерами элементов, которые будут сгруппированы в структуры. Каждый отдельный вектор внутри основного массива - это отдельная структура внутри поля *allData* GOOSE-сообщения. Группировка элементов внутри вектора структуры должна происходить от меньшего номера элемента к большему. Сигналы внутри вектора должны находиться на соседних входах блока. *Пример, можно - [1 2 3], [3 4 5], [2 3], нельзя - [1 3 5], [1 2 4].*
  Группировка векторов структур внутри массива *cell-array* должна происходить аналогичным образом: от вектора с наименьшими номерами элементов к вектору с наибольшими номерами элементов. *Пример, можно - {[1 2 3] [4 5]}, {[1 2] [4 5]}, {[1 2] [3 4]}, нельзя - {[4 5] [1 2 3]}, {[4 5] [1 2], [3 4] [1 2]}.*
  Элементы, не входящие ни в один из векторов внутри cell-array массива будут инкапсулированы в GOOSE-сообщение вне структуры, последовательно, согласно номеру входа, который они занимают. *Пример, {[1 2] [4 5]} - состав поля allData: 1 и 2 сигналы в структуре, третий сигнал вне структуры, 4 и 5 сигналы в структуре, {[3 4]} - состав поля allData: 1 и 2 сигналы вне структуры, 3 и 4 сигналы в структуре, 5 сигнал вне структуры.*

Панель “Интервал передачи сообщения”

- **Шаг расчета, с**: Шаг расчета, с которым будет работать данный блок. Должен быть кратен основному шагу расчета, указанному в настройках модели.
- **Нормальный интервал передачи T0, с**: Значение нормального интервала времени передачи между GOOSE-сообщениями, с.
- **Значения интервалов времени T1 ... Tn, с**: Вектор значений интервалов времени передачи T1 ... Tn между GOOSE-сообщениями, после изменения хотя бы одного из атрибутов данных сообщения, с.
- **Количества интервалов времени**: Вектор количества интервалов времени передачи T1 ... Tn между GOOSE-сообщениями до достижения нормального периода передачи.

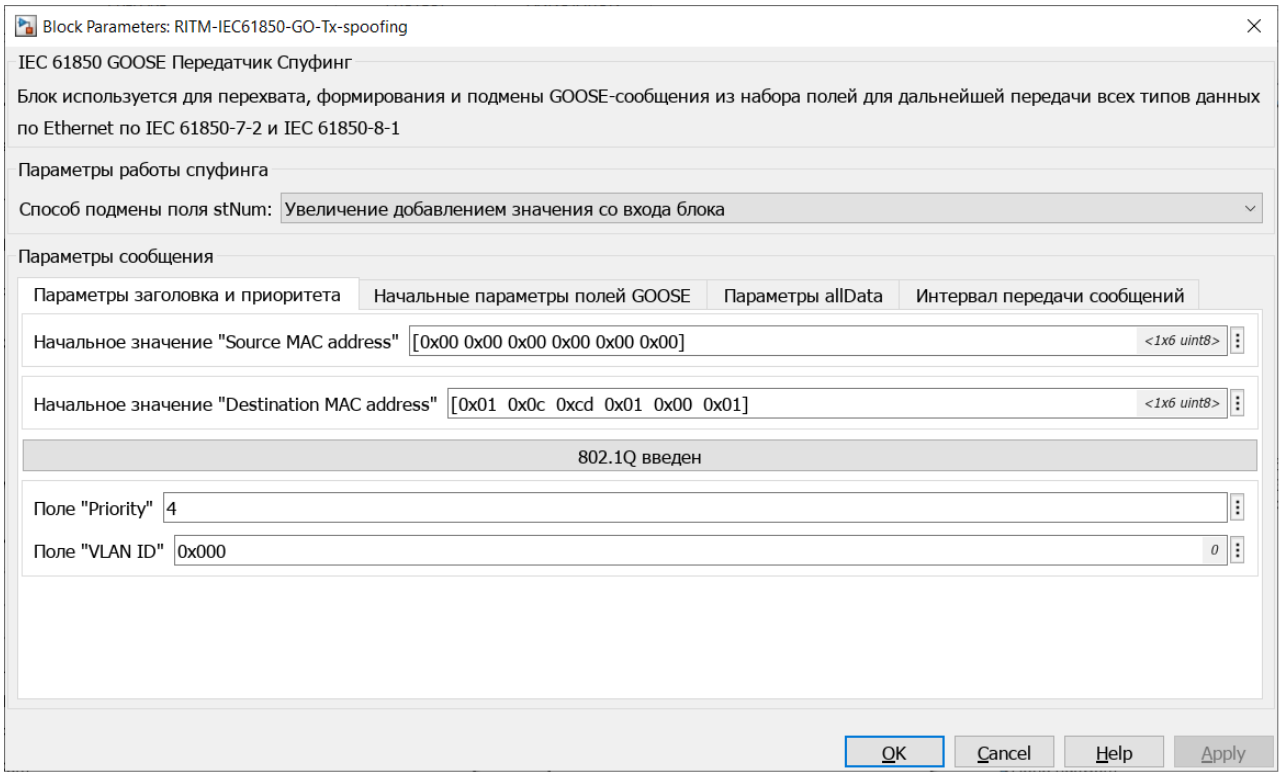


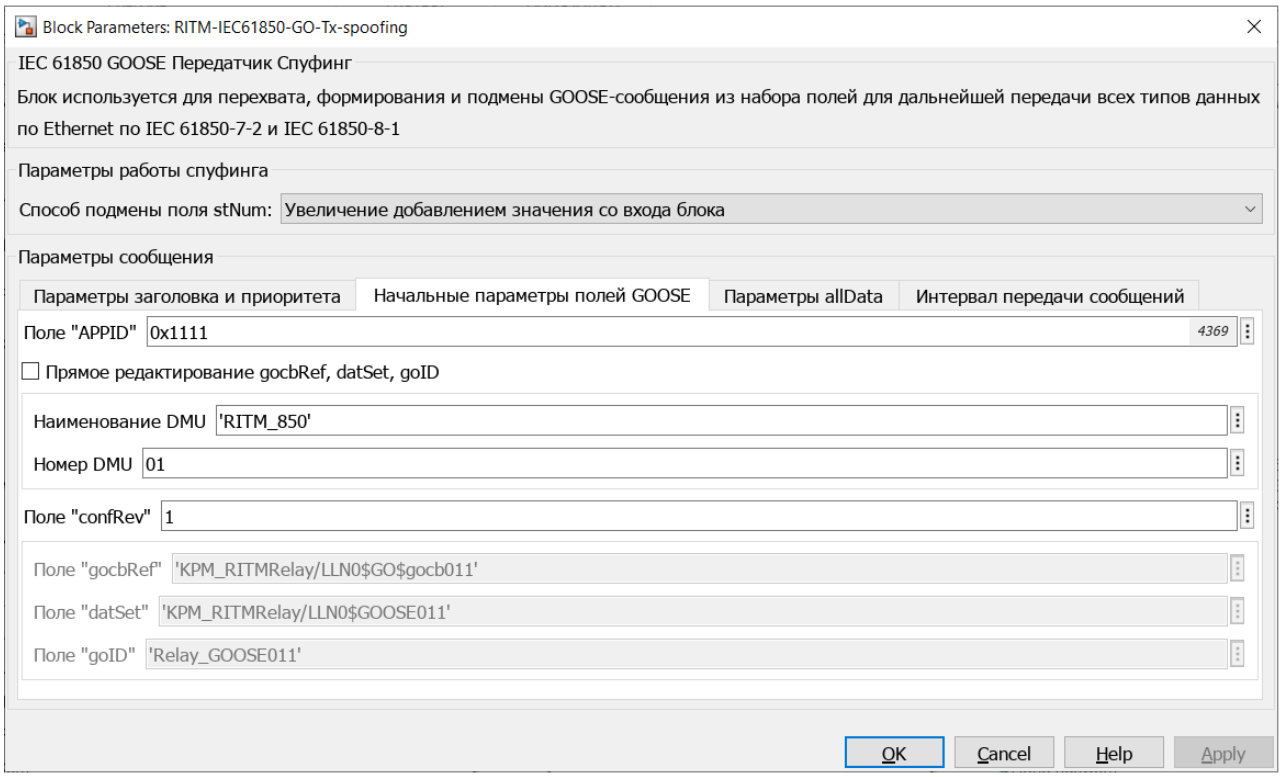
Блок IEC 61850 GOOSE Передатчик Спуфинг

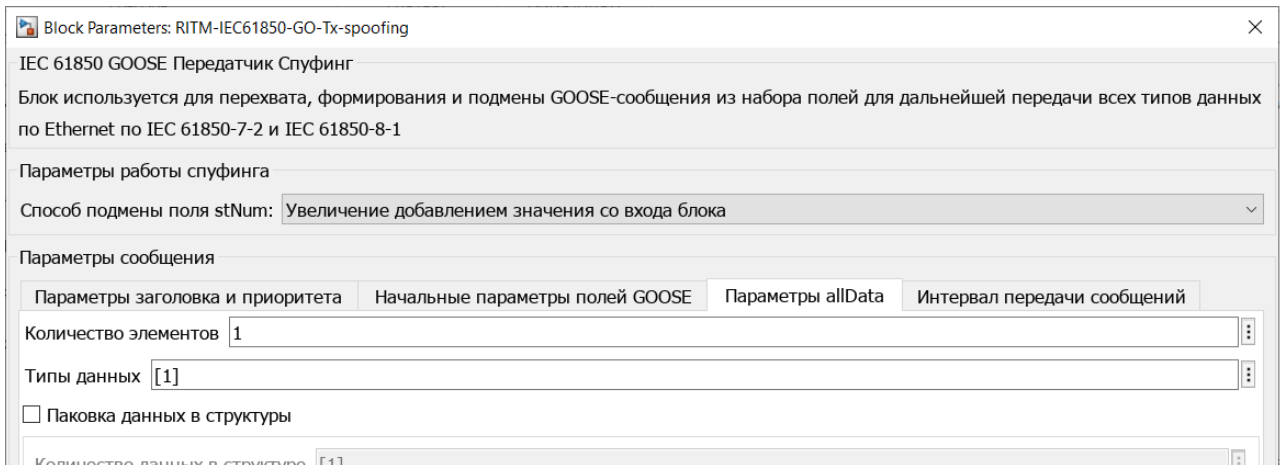
Блок используется для перехвата, формирования и подмены GOOSE-сообщения из набора полей для дальнейшей передачи всех

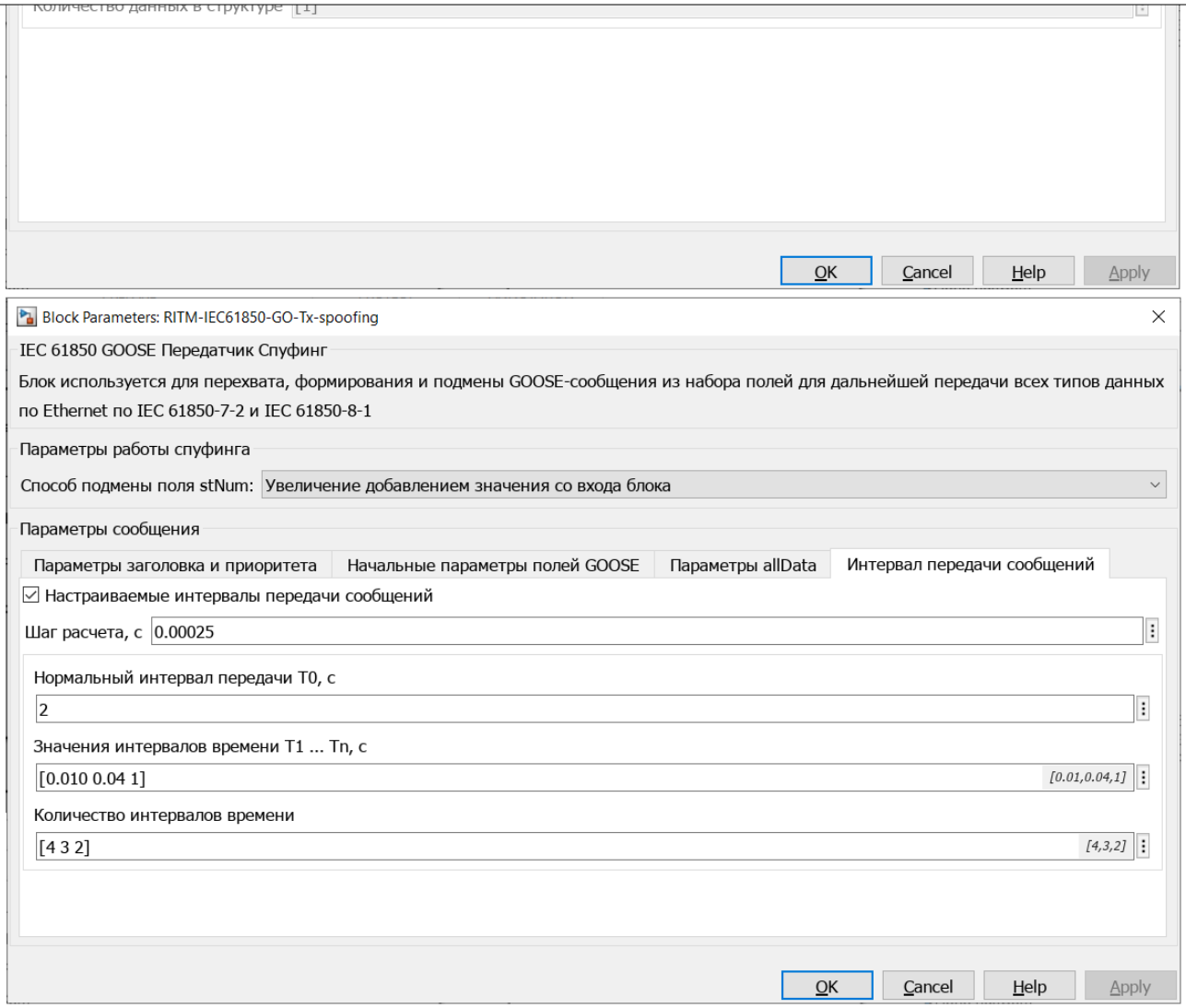
типов данных по Ethernet по IEC 61850-7-2 и IEC 61850-8-1.

Маска блока









Входы блока

- **Элемент allData №1 - Элемент allData №60**: Сигнал любого типа данных в соответствии с таблицей A.2 IEC61850-8-1 для паковки и отправки GOOSE-сообщением. Количество входов задается параметром **Количество элементов** маски блока.
- **Активация спуфинга**: Вход для активации подмены перехваченного GOOSE-сообщения по восходящему фронту входного сигнала путем подхвата значений со входов значений полей сообщения.
- **Накопление stNum**: Вход для реализации увеличения значения stNum в соответствии со способом выбранным в параметрах блока.
- **MAC отправителя**: Значение MAC-адреса отправителя в форме массива из 6 байт, тип данных *uint8*.
- **MAC получателя**: Значение MAC-адреса получателя в форме массива из 6 байт, тип данных *uint8*.
- **Поле APPID**: Значение поля APPID сообщения, тип данных *uint16*.
- **Поле Reserved1**: Значение поля Reserved1 сообщения, тип данных *uint16*.
- **Поле gocbRef**: Значение поля gocbRef сообщения в виде массива, тип данных *uint8*.
- **Длина gocbRef**: Значение длины поля gocbRef сообщения, тип данных *uint8*.
- **Поле datSet**: Значение поля datSet сообщения в виде массива, тип данных *uint8*.
- **Длина datSet**: Значение длины поля datSet сообщения, тип данных *uint8*.
- **Поле goID**: Значение поля goID сообщения в виде массива, тип данных *uint8*.
- **Длина goID**: Значение длины поля goID сообщения, тип данных *uint8*.
- **Поле t**: Значение поля t сообщения в виде массива, тип данных *uint8*.
- **Поле stNum**: Значение поля stNum сообщения, тип данных *uint32*.
- **Поле sqNum**: Значение поля sqNum сообщения, тип данных *uint32*.
- **Поле simulation**: Значение поля simulation сообщения, тип данных *boolean*.
- **Поле confRev**: Значение поля confRev сообщения, тип данных *uint32*.
- **Поле ndsCom**: Значение поля ndsCom сообщения, тип данных *boolean*.
- **Секунды**: Время в секундах, прошедшее с полуночи 1 января 1970 года, тип данных *uint64*.
- **Наносекунды**: Число наносекунд, прошедшее с начала секунды, тип данных *uint64*.

Выходы блока

- **Размер данных**: Размер GOOSE-сообщения, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные**: Сформированное GOOSE-сообщение, как набор байт, тип данных *uint8*.

Параметры блока

Панель “Параметры работы спуфинга”

- **Способ подмены поля stNum**: Позволяет выбрать способ подмены значения поля stNum ложного потока GOOSE-сообщений для реализации спуфинга. Опция **Увеличение добавлением значения со входа блока** - добавляет единицу к значению счетчика stNum при появлении на входе **Накопление stNum** возрастающего фронта. Опция **Увеличение путем симуляции изменения набора данных** - имитирует изменение набора данных GOOSE-сообщения без фактического изменения данных при фиксации возрастающего фронта на входе **Накопление stNum**.

Панель “Параметры заголовка и приоритета”

- **Начальное значение “Source MAC address”**: Массив из 6 байт - MAC-адрес источника передачи GOOSE-сообщений до активации спуфинга.
- **Начальное значение “Destination MAC address”**: Массив из 6 байт - MAC-адрес приемника GOOSE-сообщений до активации спуфинга.
- **802.1Q введен/выведен**: Кнопка включения/отключения формирования поля VLAN.
- **Поле “Priority”**: Позволяет задать тег приоритета передачи трафика, формат *uint8*.
- **Поле “VLAN ID”**: Позволяет задать идентификатор, какой сети принадлежит поток (12 бит), тип данных *uint16*.

Панель “Начальные параметры полей GOOSE”

- **Поле “APPID”**: Позволяет задать значение идентификационной метки сообщения APPID (стандарт предусматривает 0x8000 - 0xBFFF для класса сообщения 1A, 0x0000 - 0x3FFF для класса сообщения 1, однако возможно задать и иное значение) до активации спуфинга.
- **Прямое редактирование gocbRef, datSet, goID**: Позволяет переключаться между видами задания полей **gocbRef, datSet, goID** в GOOSE-сообщении: косвенно через поля **Наименование DMU** и **Номер DMU** или напрямую через поля **gocbRef, datSet, goID** маски блока до активации спуфинга.
- **Наименование DMU**: Позволяет задать имя DMU для формирования полей **gocbRef, datSet, goID** GOOSE-сообщения до активации спуфинга, тип данных *string*.
- **Номер DMU**: Позволяет задать номер DMU для формирования полей **gocbRef, datSet, goID** GOOSE-сообщения до активации спуфинга, тип данных *uint8*.
- **Поле “confRev”**: Позволяет задать версию ревизии конфигурации до активации спуфинга, тип данных *uint32*.
- **Поле “gocbRef”**: Позволяет напрямую задать значение поля **gocbRef** GOOSE-сообщения до активации спуфинга, тип данных *string*.
- **Поле “datSet”**: Позволяет напрямую задать значение поля **datSet** GOOSE-сообщения до активации спуфинга, тип данных *string*.
- **Поле “goID”**: Позволяет напрямую задать значение поля **goID** GOOSE-сообщения до активации спуфинга, тип данных *string*.

Панель “Параметры allData”

- **Количество элементов**: Позволяет задать количество передаваемых элементов **Data** в поле **allData**, а также задает количество входов блока для приема значимой информации для передачи в GOOSE-сообщении, максимальное значение - 60.
- **Типы данных**: позволяет задать типы данных элементов **Data** поля **allData** в виде вектора, где порядковый номер элемента вектора соответствует номеру входа блока **Элемент allData №...**.

Значение элемента вектора Типы данных	**Тип данных**	**Тэг поля Data**	**Длина поля Data**
1	Boolean	0x83	1
2	INT8	0x85	2
3	INT16	0x85	3
4	INT32	0x85	5
5	INT64	0x85	9
6	INT8U	0x86	2
7	INT16U	0x86	3
8	INT24U	-	-
9	INT32U	0x86	5
10	FLOAT32	0x87	4
11	ENUMERATED	0x85	2
12	CODED ENUM	0x84	2
13	OCTET STRING	0x89	20

Значение элемента вектора Типы данных	Тип данных	Тэг поля Data	Длина поля Data
14	VISIBLE STRING	0x8a	35
15	TimeStamp	0x91	8
16	Quality	0x84	3

- **Паковка данных в структуры**: Позволяет включить возможность паковки данных или части данных GOOSE-сообщения в формат структуры.
- **Количество данных в структуре**: Вектор элементы которого указывают сколько элементов данных GOOSE-сообщения будет сформировано в форму структуры.

Панель “Интервал передачи сообщения”

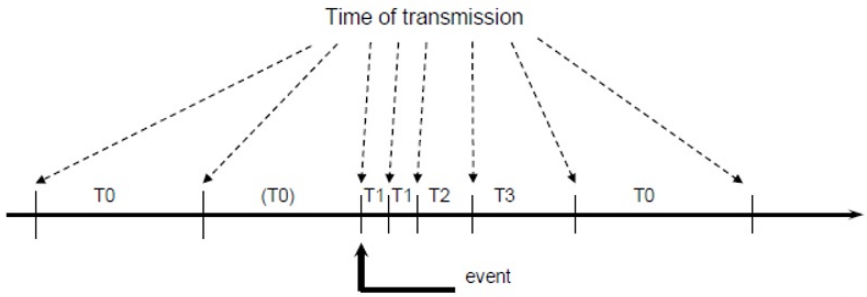
- **Настраиваемые интервалы передачи сообщений**: Позволяет включить возможность гибкой настройки времени и количества интервалов передачи сообщений.
- **Шаг расчета, с**: Шаг модельного времени, с.

При активации функции **Настраиваемые интервалы передачи сообщений**:

- **Нормальный интервал передачи T0, с**: Значение нормального интервала времени передачи между GOOSE-сообщениями, с.
- **Значения интервалов времени T1 … Tn, с**: Вектор значений интервалов времени передачи T1 … Tn между GOOSE-сообщениями, после изменения хотя бы одного из атрибутов данных сообщения, с.
- **Количества интервалов времени**: Вектор количества интервалов времени передачи T1 … Tn между GOOSE-сообщениями до достижения нормального периода передачи.

При деактивации функции **Настраиваемые интервалы передачи сообщений**:

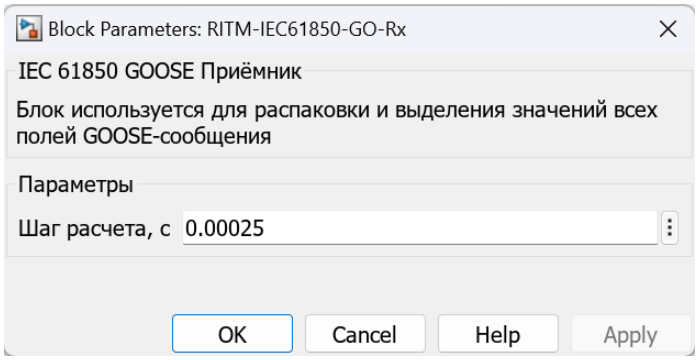
- **Нормальный интервал передачи T0, с**: Значение нормального интервала времени передачи между GOOSE-сообщениями, с.
- **Наименьший интервал T1, с**: Значение минимального интервала времени передачи между GOOSE-сообщениями, после изменения хотя бы одного из атрибутов данных сообщения, с.
- **Интервал до достижения нормального состояния T2, с**: Значение второго интервала времени передачи между GOOSE-сообщениями до достижения нормального состояния, с.
- **Интервал до достижения нормального состояния T3, с**: Значение третьего интервала времени передачи между GOOSE-сообщениями до достижения нормального состояния, с.



Блок IEC 61850 GOOSE Приёмник

Блок используется для распаковки и выделения значений всех полей GOOSE-сообщения.

Маска блока



Параметры блока

- **Шаг расчета, с**: Шаг модельного времени, с.

Вход блока

- **Протокол**: Значение поля метки протокола, которую содержит принятый пакет, полученный с блока Raw Ethernet Распаковка пакета.
- **Данные**: Сегмент данных пакета, полученный с блока Raw Ethernet Распаковка пакета.

Выход блока

- **Поля кадра**: Шина, содержащая все поля GOOSE-сообщения. Для выделения значения определенного поля или набора полей необходимо воспользоваться блоком **Bus Selector** библиотеки блоков Simulink. Все элементы **Data** поля **allData** содержатся в сигнале **“allData data”** шины, который представляет из себя матрицу со значениями этих элементов, где каждая строка является значением соответствующего элемента **Data** в байтах. Количество байт с информацией каждого элемента **Data** (элементов строки матрицы) соответствует значению ячейки вектора сигнала **“allData length”** шины.

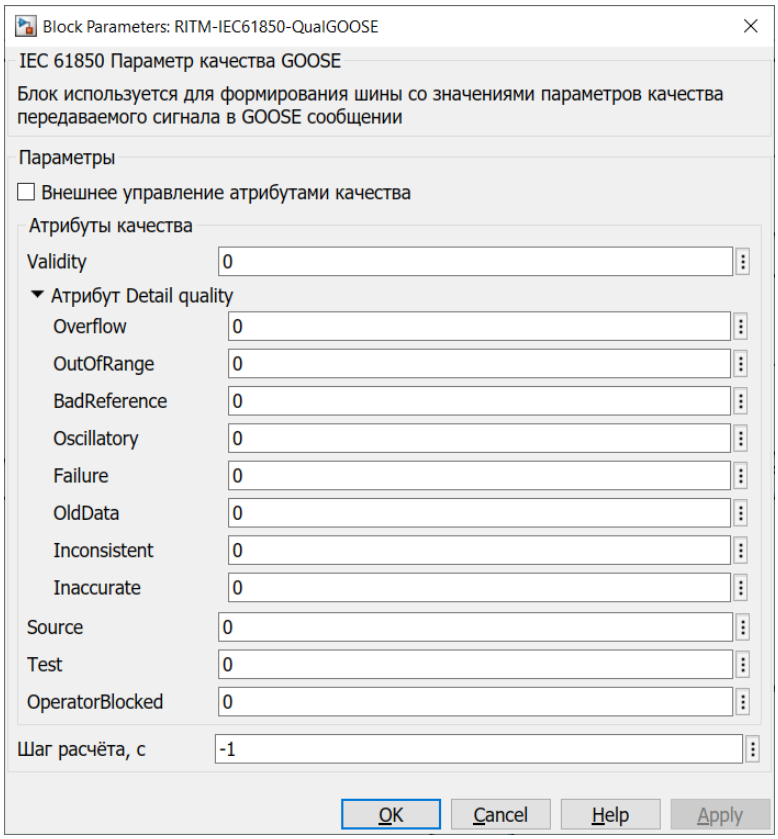
Пример: Передача сигналов любых типов данных по GOOSE-сообщениям.

Значение на выходе блока устанавливается и удерживается при захвате GOOSE-сообщения и обновляется при захвате последующего GOOSE-сообщения.

Блок IEC 61850 Параметр качества GOOSE

Блок используется для формирования значения качества передаваемого сигнала в GOOSE-сообщении.

Маска блока



Входы блока

Входы блока становятся активными в случае активации опции **“Внешнее управление атрибутами качества”** в параметрах блока. В таком случае становится возможным задать через соответствующие входы параметры: **Validity**, **Overflow**, **OutOfRange**, **BadReference**, **Oscillatory**, **Failure**, **oldData**, **inconsistency**, **inaccurate**, **Source**, **Test**, **operatorBlocked**.

Выход блока

На выходе блока формируется значение параметра качества, соответствующее 2 байтам поля Quality GOOSE-сообщения в зависимости от значения атрибутов качества, предназначенное для подачи на вход блока IEC 61850 GOOSE Передатчик All types.

Параметры блока

- **Внешнее управление атрибутами качества**: Позволяет включить/отключить возможность задать атрибуты качества через соответствующие входы блока.
- **Шаг расчета**: Шаг модельного времени, с.

Атрибуты качества

- **Validity**: Атрибут качества Validity. Данный атрибут может принимать одно из трех значений:
  значение Good сообщает, что неисправностей не обнаружено. Тип данных *uint8*;
  значение Invalid – что получаемая информация неверна и ее нельзя использовать. Тип данных *uint8*;
  значение Questionable сообщает о проблемах с сервером, однако передаваемая информация может оставаться актуальной.

Тип данных *uint8*.

- **Source**: Атрибут, предоставляющий информацию об источнике данных. Значение данных может быть истинным, исходящим от первичного процесса или вычисленным на основе данных первичного процесса (process), и установленным пользователем (substituted). Тип данных *uint8*.
- **Test**: Идентификатор нахождения устройства в режиме тестирования. Тип данных *boolean*.
- **operatorBlocked**: Идентификатор блокирования обновления данных оператором. Тип данных *boolean*.

Атрибут Detail quality

Атрибуту Validity значения invalid и questionable присваиваются в зависимости от состояния дополнительных идентификаторов качества атрибута Detail quality. Атрибут Detail quality иметь следующие идентификаторы:

- **Overflow**: Идентификатор отсутствия правильного представления соответствующих значений данных. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **OutOfRange**: Идентификатор превышения значением измеренной величины предопределенного допустимого диапазона значений. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **BadReference**: Идентификатор потери калибровки устройства. Рассмотренный идентификатор применим только для измерений. Тип данных *boolean*.
- **Oscillatory**: Идентификатор фиксации дребезга для предотвращения излишней перегрузки каналов передачи данных. Тип данных *boolean*.
- **Failure**: Идентификатор обнаружения системой самодиагностики устройства внутренней или внешней неисправности. Тип данных *boolean*.
- **oldData**: Идентификатор отсутствия обновления соответствующих данных в течение определенного времени. Тип данных *boolean*.
- **inconsistency**: Идентификатор обнаружения системой самодиагностики устройства несоответствия данных действительности. Тип данных *logical*.
- **inaccurate**: Идентификатор несоответствия значений данных требуемым точностным характеристикам. Тип данных *boolean*.

Примеры и демонстрации RITM IEC 61850 GOOSE

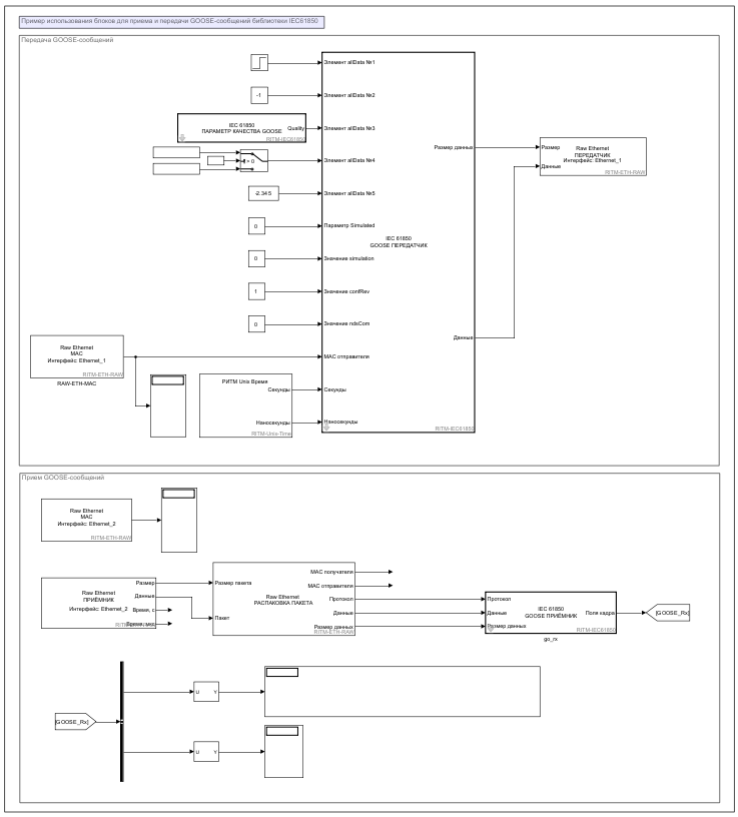
Содержание

- Передача сигналов любых типов данных по GOOSE-сообщениям
- Пример моделирования GOOSE-спуфинга

Передача сигналов любых типов данных по GOOSE-сообщениям

Пример использования блоков IEC 61850 для передачи и приема сигналов любого типа данных по GOOSE-сообщениям в соответствии с таблицей A.2 IEC 61850-8-1.

```
open_system('ritmdemo_block_iec61850_goose.slx')
```

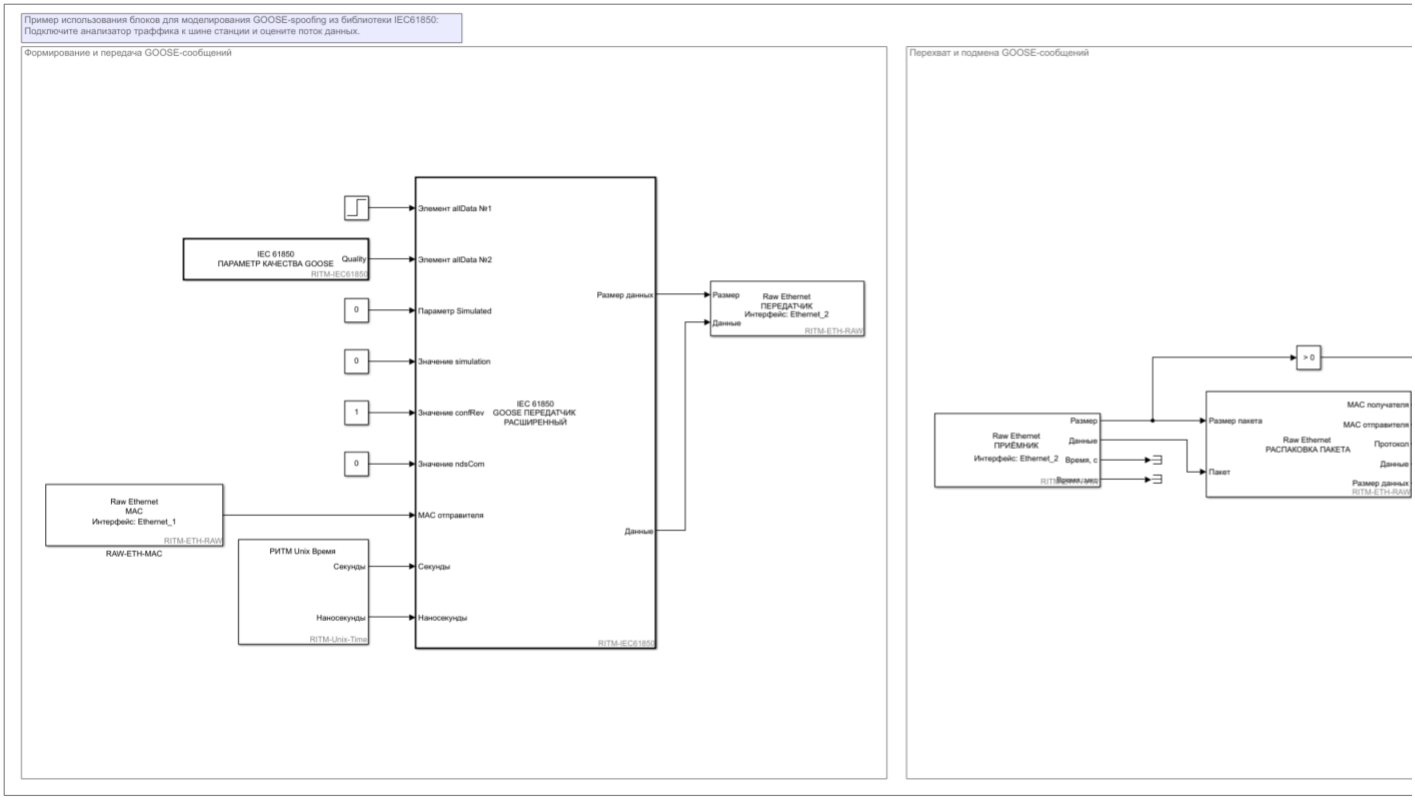


Пример моделирования GOOSE-спуфинга

Пример использования блоков IEC 61850 для моделирования GOOSE-спуфинга.

```
open_system('ritmdemo_block_iec61850_goose_spoofing.slx')
```

В данном примере создается поток GOOSE-сообщений для передачи в шину станции через один из Ethernet портов машины РИТМ. Данный поток отслеживается другим портом машины для его перехвата, декомпозиции на поля и дальнейшей подмены для отправки в шину станции ложного потока GOOSE-сообщений с помощью блока IEC 61850 GOOSE Передатчик Спуфинг.
Для правильной работы демонстрации создайте шину станции, для этого подключите Ethernet порты машины РИТМ с помощью кабеля типа патч-корд к внешнему коммутатору. Подключите к шине станции сетевой анализатор и оцените поток данных.
Обратите внимание, что способ накопления значения поля stNum в ложном потоке GOOSE-сообщений возможно изменить в параметрах блока IEC 61850 GOOSE Передатчик Спуфинг, а также меняя параметры блока генератора импульсов.


Пример использования блоков для моделирования GOOSE-spoofing из библиотеки IEC61850:
Подключите анализатор траффика к шине станции и оцените поток данных.
Формирование и передача GOOSE-сообщений
Элемент allData №1
IEC 61850
ПАРАМЕТР КАЧЕСТВА GOOSE
Quality
RITM-IEC61850
Элемент allData №2
0
Параметр Simulated
0
Значение simulation
1
Значение confRev
IEC 61850
GOOSE ПЕРЕДАТЧИК
РАСШИРЕННЫЙ
0
Значение ndsCom
Raw Ethernet
MAC
Интерфейс: Ethernet_1
RITM-ETH-RAW
RAW-ETH-MAC
MAC отправителя
РИТМ Unix Время
Секунды
Секунды
Наносекунды
Наносекунды
RITM-Unix-Time
Размер данных
Данные
RITM-IEC61850
Размер
Raw Ethernet
ПЕРЕДАТЧИК
Интерфейс: Ethernet_2
Данные
RITM-ETH-RAW
Перехват и подмена GOOSE-сообщений
> 0
Raw Ethernet
ПРИЁМНИК
Интерфейс: Ethernet_2
Размер
Данные
Время, с
Время, нс
RITM-ETH-RAW
Размер пакета
Пакет
Raw Ethernet
РАСПАКОВКА ПАКЕТА
MAC получателя
MAC отправителя
Протокол
Данные
Размер данных
RITM-ETH-RAW

Блоки поддержки PRP

Содержание

- Библиотека блоков для работы с PRP по IEC 62439-3-2016

Библиотека блоков для работы с PRP по IEC 62439-3-2016

Библиотека блоков для формирования кадров с RCT для дальнейшей отправки по Ethernet и отбрасывания дубликатов PRP-кадров, которые были приняты по Ethernet.

Библиотека блоков для работы с PRP по IEC 62439-3-2016

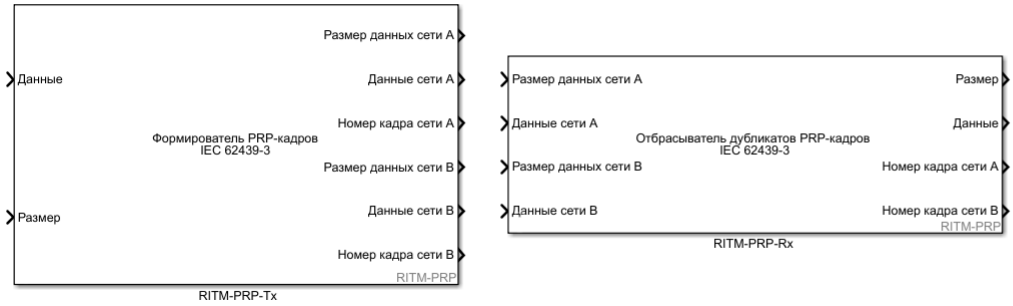
Содержание

- Описание
- Библиотека блоков
- Блок Формирователь PRP-кадров
  - Маска блока
  - Входы блока
  - Выходы блока
- Блок Отбрасыватель дубликатов PRP-кадров
  - Маска блока
  - Входы блока
  - Выходы блока

Описание

Библиотека блоков для формирования кадров с RCT для дальнейшей отправки по Ethernet и отбрасывания дубликатов PRP-кадров, которые были приняты по Ethernet.
Перед использованием необходимо произвести настройку Ethernet портов КПМ РИТМ, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

Библиотека блоков



Блок Формирователь PRP-кадров

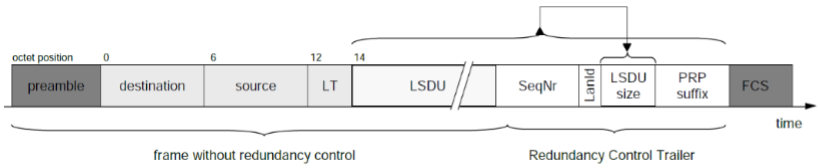
Блок используется для формирования PRP-кадров для дальнейшей передачи по Ethernet. К каждому входящему кадру присоединяется трейлер Redundancy Control Trailer (RCT). RCT размером 6 байт состоит из:

Номер кадра в последовательности SeqNr - 16 бит

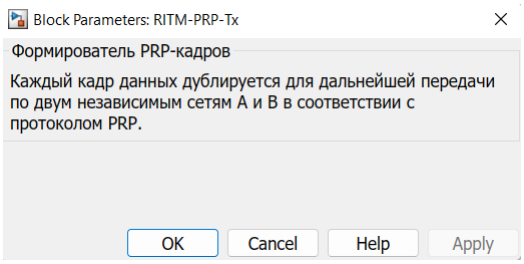
Путь LanId - идентификатор сети A (1010) или B (1011), по которой будет передаваться пакет - 4 бита

Размер поля данных LSDU size - 12 бит (данные + RCT)

PRP suffix - 16 бит (0x88FB)



Маска блока



Входы блока

- **Данные**: Сформированный кадр, как набор байт, тип данных *uint8*.
- **Размер**: Размер кадра, как число передаваемых байт, тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.

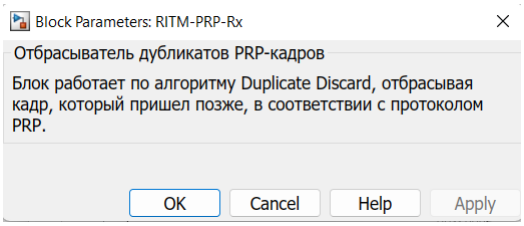
Выходы блока

- **Размер данных сети A**: Размер кадра для передачи по сети A, как число передаваемых байт (+6 байт к исходному размеру за счет RCT), тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные сети A**: Сформированный кадр с RCT сети A (Lan ID = 1010), как набор байт, тип данных *uint8*.
- **Номер кадра сети A**: Номер кадра сети A (значение поля SeqNr), тип данных *uint16*.
- **Размер данных сети B**: Размер кадра для передачи по сети B, как число передаваемых байт (+6 байт к исходному размеру за счет RCT), тип данных *uint32*. Не более значения параметра **Максимальный размер сообщения** блока Raw Ethernet Передатчик.
- **Данные сети B**: Сформированный кадр с RCT сети B (Lan ID = 1011), как набор байт, тип данных *uint8*.
- **Номер кадра сети B**: Номер кадра сети B (значение поля SeqNr), тип данных *uint16*.

Блок Отбрасыватель дубликатов PRP-кадров

Блок работает по алгоритму Duplicate Discard, отбрасывая кадр, который пришел позже, в соответствии с протоколом PRP.

Маска блока



Входы блока

- **Размер данных сети A**: Размер принятого блоком Raw Ethernet Приёмник кадра сети A в виде числа байт, тип данных *uint32*.
- **Данные сети A**: Данные принятого блоком Raw Ethernet Приёмник кадр с RCT сети A, как набор байт, тип данных *uint8*.
- **Размер данных сети B**: Размер принятого блоком Raw Ethernet Приёмник кадра сети A в виде числа байт, тип данных *uint32*.
- **Данные сети B**: Данные принятого блоком Raw Ethernet Приёмник кадр с RCT сети B, как набор байт, тип данных *uint8*.

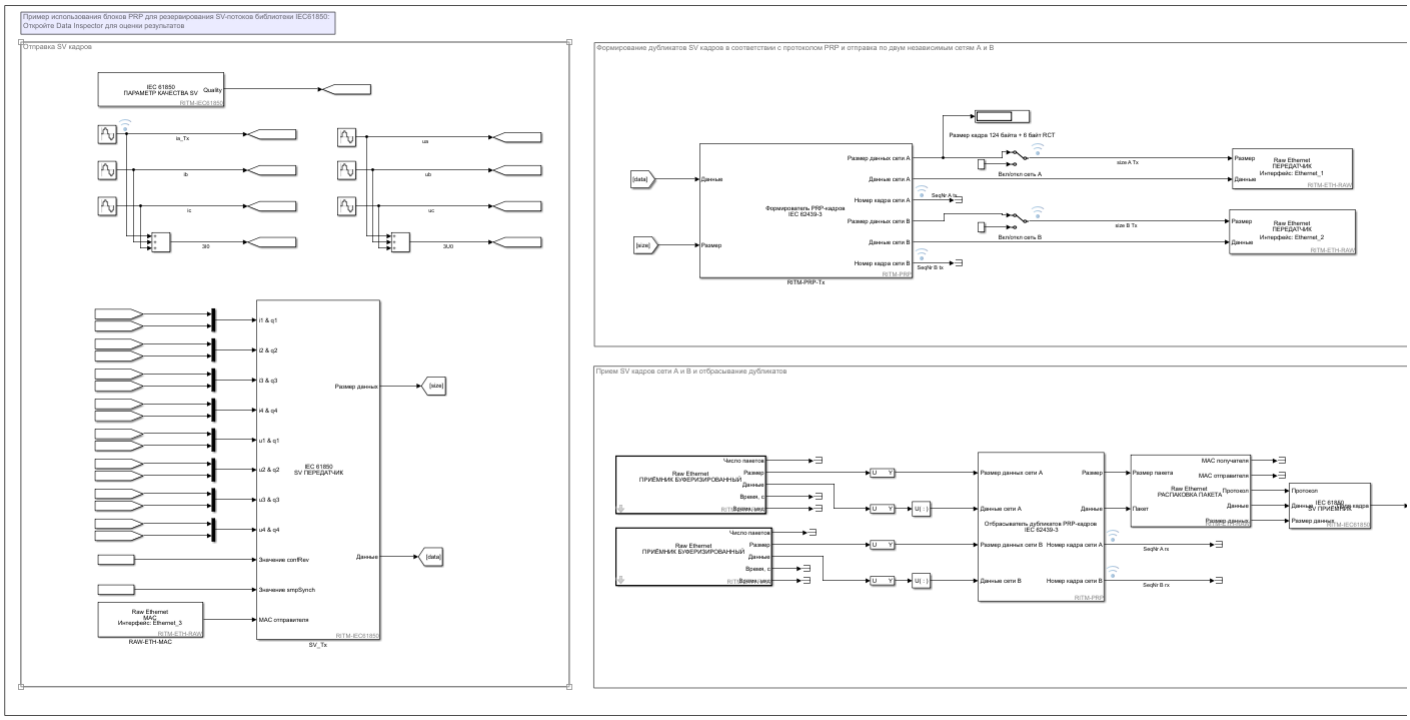
Выходы блока

- **Размер**: Размер кадра после удаления RCT в виде числа байт, тип данных *uint32*.
- **Данные**: Кадр без RCT, как набор байт, тип данных *uint8*.
- **Номер кадра сети A**: Номер кадра сети A (значение поля SeqNr), тип данных *double*.
- **Номер кадра сети B**: Номер кадра сети B (значение поля SeqNr), тип данных *double*.

Примеры и демонстрации RITM PRP

Пример использования блоков PRP для передачи и приема SV-потока.

```
open_system('ritmdemo_block_prp.slx')
```



Блоки поддержки C37.118

Содержание

- Библиотека блоков для работы с C37.118

Библиотека блоков для работы с C37.118

Библиотека блоков для отправки и приема синхронизированных векторных измерений.

Библиотека блоков для работы с С37.118

Содержание

- Описание
- Библиотека блоков
- Блок С37.118 Передатчик
  - Маска блока
  - Входы блока
  - Параметры блока
- Блок С37.118 УСВИ
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок С37.118 Приемник
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок С37.118 Опорный сигнал
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока
- Блок С37.118 Измеритель СВИ
  - Маска блока
  - Входы блока
  - Выходы блока
  - Параметры блока

Описание

Библиотека блоков для отправки синхронизированных векторных измерений (СВИ) и их приема по стандарту С37.118.

Стандарт С37.118 определяет методы обмена данными СВИ в реальном времени между УСВИ и КСВД.

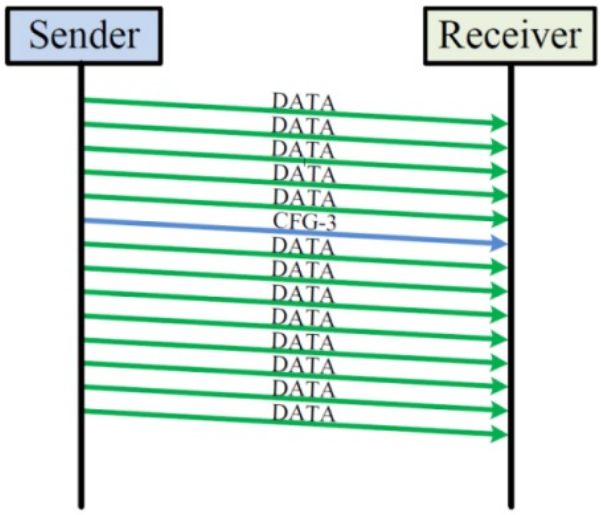
- **Phasor Measurement Unit (PMU)** — это устройство СВИ, которое производит синхронизированные измерения действующих значений и фазового угла токов или напряжений, частоты и скорости изменения частоты (ROCOF) в однозначно определенные моменты времени.
- **Phasor Data Concentrator (PDC)** - это устройство для приема, обработки, хранения и передачи СВИ.

Стандарт С37.118.2 описывает 4 типа кадров: data, configuration, header и command.

- **Кадр с данными (data frame)** - это измерения, выполненные УСВИ.
- **Кадр конфигурации (configuration frame)** - это машиночитаемое сообщение, описывающее типы данных, коэффициенты масштабирования и другие метаданные для data, которые отправляет УСВИ/КСВД.
- **Кадр с заголовком (header frame)** - это удобочитаемая описательная информация, отправляемая из УСВИ/КСВД.

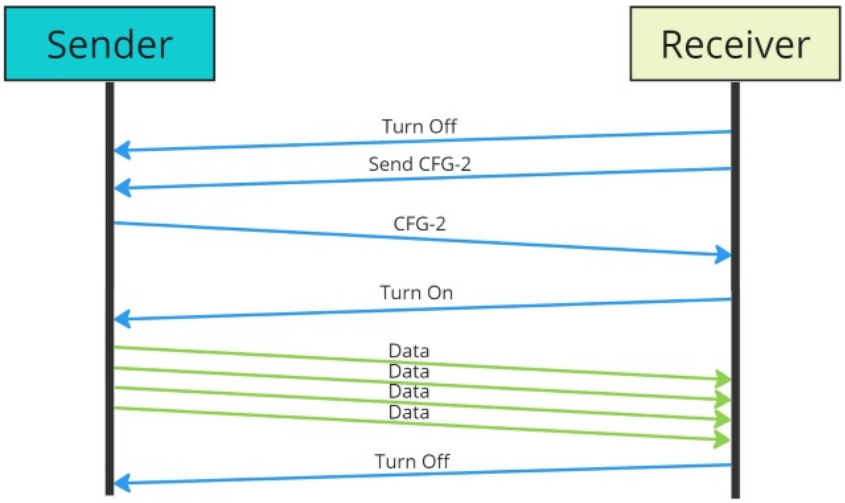
- **Кадр с командой (command frame)** - машиночитаемые коды, отправляемые в УСВИ/КСВД для управления или настройки.

В спонтанном режиме по UDP отправитель обычно непрерывно отправляет кадры с данными получателю.

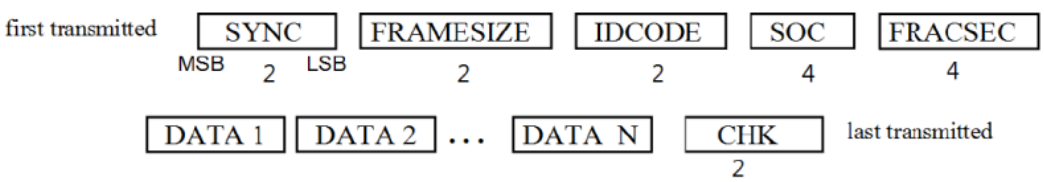


Кадры данных отправляются с частотой отчетов, указанной в настройках. Дополнительно, с заданными пользователем интервалами времени (например, один раз в минуту), отправляется кадр конфигурации CFG-2/CFG-3, чтобы информировать получателя о структуре кадров с данными, которые передаются в настоящее время.

В управляемом режиме все собщения передаются либо по UDP, либо по TCP, либо кадры данных передаются по UDP, а все остальные типы сообщений передаются по TCP.



Все кадры начинаются с 2-байтового SYNC, за которым следует 2-байтовый FRAMESIZE, 2-байтовое IDCODE, временная метка, состоящая из 4-байтовой секунды эпохи SOC и 4-байтовой FRACSEC, которая включает в себя 24-битное целое число и 8-битный флаг качества времени.



Структура кадра данных:

№	Поле	Размер	Комментарий

№	Поле	Размер	Комментарий
1	SYNC	2	Байт синхронизации, за которым следует тип кадра и номер версии. Первый байт: AA hex Второй байт: 01 hex(кадр версии 1, IEEE Std C37.118-2005) 02 hex(кадр версии 2, IEEE Std C37.118-2011)
2	FRAMESIZE	2	Количество байт в кадре, включая CHK. Максимум - 65535.
3	IDCODE	2	Идентификационный номер источника потока, 16-битное целое число, присваиваемое пользователем. Диапазон 1 - 65534, 0 и 65535 - зарезервированы.
4	SOC	4	Метка времени SOC, 32-битное целое число, Unix-время, определяющее время измерения или время передачи для кадров.
5	FRACSEC	4	Доли секунды и качество времени Time Quality. При делении на TIME_BASE из кадра конфигурации получается фактическая доля секунды.
6	STAT	2	Битовые флаги. Бит 15–14: Ошибка данных: 00 = хорошие данные измерения, ошибок нет. 01 = ошибка УСВИ. Нет информации о данных. 10 = УСВИ в тестовом режиме (не использовать значения) или были вставлены отсутствующие теги данных (не использовать значения). 11 = ошибка УСВИ (не использовать значения). Бит 13: синхронизация УСВИ, значение равно 0 при синхронизации с отслеживаемым источником времени UTC. Бит 12: сортировка данных, 0 по отметке времени, 1 по прибытию. Бит 11: обнаружен триггер УСВИ, значение 0, если триггера нет. Бит 10: изменение конфигурации, установлен в 1 на 1 минуту, чтобы сообщить об изменении конфигурации, и очистить до 0, когда изменения вступят в силу. Бит 09: данные изменены, 1, если данные изменены постобработкой, 0 в противном случае. Биты 08–06: качество времени УСВИ. Биты 05–04: время разблокировки: 00 = синхронизация заблокирована или разблокирована < 10 с (наилучшее качество). 01 = 10 с ≤ время разблокировки < 100 с. 10 = 100 с < время разблокировки ≤ 1000 с. 11 = время разблокировки > 1000 с. Биты 03–00: причина срабатывания 1111–1000 - доступно для определения пользователем; 0111 - Цифровой; 0101 - Высокий ROCOF; 0011 - Разница фазового угла; 0001 - Магнитуда низкая; 0110 - Зарезервировано; 0100 - Высокая или низкая частота; 0010 - Высокая магнитуда; 0000 - Вручную.
7	PHASORS	4 × PHNMR или 8 × PHNMR	Векторные измерения. Может быть однофазным или трехфазным, прямой, обратной или нулевой последовательности. 4 или 8 байт в зависимости от используемого формата в поле FORMAT в кадре конфигурации. Количество значений определяется полем PHNMR в кадрах конфигурации.
8	FREQ	2/4	Частота (с фиксированной или плавающей запятой).
9	DFREQ	2/4	ROCOF (с фиксированной или плавающей запятой).
10	ANALOG	2 × ANNMR или 4 × ANNMR	Аналоговые измерения, 2 или 4 байта на значение в зависимости от используемого формата с фиксированной или плавающей запятой, как указано в поле FORMAT в кадре конфигурации.. Количество значений определяется полем ANNMR в кадре конфигурации.
11	DIGITAL	2 × DGNMR	Цифровые данные, обычно представляющие 16 цифровых точек состояния (каналов). Количество значений определяется полем DGNMR в кадре конфигурации.
	Repeat 6–11		Поля 6–11 повторяются для стольких УСВИ, сколько указано в поле NUM_PMU в кадре конфигурации.
	CHK	2	Контрольная сумма CRC-CCITT, 16-разрядное целое число без знака.

Структура кадра конфигурации:

№	Поле	Размер	Краткое описание

№	Поле	Размер	Краткое описание
1	SYNC	2	Первый байт: AA hex. Второй байт: 21 hex для конфигурации 1. 31 hex для конфигурации 2. Оба фрейма относятся к версии 1 (IEEE Std C37.118-2005).
2	FRAMESIZE	2	Количество байт в кадре, включая CHK. Максимум - 65535.
3	IDCODE	2	Идентификационный номер источника потока, 16-битное целое число, присваиваемое пользователе. Диапазон 1 - 65534, 0 и 65535 - зарезервированы.
4	SOC	4	Метка времени SOC, 32-битное целое число, Unix-время, определяющее время измерения или время передачи для кадров.
5	FRACSEC	4	Доли секунды и качество времени Time Quality. При делении на TIME_BASE из кадра конфигурации получается фактическая доля секунды.
6	TIME_BASE	4	Размерность доли секундной метки времени (FRACSEC) во всех кадрах. Биты 31–24: зарезервированы для флагов (старшие 8 бит). Биты 23–0: 24-битное целое число без знака, FRACSEC. Фактическая дробная секунда кадра данных = FRACSEC/TIME_BASE.
7	NUM_PMU	2	Количество УСВИ, включенных в кадр данных. Лимит не указан. Фактическое ограничение будет определяться ограничением в 65 535 байт в одном кадре (поле FRAMESIZE).
8	STN	16	Имя станции – 16 байт в формате ASCII.
9	IDCODE	2	Идентификационный номер источника данных идентифицирует источник каждого блока данных.
10	FORMAT	2	Формат данных в кадре данных, 16-битный флаг. Биты 15–4: не используются Бит 3: FREQ/DFREQ. 0 = 16-битное целое число, 1 = с плавающей запятой Бит 2:ANALOG. 0 = 16-битного целого числа, 1 = с плавающей запятой Бит 1: PHASORS. 0 = 16-битное целое число, 1 = с плавающей запятой Бит 0: Представление PHASORS. 0 = комплексный вид (rectangular), 1 = полярный вид (polar).
11	PHNMR	2	Количество PHASOR — 2-байтовое целое число.
12	ANNMR	2	Количество ANALOG — 2-байтовое целое число.
13	DGNMR	2	Количество DIGITAL – 2-байтовое целое число.
14	CHNAM	16 × (PHNMR + ANNMR + 16 × DGNMR)	Название PHASOR, ANALOG и DIGITAL — 16 байт для каждого канала (16 каналов в каждом DIGITAL) в формате ASCII в том же порядке, в котором они передаются.
15	PHUNIT	4 × PHNMR	Коэффициент преобразования для PHASOR.
16	ANUNIT	4 × ANNMR	Коэффициент преобразования для ANALOG.
17	DIGUNIT	4 × DGNMR	Поле маски для DIGITAL.
18	FNOM	2	Код номинальной частоты линии (16-битное целое число без знака) Биты 15 –1: зарезервировано. Бит 0: 1 = 50 Гц; 0 = 60 Гц.
19	CFGCNT	2	Счетчик изменений конфигурации.
	Repeat 8–19		Поля 8–19 повторяются для такого количества УСВИ, как в поле 7 (NUM_PMU).
20+	DATA_RATE	2	Скорость передачи данных. Например, DATA_RATE = 15 - это 15 кадров в секунду.

№	Поле	Размер	Краткое описание
21+	CHK	2	Контрольная сумма CRC-CCITT, 16-разрядное целое число без знака.

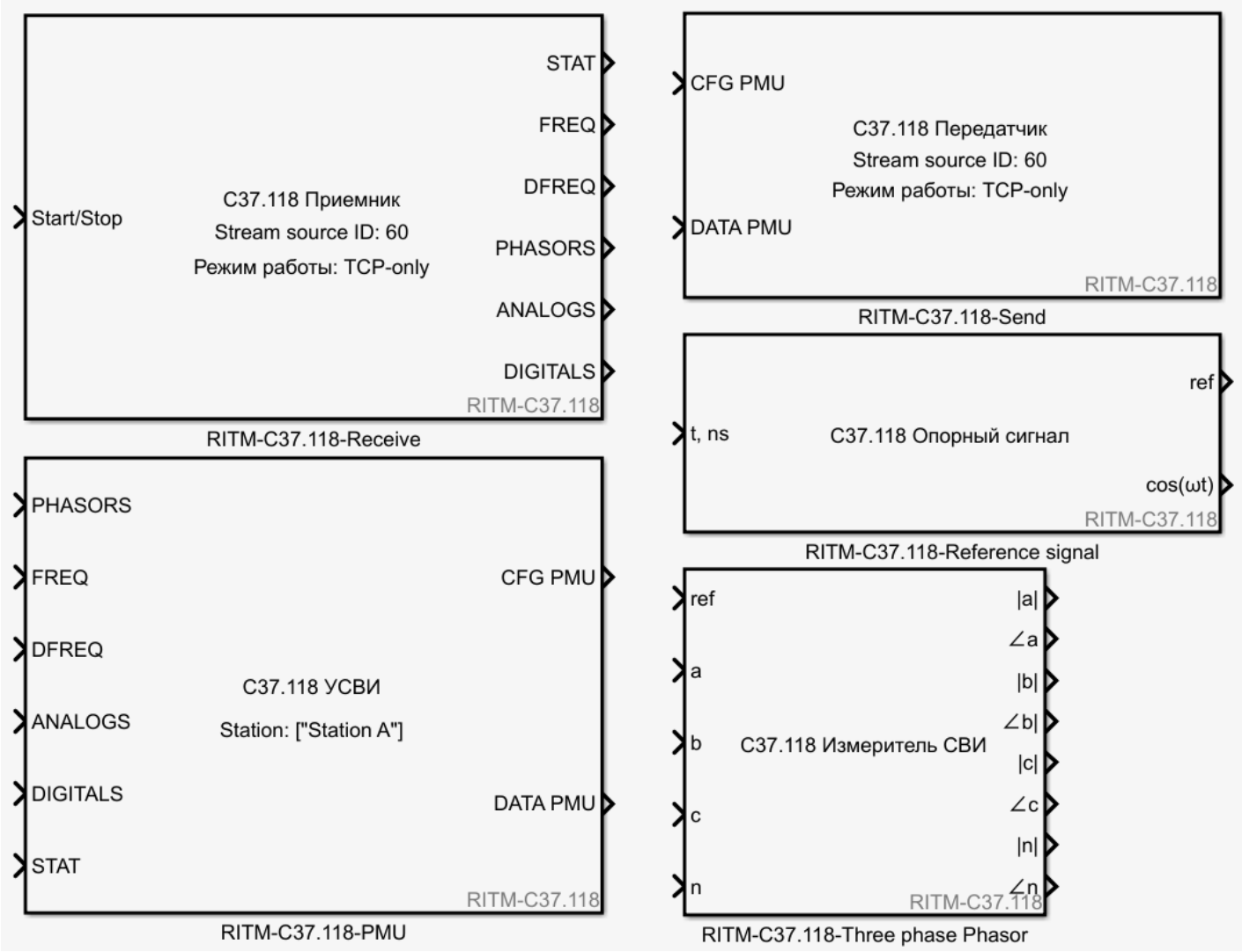
Структура кадра управления:

№	Поле	Размер	Краткое описание
1	SYNC	2	Первый байт: AA hex. Второй байт: 21 hex для конфигурации 1. 31 hex для конфигурации 2. Оба фрейма относятся к версии 1 (IEEE Std C37.118-2005).
2	FRAMESIZE	2	Количество байт в кадре, включая CHK. Максимум - 65535.
3	IDCODE	2	Идентификационный номер источника потока, 16-битное целое число, присваиваемое пользователе. Диапазон 1 - 65534, 0 и 65535 - зарезервированы.
4	SOC	4	Метка времени SOC, 32-битное целое число, Unix-время, определяющее время измерения или время передачи для кадров.
5	FRACSEC	4	Доли секунды и качество времени Time Quality. При делении на TIME_BASE из кадра конфигурации получается фактическая доля секунды.
6	CMD	2	Команда, которую требуется выполнить получателю этой команды: Turn Off, Turn On, Send HDR, Send CFG-1, Send CFG-2, Send CFG-3.
7	ExtFrame	0 - 65518	Расширенные данные кадра.
7+	CHK	2	Контрольная сумма CRC-CCITT, 16-разрядное целое число без знака.

Для получения подробной информации о протоколе IEEE C37.118 посетите страницу IEEE Standard for Synchrophasor Data Transfer for Power Systems.

Перед использованием необходимо произвести настройку Ethernet портов КПМ РИТМ, как описано в разделе РИТМ.Управление машинами -> Настройка сетевых интерфейсов.

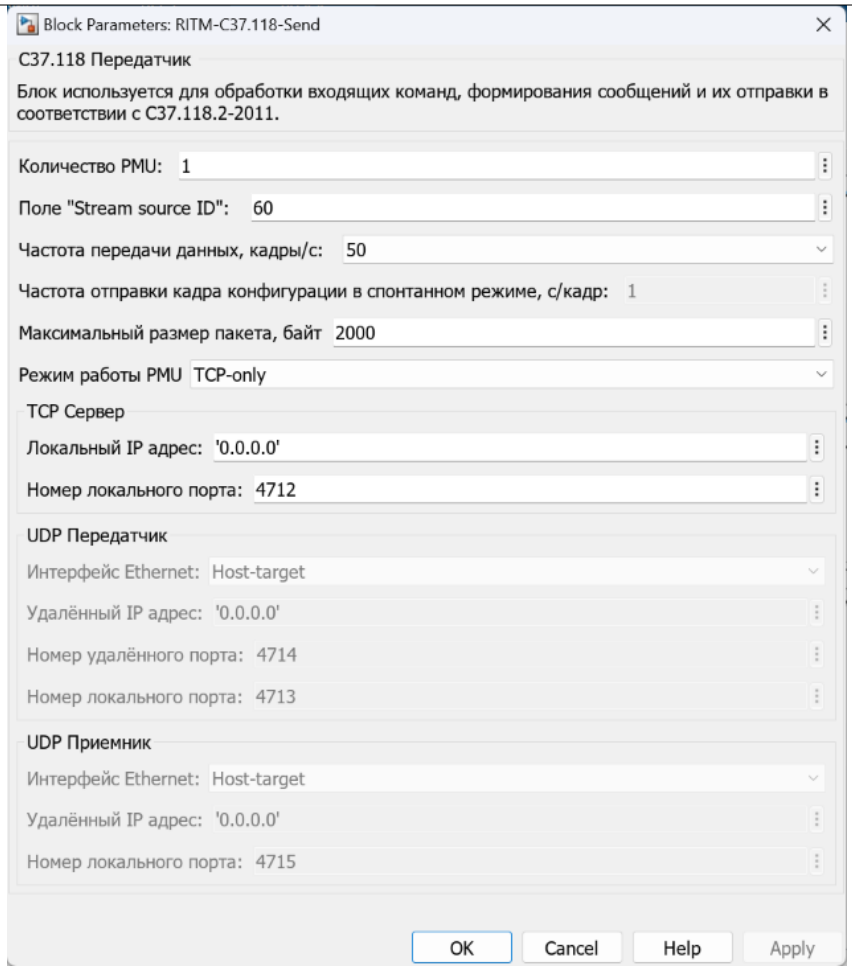
Библиотека блоков



Блок C37.118 Передатчик

Блок используется для приема и обработки команд и отправки сообщений в соответствии с IEEE C37.118.2-2011. Входы блока подключаются к выходам блока C37.118 УСВИ. Если используются несколько блоков C37.118 УСВИ, то выходные данные от них объединяются в нужной последовательности в один вектор с помощью блока *Mux* и заводятся на одноименные входы блока C37.118 Передатчик. Блок поддерживают передачу данных в следующих режимах: *Spontaneous data transmission by UDP, UDP-only, TCP-only, TCP/UDP*. Блок может обрабатывать и отвечать на следующие типы команд: Turn On, Turn Off, Send CFG-2.

Маска блока



Входы блока

- **CFG PMU**: Кадр конфигурации из блока C37.118 УСВИ для добавления в существующий поток, как число передаваемых байт, тип данных *uint8*.
- **DATA PMU**: Кадр данных из блока C37.118 УСВИ для добавления в существующий поток данных, как число передаваемых байт, тип данных *uint8*.

Параметры блока

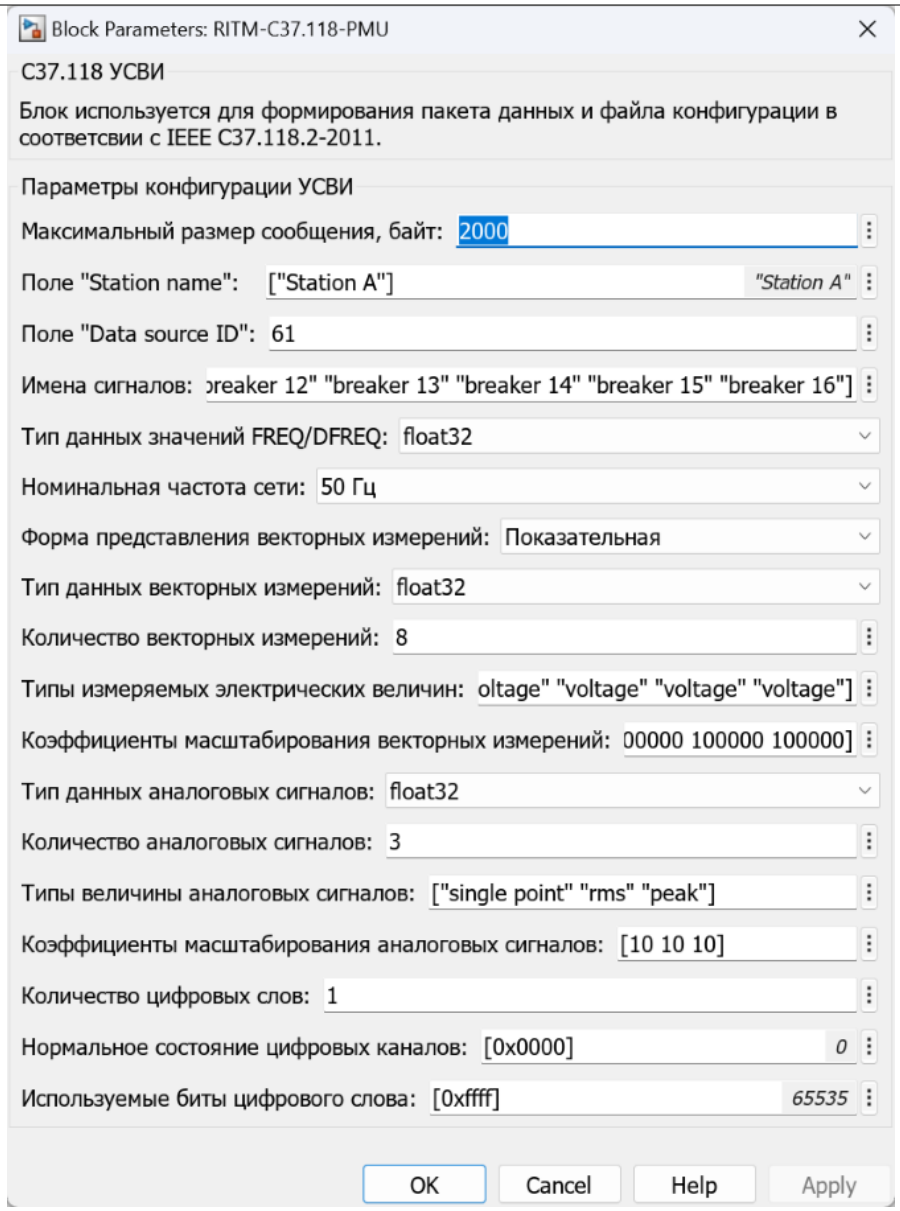
Название параметра	**Описание**
Количество УСВИ	Укажите количество УСВИ в потоке. *Лимит от 1 до фактического ограничения, которое будет определяться размером в 65 535 байт одного кадра.*
Поле “Stream source ID”	Укажите идентификатор потока данных.
Частота передачи данных, кадры/с	Укажите частоту отправки кадров с данными, измеряемую в кадрах в секунду.
Частота отправки кадра конфигурации в спонтанном режиме, с/кадр	Укажите количество секунд между отправкой кадра конфигурации в спонтанном режиме работы.
Максимальный размер пакета	Максимальный размер кадра для передачи по TCP/UDP.
Режим работы УСВИ	Выберите один из предусмотренных стандартом IEEE C37.118.2-2011 режимов работы УСВИ (Spontaneous data transmission by UDP, UDP-only, TCP-only, TCP/UDP) и выставьте соответсвующие настройки в доступных полях.

Название параметра	Описание
TCP-Сервер	**Локальный IP-адрес:** IPv4-адрес заданный устройству, который будет прослушиваться сервером. **Номер локального порта:** Порт, который будет прослушиваться сервером.
UDP-Передатчик	**Интерфейс Ethernet:** Порт Ethernet, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе “КПМ РИТМ”. **Удаленный IP-адрес:** IPv4 адрес получателя сообщений. IP адрес, равный 0.0.0.0, предназначен для отправки сообщений на локальную машину. **Номер удаленного порта:** Порт удаленного получателя, на который будет происходить отправка сообщений. **Номер локального порта:** Локальный порт UDP, с которого будет производиться отправка сообщений. При заданном в маске нулевом значении, номер порта будет определен операционной системой самостоятельно из доступных.
UDP-Приемник	**Интерфейс Ethernet:** Порт Ethernet, который будет использоваться для приема сообщений. Номера портов указаны на корпусе КПМ “РИТМ”. **Удаленный IP-адрес:** IPv4 адрес отправителя сообщений. IP адрес, равный 0.0.0.0, предназначен для приема сообщений от любого отправителя. **Номер локального порта:** Порт, который будет прослушиваться приёмником

Блок C37.118 УСВИ

Блок создает новый поток данных СВИ и файл конфигурации устройства, и пакует выходные данные в два вектора типа uint8 в соответствии с IEEE C37.118.2-2011. Выходы блока подключаются к одноименным входам блока C37.118 Передатчик. Если используются несколько блоков C37.118 УСВИ, то их выходы объединяются в один вектор с помощью блока *Mux* и подключаются к тем же входам блока C37.118 Передатчик.

Маска блока



Входы блока

- **PHASORS**: Векторные измерения в показательной форме: один вектор - действующее значение сигнала и фазовый угол синхронизированный с ref в радианах, тип данных *double*. Если определены два вектора, значит четыре значения должны быть определены по порядку [Ph1 mag, Ph1 ang, Ph2 mag, Ph2 ang] с помощью блока Mux.
- **FREQ**: Значение частоты, тип данных *double*.
- **DFREQ**: Значение производной частоты, тип данных *double*.
- **ANALOGS**: Аналоговые измерения, тип данных *double*. Если определены три аналоговых сигнала, значения должны быть определены по порядку [An1 An2 An3] с помощью блока *Mux*.
- **DIGITALS**: Дискретные сигналы, тип данных *double* которые складываются в Цифровое слово. Дискретные сигналы подаются на вход DIGITALS по порядку с помощью блока *Mux*. Количество сигналов может быть меньше 16 сигналов для каждого цифрового слова. Не используемые биты цифрового слова будут установлены в ноль. Если определены два цифровых слова и более, то значения будут складываться по порядку от младшего бита первого слова к старшему биту последнего.
- **STAT**: Значение поля STAT, тип данных *uint16*.

Выходы блока

- **CFG PMU**: Кадр конфигурации для добавления в существующий поток, как число передаваемых байт, тип данных *uint8*.
- **DATA PMU**: Кадр данных для добавления в существующий поток данных, как число передаваемых байт, тип данных *uint8*.

Параметры блока

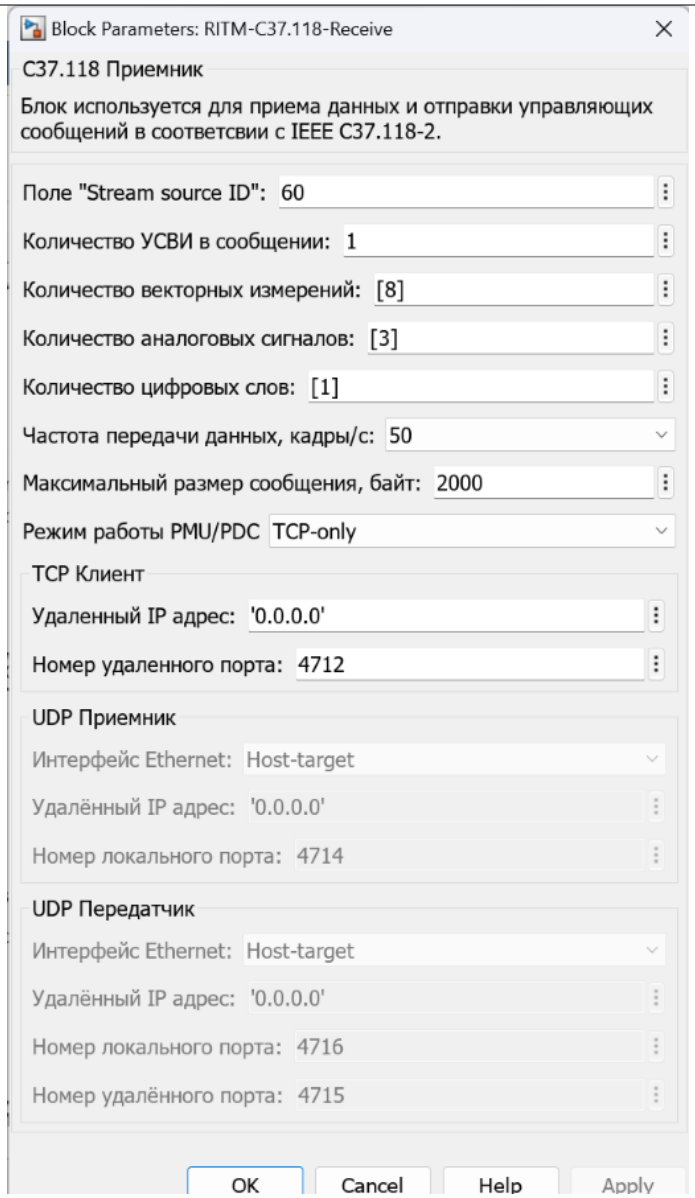
Название параметра	Описание
Поле “Максимальный размер сообщения, байт”:	Максимальный размер кадра для передачи для данного УСВИ по TCP/UDP.
Поле “Station name”:	Укажите имя станции УСВИ. Например: *[“Station A”].
Поле “Data source ID”:	Укажите идентификатор исходного источника данных, который определяет данные от конкретного УСВИ в общем потоке данных.
Имена сигналов:	Укажите массив названий сигналов по порядку. Например: *[“Ua” “Ub” “Uc” “analog1” “analog2” “analog3” “breaker 1” “breaker 2” “breaker 3” “breaker 4” “breaker 5” “breaker 6” “breaker 7” “breaker 8” “breaker 9” “breaker 10” “breaker 11” “breaker 12” “breaker 13” “breaker 14” “breaker 15” “breaker 16”].*
Тип данных значений FREQ/DFREQ:	Укажите тип данных для FREQ и DFREQ. integer - целое число, float - число с плавающей точкой.
Номинальная частота сети:	Выберите номинальную частоту сети 50 или 60 Гц.
Форма представления векторных измерений:	Векторные измерения могут быть представлены в показательной форме (polar) и в алгебраической (rectangular).
Тип данных векторных измерений:	Векторные измерения могут быть указаны в виде чисел с плавающей точкой.
Количество векторных измерений:	Количество векторных измерений в УСВИ. Если указать количество векторных измерений - 0, то данные на входе будут просто игнорироваться.
Типы измеряемых электрических величин:	Укажите массив типов измеряемых электрических величин. Напряжение - “voltage”, ток - “current”. Например: *[“voltage” “voltage” “voltage” “current” “current” “current”]*.
Коэффициенты масштабирования векторных измерений:	Укажите массив коэффициентов масштабирования векторных измерений.
Тип данных аналоговых сигналов:	Выберите между целым числом или типом с плавающей запятой для аналоговых данных.
Количество аналоговых сигналов	Укажите количество измеряемых аналоговых сигналов. Если указать количество аналоговых сигналов - 0, то данные на входе будут просто игнорироваться.
Тип величины аналоговых сигналов	Определите тип аналогового сигнала. Тип сигнала может быть мгновенным значением - “single point”, среднеквадратичным - “rms”, амплитудным - “peak”. Например, *[“single point” “rms” “peak”]*.
Коэффициенты масштабирования аналоговых сигналов:	Укажите массив коэффициентов масштабирования аналоговых сигналов.

Название параметра	Описание
Количество цифровых слов:	Укажите количество цифровых слов (поле DGNMR). Если указать количество цифровых слов - 0, то данные на входе будут просто игнорироваться.
Нормальное состояние цифровых каналов:	Укажите нормальное рабочее состояние цифровых каналов. Если передаются несколько цифровых слов, то укажите массив чисел в формате *uint16*.
Используемые биты цифрового слова:	Укажите используемые цифровые каналы каждого цифрового слова в формате *uint16*.

Блок C37.118 Приемник

Блок извлекает СВИ из полученных кадров данных, которые доступны в виде сигнала на выходном порту. Выходные порты соединяются с блоком Demux, чтобы демультиплексировать измерения. Блок содержит данные, считанные из последнего кадра данных.
Блок поддерживает все режимы работы, описанные в стандарте IEEE C37.118.2-2011: *Spontaneous data transmission by UDP, UDP-only, TCP-only, TCP/UDP.*
Управление работой блока осуществляется сигналом на входе “Start/Stop”. Блок не применяет параметры конфигурации из файла CFG-2 автоматически. Для корректного извлечени СВИ, необходимо вручную заполнить все поля в маске блока C37.118 Приемник.
Блок принимает данные векторных измерений, аналоговых сигналов, частоту и скорость изменения частоты только в формате float32.

Маска блока



!

Входы блока

- **Start/Stop**: Осуществляет управление блоком. На входе принимаются значения ‘1’ - Start и ‘0’ - Stop. В одном из управляемых режимов работы устройства, управление осуществляется по входу “Start/Stop”. При изменении сигнала с ‘0’ на ‘1’ на входе блока, сначала отправляется команда Turn Off, затем запрашивается файл конфигурации Send CFG-2 и после получения CFG-2, отправляется команда Turn On. При изменении сигнала с ‘1’ на ‘0’ отправляется команда Turn Off. В спонтанном режиме работы блок начинает сразу принимать данные, независимо от состояния входа “Start/Stop”. Так же в спонтанном режиме работы блок не отправляет никаких команд.

Выходы блока

- **STAT**: Значение поля STAT, тип данных *uint16*. *Если получены измерения с двух УСВИ, значит два значения будут демультиплексированы по порядку [St1, St2] с помощью блока Demux.*
- **FREQ**: Значение частоты, тип данных *single*. *Если получены измерения с двух УСВИ, значит два значения будут демультиплексированы по порядку [F1, F2] с помощью блока Demux.*
- **DFREQ**: Значение производной частоты, тип данных *single*. *Если получены измерения с двух УСВИ, значит два значения будут демультиплексированы по порядку [Df1, Df2] с помощью блока Demux.*
- **PHASORS**: Векторные измерения в показательной форме: один вектор - действующее значение сигнала и фазовый угол синхронизированный с ref в радианах, тип данных *single*. *Если получено два вектора, значит четыре значения будут демультиплексированы по порядку [Ph1 mag, Ph1 ang, Ph2 mag, Ph2 ang] с помощью блока Demux.*
- **ANALOGS**: Аналоговые измерения, тип данных *single*. *Если получены три аналоговых сигнала, значения должны будут демультиплексированы по порядку [An1 An2 An3] с помощью блока Demux.*

- **DIGITALS**: Цифровое слово, тип данных *uint16. Если получено два цифровых состояния, значения должны демультиплексированы по порядку [Dg1 Dg2] с помощью блока Demux.*

Выходные значения FREQ, DFREQ, PHASORS и ANALOGS всегда относятся к типу с плавающей точкой.

Параметры блока

Название параметра	**Описание**
Поле “Stream source ID”:	Укажите идентификатор потока.
Количество УСВИ в сообщении:	Укажите количество векторных УСВИ в потоке. Не менее 1.
Количество векторных измерений:	Укажите количество векторных измерений в УСВИ. *Если количество УСВИ равно 3, то количество векторных измерений должно быть задано в виде массива, например: [6, 0, 3]. Соответственно, в параметрах Demux, который подключается к выходному порту PHASORS, следует указать число выходных сигналов 18. Тогда измерения будут выведены в следующем порядке: [Ph1 mag, Ph1 ang, Ph2 mag, Ph2 ang, Ph3 mag, Ph3 ang, Ph4 mag, Ph4 ang, Ph5 mag, Ph5 ang, Ph6 mag, Ph6 ang, Ph7 mag, Ph7 ang, Ph8 mag, Ph8 ang, Ph9 mag, Ph9 ang], где первые шесть векторов относятся к УСВИ 1, а последние три к УСВИ 3. В случае, если ни в одном из УСВИ нет векторных измерений, например [0, 0, 0], то на выходе блока будет один сигнал с нулевым значением.*
Количество аналоговых сигналов:	Укажите количество аналоговых измерений в УСВИ. *Если количество УСВИ равно 2, то количество аналоговых измерений должно быть задано в виде массива, например: [1, 2]. Соответственно, в параметрах Demux, который подключается к выходному порту ANALOGS, следует указать число выходных сигналов 3. Тогда измерения будут выведены в следующем порядке: [An1, An2, An3], где первое аналоговое измерение относятся к УСВИ 1, а последние два к УСВИ 2. В случае, если ни в одном из УСВИ нет аналоговых измерений, например [0, 0], то на выходе блока будет один сигнал с нулевым значением.*

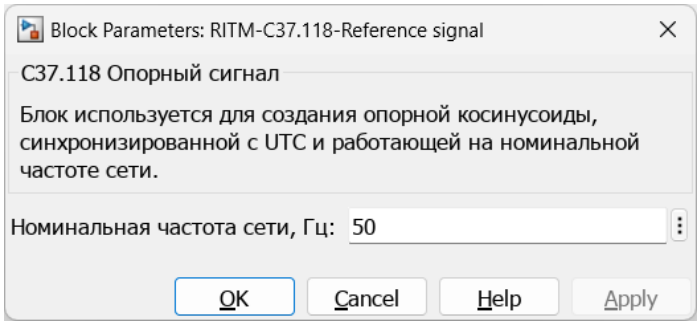
Название параметра	Описание
Количество цифровых слов:	Укажите количество цифровых слов в УСВИ. *Если количество УСВИ равно 2, то количество цифровых слов должно быть задано в виде массива, например: [2, 1]. Соответственно, в параметрах Demux, который подключается к выходному порту* DIGITALS*, следует указать число выходных сигналов 3. Тогда измерения будут выведены в следующем порядке: [Dg1, Dg2, Dg3], где первые два цифровых состояния относятся к* УСВИ 1*, а последний к УСВИ 2. В случае, если ни в одном из УСВИ нет цифровых слов, например [0, 0], то на выходе блока будет один сигнал с нулевым значением.*
Частота передачи данных, кадры/с:	Укажите частоту отправки кадров с данными источника СВИ, измеряемую в кадрах в секунду.
Максимальный размер сообщения, байт:	Максимальный размер кадра для приема по TCP/UDP.
Режим работы УСВИ/КСВД:	Выберите один из предусмотренных стандартом IEEE C37.118.2-2011 режимов работы УСВИ/КСВД (Spontaneous data transmission by UDP, UDP-only, TCP-only, TCP/UDP) и выставьте соответствующие настройки в доступных полях.
TCP Клиент:	**Удаленный IP адрес**: IPv4-адрес заданный устройству, к которому будет производиться подключение. **Номер удаленного порта**: Порт, к которому будет производиться подключение.
UDP Приемник:	**Интерфейс Ethernet:** Порт Ethernet, который будет использоваться для приема сообщений. Номера портов указаны на корпусе КПМ РИТМ. **Удаленный IP-адрес:** IPv4 адрес отправителя сообщений. IP адрес, равный 0.0.0.0, предназначен для приема сообщений от любого отправителя. **Номер локального порта:** Порт, который будет прослушиваться приёмником.

Название параметра	Описание
UDP Передатчик:	**Интерфейс Ethernet:** Порт Ethernet, который будет использоваться для передачи сообщений. Номера портов указаны на корпусе КПМ РИТМ. **Удаленный IP-адрес:** IPv4 адрес получателя сообщений. IP адрес, равный 0.0.0.0, предназначен для отправки сообщений на локальную машину. **Номер локального порта:** Локальный порт UDP, с которого будет производиться отправка сообщений. При заданном в маске нулевом значении, номер порта будет определен операционной системой самостоятельно из доступных. **Номер удаленного порта:** Порт удаленного получателя, на который будет происходить отправка сообщений.

Блок С37.118 Опорный сигнал

Блок создает опорный сигнал функции косинуса на номинальной системной частоте, синхронизированной с UTC. Фазовый угол равен 0 радиан в момент смены секунды. Блок устанавливается после Блока РИТМ Unix Время, получая от него на вход сигнал **Наносекунды**.

Маска блока



Входы блока

- **t, ns**: Целое число наносекунд, тип данных *uint64*, получаемые от Блока РИТМ Unix Время.

Выходы блока

- **ref**: Значение фазового угла опорной косинусоиды, тип данных *double*. Используется в качестве опорного сигнала для вычисления фазового угла передаваемых СВИ в блоке Измеритель СВИ.
- **cos(ωt)**: Значение функции косинуса, тип данных *double*.

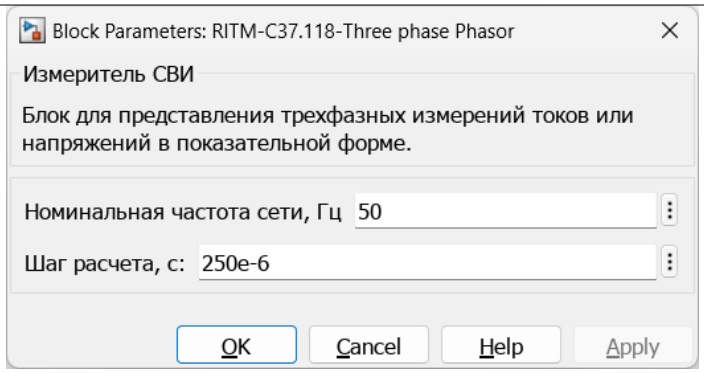
Параметры блока

- **Номинальная частота сети, Гц**: Укажите номинальную частоту косинусоиды.

Блок С37.118 Измеритель СВИ

Блок используется для представления трехфазных измерений токов или напряжений в полярных координатах. Фаза рассчитывается относительно опорного сигнала из блока С37.118 Опорный сигнал. Полученные действующие значения и фазовые углы соединяются в блоке *Mux* и передаются на блок С37.118 УСВИ.

Маска блока



Входы блока

- **ref**: Значение фазового угла опорной косинусоиды, тип данных *double*, полученное от блока C37.118 Опорный сигнал. Используется в качестве опорного сигнала для вычисления фазового угла передаваемых СВИ.
- **a, b, c, n**: Измерения фаз A, B, C и нулевой последовательности, тип данных *double*. На самом деле, вы можете завести на входы любые необходимые вам измерения, они будут разложены на вектор в показательной форме на соответствующих выходах.

Выходы блока

- **|a|, |b|, |c|, |n|**: Амплитудное значение соответствующего сигнала на входе, тип данных *double*. Используется в качестве опорного сигнала для вычисления фазового угла передаваемых СВИ.
- **∠a, ∠b, ∠c, ∠n**: Значение фазы , тип данных *double*.

Параметры блока

- **Номинальная частота сети, Гц**: Укажите номинальную частоту косинусоиды в герцах.
- **Шаг расчета, с**: Укажите шаг расчета блока в секундах.

Примеры и демонстрации РИТМ С37.118

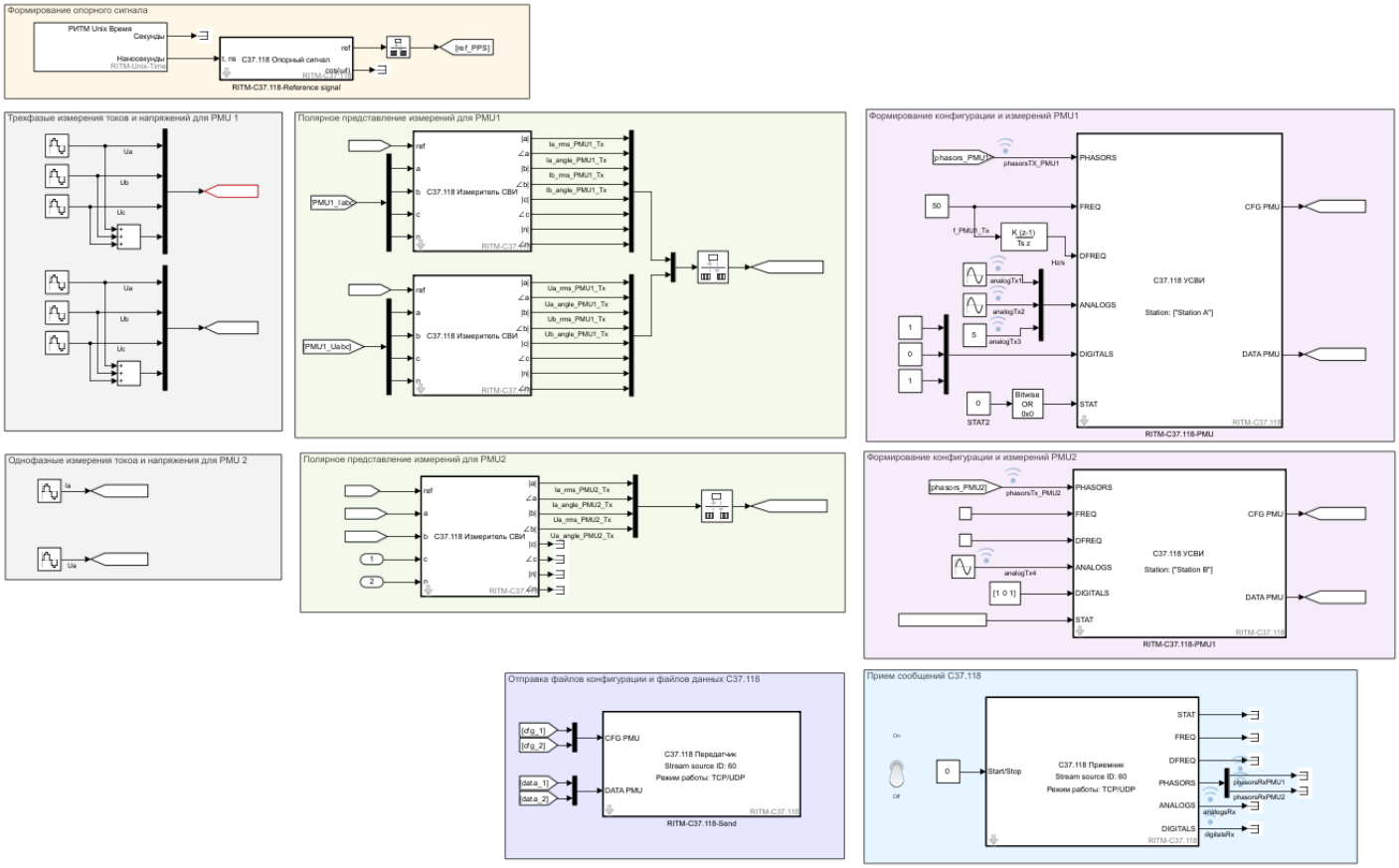
Содержание

- Передача и прием синхронизированных векторных измерений (СВИ) по UDP

Передача и прием синхронизированных векторных измерений (СВИ) по UDP

Откройте модель.

```
open_system('ritmdemo_block_c37_118.slx')
```



В полях серого цвета содержатся блоки Sin Wave, которые моделируют измерения мгновенных значений тока и напряжения.

Далее, следуют поля зеленого цвета, которые переводят мгновенные значения в полярный вид. Для расчета фазового угла относительно внутренних часов КПМ РИТМ, выходной порт ref блока С37.118 Опорный сигнал подключен к входному порту блока С37.118 Измеритель СВИ.

Векторные измерения в полярном виде из блоков “С37.118 Измеритель СВИ” подключаются к блокам “С37.118 УСВИ”, которые создают новые файлы данных и файлы конфигурации.

Выходные порты блоков “С37.118 УСВИ” подключаются к блоку “С37.118 Передатчик”, в котором устанавливается IP адрес и соответствующие порты для передачи СВИ.

Если закомментировать блок модели “Прием сообщений С37.118” и выставить правильные настройки внутри блока С37.118 Передатчик, то можно взаимодействовать с блоком через программу PMU Connection Tester и увидеть все сообщения в WireShark.

Прием СВИ осуществляется с помощью блока “С37.118 Приемник”. Управление блоком осуществляется по входу “Start/Stop”. В данной демо-модели сигнал на вход подается из константы, которая, для удобства, привязана к дашборду “On/Off”. Используйте Data Inspector для просмотра значений СВИ, переданных из верхней части модели.

Блоки для моделирования электрических машин

Содержание

- Библиотека блоков для моделирования электрических машин

Библиотека блоков для моделирования электрических машин

Библиотека блоков с дополнительными блоками для Simscape Electrical для физического моделирования электрических машин.

Блоки для моделирования электрических машин

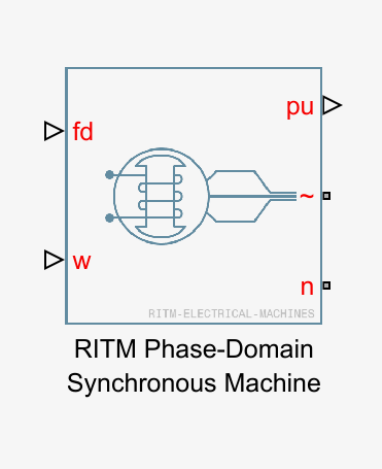
Содержание

- Описание
- Библиотека блоков
- Блок RITM Phase-Domain Synchronous Machine
  - Маска блока
  - Порты
  - Параметры блока
  - Параметры вкладки “Main”:
  - Параметры вкладки “Impedances”:
  - Параметры вкладки “Time Constants”:
  - Параметры вкладки “Mechanical”:
  - Параметры вкладки “Faulted Windings”:
  - Параметры вкладки “Saturation”:
  - Параметры вкладки “Initial Conditions”:
- Список литературы

Описание

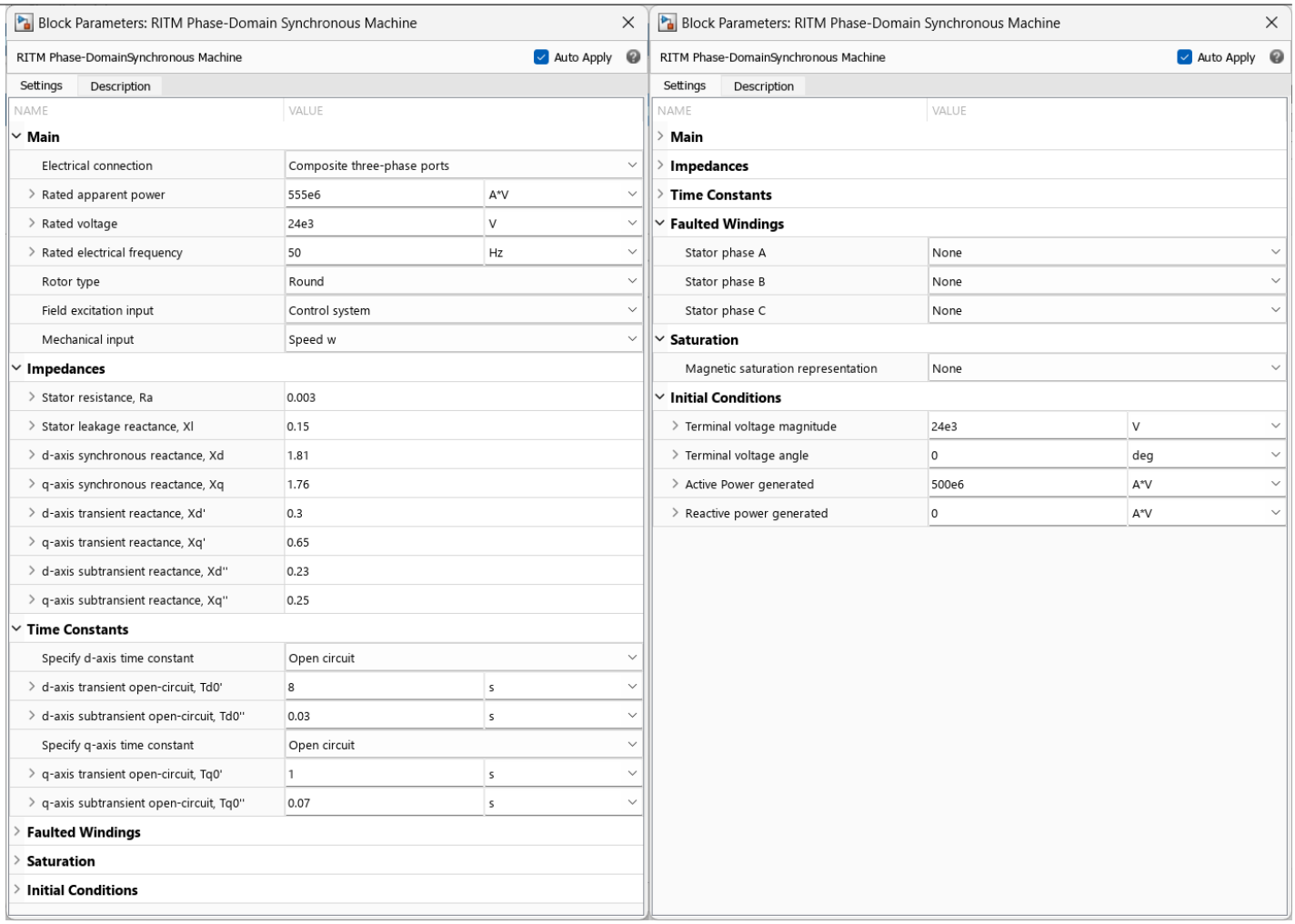
В данном разделе приводится информация о блоках для физического моделирования электрических машин.

Библиотека блоков



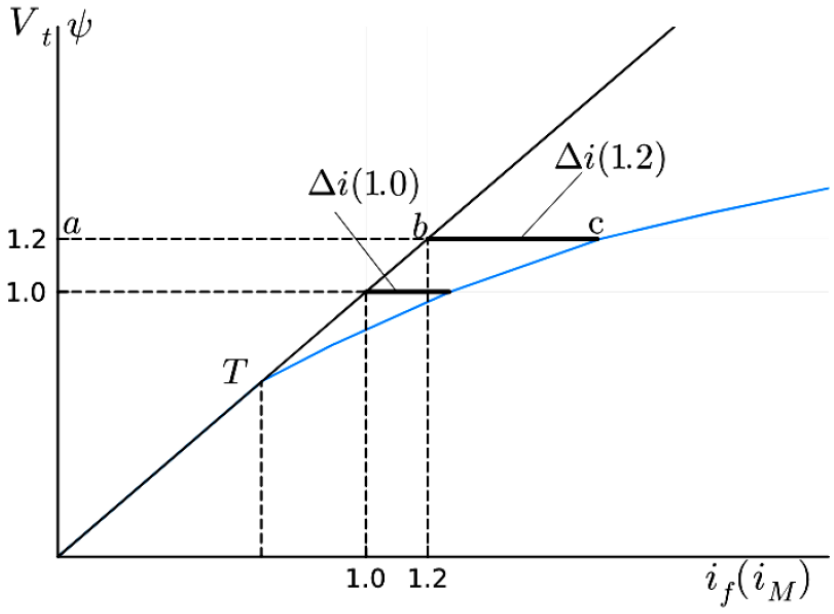
Блок RITM Phase-Domain Synchronous Machine

Маска блока



Блок RITM Phase-Domain Synchronous Machine моделирует синхронную машину в фазных координатах с внутренними повреждениями. Можно использовать эту машину для моделирования внутренних повреждений, таких как межвитковые замыкания, междуфазные замыкания, замыкания на землю как в обмотках статора, так и обмотке возбуждения. Количество одновременно выбираемых точек повреждения составляет до двух на каждую из обмоток машины.
Для реализации этой модели используется метод фазных координат [1]: значения индуктивностей машины зависят от положения ротора и уровня насыщения. Математическая модель в фазных координатах позволяет имитировать внутренние повреждения синхронных машин. Для моделирования внутренних повреждений блок синхронной машины способен вычислить само- и взаимоиндукцию обмоток, включая неисправные обмотки, как функции положения ротора и насыщения.
При моделировании эффекта насыщения и принятии стандартных допущений, в моделях машин RSCAD и библиотеки Simscape Electrical, имеем следующее:
- Насыщению подвергаются только индуктивности намагничивания *Lad* и *Laq*; - Насыщение определяется потокосцеплением в воздушном зазоре; - Характеристика насыщения соответствует кривой насыщения при разомкнутой обмотке статора; - Перекрестное насыщение отсутствует.

В модели RITM Phase-Domain Synchronous Machine моделирование насыщения реализовано, как функция намагничивания по продольной оси по аналогии с RSCAD, а также по двум осям по аналогии с Simscape Electrical.
Задание параметров намагничивания возможно двумя способами:
- По точкам кривой насыщения характеристики холостого хода синхронной машины $V_t = g(i_f)$, где V_t - напряжение на выводах генератора в режиме холостого хода в о.е., i_f - ток возбуждения в о.е.
- С использованием нормализованной характеристики холостого хода, рисунок ниже. До точки T насыщение отсутствует, характеристика линейная, исходящая из начала координат под углом 45˚. После точки T характеристика холостого хода определяется функцией насыщения от потокосцепления (напряжения), задающей смещение кривой вправо относительно прямой, соответствующей линейной магнитной системе. Это смещение представляет собой величину Δi на которую необходимо увеличить ток возбуждения, чтобы получить заданное напряжение на выводах генератора.



В блоке реализован функционал расчета базовых и начальных параметров машины. На основе значений во вкладке **Initial Conditions** блок машины рассчитывает начальные параметры для обмотки возбуждения и входные значения механического вращения, необходимые для достижения установившегося режима. Такой механизм сокращает нежелательные переходные процессы при моделировании. Для получения этих данных необходимо вызвать контекстное меню. Нажмите правой кнопкой мыши на блок, выберете поле Electrical, а затем один из вариантов: Display Base Values, Display Associated Base Values, Display Associated Initial Conditions. Подробнее читайте в разделе документации Machine Plotting and Display Options основной документации MATLAB.

Порты

Порты блока являются ненаправленными Simscape соединениями и подчиняются всем правилам построения физических моделей технических систем с помощью блоков библиотек Simscape, в том числе Simscape Electrical.

- **pu**: Порт, выводящий текущие значения переменных машины:
  Напряжение возбуждения (база цепи возбуждения, *Efd*), *pu_fd_Efd*;
  Ток возбуждения (база цепи возбуждения, *Ifd*), *pu_fd_Ifd*;
  Электромагнитный момент, *pu_torque*; Частота вращения ротора, *pu_velocity*;
  Напряжение статора по оси *d*, *pu_ed*;
  Напряжение статора по оси *q*, *pu_eq*;
  Напряжение нулевой последовательности статора, *pu_e*0;
  Ток по оси *d* статора, *pu_id*;
  Ток по оси *q* статора, *pu_iq*;
  Ток нулевой последовательности статора *pu_i*0;
  Электрический угол ротора, *electrical_angle_out*;
  Для подключения к этому порту используйте блок Synchronous Machine Measurement библиотеки Simscape Electrical. Обратите внимание, что данный блок производит дополнительные вычисления, которые могут понадобиться для моделирования.
- **~**: Электрический порт, расширяемый трехфазный порт, связанный с обмотками статора.
- **n**: Электрический порт, связанный в точке обмотки с нейтралью по типу Звезда.
- **A1-2**: Электрические порты, связанные с внутренними узлами обмотки фазы A.
- **B1-2**: Электрические порты, связанные с внутренними узлами обмотки фазы B.
- **C1-2**: Электрические порты, связанные с внутренними узлами обмотки фазы C.
- **F1-2**: Электрические порты, связанные с внутренними узлами обмотки возбуждения.
- **Efd**: Порт для прямого регулирования внешним сигналом в о.е. напряжения на обмотки возбуждения машины.
- **fd+**: Электрический порт, связанный с положительным выводом обмотки возбуждения.
- **fd-**: Электрический порт, связанный с отрицательным выводом обмотки возбуждения.
- **Pm**: Порт для прямого регулирования внешним сигналом в о.е. механической мощности машины.
- **w**: Порт для прямого регулирования внешним сигналом в о.е. скорости вращения машины.
- **R**: Механический порт, связанный с ротором машины.
- **C**: Механический порт, связанный с корпусом машины.

Параметры блока

Параметры вкладки “Main”:

- **Rated apparent power**: Номинальная полная мощность;
- **Rated voltage**: Номинальное среднеквадратичное линейное напряжение;
- **Rated electrical frequency**: Номинальная электрическая частота, при которой указывается номинальная полная мощность;
- **Rotor type**: Выбор типа ротора синхронной машины: явнополюсный (Salient) или неявнополюсный (Round). В зависимости от типа ротора будет меняться набор параметров для вкладки **Impedances**;
- **Field excitation input**: Выбор типа входного порта обмотки возбуждения:

Control system — создает порт **Efd** для прямого регулирования внешним сигналом напряжения на обмотке возбуждения машины в о.е. С данной настройкой становится недоступным выбор точек для моделирования повреждений в обмотке ротора на вкладке **Faulted Windings**;
Power system port — создает порты **fd+** и **fd-** выводов обмотки возбуждения. С такой настройкой открывается опция моделирования повреждений в обмотке ротора, а также становятся доступны для редактирования параметры: **Field circuit current** и **Field circuit voltage**;

- **Mechanical input**: Выбор типа входного порта управления механическим движением машины:
Speed w — создает порт **w** прямого регулирования внешним сигналом скорости вращения машины в о.е.;
Mechanical power Pm — создает порт **Pm** для прямого регулирования внешним сигналом механической мощности машины в о.е. Данная настройка делает доступной вкладку **Mechanical** для настройки механических параметров машины;
Mechanical rotation port — создает механические порты **R** и **C**.
- **Field circuit current**: Ток цепи возбуждения;
- **Field circuit voltage**: Напряжение цепи возбуждения;

Параметры вкладки “Impedances”:

- **Stator resistance, Ra**: Сопротивление статора в о.е. Этот параметр должен быть больше 0;
- **Stator leakage reactance, Xl**: Реактивное рассеяние статора в о.е. Этот параметр должен быть больше 0;
- **d-axis synchronous reactance, Xd**: Синхронная реактивность по оси d в о.е. Этот параметр должен быть больше 0;
- **q-axis synchronous reactance, Xq**: Синхронная реактивность по оси q в о.е. Этот параметр должен быть больше 0;
- **d-axis transient reactance, Xd’**: Переходная реактивность по оси d в о.е. Этот параметр должен быть больше 0;
- **q-axis transient reactance, Xq’**: Переходная реактивность по оси q в о.е. Этот параметр должен быть больше 0. Данный параметр появляется для выбора параметра **Rotor type** в значение Round;
- **d-axis subtransient reactance, Xd”**: Сверхпереходная реактивность по оси d в о.е. Этот параметр должен быть больше 0;
- **q-axis subtransient reactance, Xq”**: Сверхпереходная реактивность по оси q в о.е. Этот параметр должен быть больше 0.

Параметры вкладки “Time Constants”:

- **Specify d-axis time constant**: Задает способ задания постоянной времени оси d на выбор между *Open circuit* и *Short circuit*. Настройка этого параметра влияет на видимость параметров постоянной времени оси d;
- **d-axis transient open-circuit, Td0’**: Переходная постоянная времени разомкнутой цепи возбуждения по оси d. Этот параметр должен быть: больше 0; больше, чем *d-axis subtransient open-circuit, Td0’’*.
Этот параметр используется, если для параметра *Specify d-axis transient time constant* установлено значение *Open circuit*.
- **d-axis transient short-circuit, Td’**: Переходная постоянная времени замкнутой цепи возбуждения по оси d. Этот параметр должен быть: больше 0; больше, чем *d-axis subtransient short-circuit, Td’’*.
Этот параметр используется, если для параметра *Specify d-axis transient time constant* установлено значение *Short circuit*.
- **d-axis subtransient open-circuit, Td0”**: Сверхпереходная постоянная времени разомкнутой цепи возбуждения по оси d. Этот параметр должен быть больше 0.
Этот параметр используется, если для параметра *Specify d-axis transient time constant* установлено значение *Open circuit*.
- **d-axis subtransient short-circuit, Td”**: Сверхпереходная постоянная времени замкнутой цепи возбуждения по оси d. Этот параметр должен быть больше 0.
Этот параметр используется, если для параметра *Specify d-axis transient time constant* установлено значение *Short circuit*.
- **Specify q-axis time constant** — Задает способ задания постоянной времени оси q на выбор между *Open circuit* и *Short circuit*. Настройка этого параметра влияет на видимость параметров постоянной времени оси q;
- **q-axis subtransient open-circuit, Tq0”**: Сверхпереходная постоянная времени разомкнутой цепи возбуждения по оси q. Этот параметр должен быть больше 0.
Этот параметр используется, если для параметра *Specify q-axis transient time constant* установлено значение *Open circuit*.
- **q-axis subtransient short-circuit, Tq”**: Сверхпереходная постоянная времени замкнутой цепи возбуждения по оси q. Этот параметр должен быть больше 0.
Этот параметр используется, если для параметра *Specify q-axis transient time constant* установлено значение *Short circuit*.

Параметры вкладки “Mechanical”:

- **Inertia constant, H**: Постоянная инерции;
- **Specify friction type**: Выбор типа учета трения:
Speed w - от скорости вращения;
Derivation from synchronous speed dw - от ускорения синхронной скорости вращения;
- **Damping coefficient**: Коэффициент демпфирования.

Параметры вкладки “Faulted Windings”:

- **Stator phase A**: Повреждение в фазе A статора:
None - нет;
One point - одна точка;
Two points - две точки.
- **1st point of fault from neutral (%)**: Удаленность первой точки от нейтрали статора **n** в %;
- **2st point of fault from neutral (%)**: Удаленность второй точки от нейтрали статора **n** в %;
- **Stator phase B**: Повреждение в фазе B статора:
None - нет;
One point - одна точка;
Two points - две точки.
- **1st point of fault from neutral (%)**: Удаленность первой точки от нейтрали статора **n** в %;
- **2st point of fault from neutral (%)**: Удаленность второй точки от нейтрали статора **n** в %;
- **Stator phase C**: Повреждение в фазе C статора:
None - нет;

One point - одна точка;
Two points - две точки.

- **1st point of fault from neutral (%)**: Удаленность первой точки от нейтрали статора **n** в %;
- **2st point of fault from neutral (%)**: Удаленность второй точки от нейтрали статора **n** в %;
- **Field winding**: Повреждение в обмотке возбуждения:
  None - нет;
  One point - одна точка;
  Two points - две точки.
- **1st point of fault from fd- (%)**: Удаленность первой точки от отрицательного вывода обмотки возбуждения **fd-** в %;
- **2st point of fault from fd- (%)**: Удаленность второй точки от отрицательного вывода обмотки возбуждения **fd-** в %.

Параметры вкладки “Saturation”:

- **Magnetic saturation representation**: Выбор типа задания кривой насыщения машины:
  None - нет;
  Points - по точкам;
  Factors - по смещению аналитической функции.
- **Magnetization current axes**: Выбор функции насыщения, как функции тока намагничивания:
  d-axes - по оси d;
  dq-axes - по осям d и q.
- **Per-unit field current saturation data, ifd**: Ток возбуждения, *ifd* в о.е., характеристики намагничивания. Этот параметр должен содержать вектор как минимум из пяти элементов;
- **Per-unit stator voltage saturation data, Vag**: Напряжение статора в режиме холостого хода, *Vag* в о.е., характеристики намагничивания. Этот параметр должен содержать вектор как минимум из пяти элементов;
- **Saturation factor at 1.0 pu open-circuit V**: Смещение вправо кривой насыщения вправо в точке напряжения статора в 1 о.е. в режиме холостого хода;
- **Saturation factor at 1.2 pu open-circuit V**: Смещение вправо кривой насыщения вправо в точке напряжения статора в 1.2 о.е. в режиме холостого хода.

Параметры вкладки “Initial Conditions”:

- **Terminal voltage magnitude**: Начальное значение напряжения генератора;
- **Terminal voltage angle**: Начальное значение угла напряжения генератора;
- **Active power generated**: Начальное значение генерируемой активной мощности;
- **Reactive power generated**: Начальное значение генерируемой реактивной мощности.

Список литературы

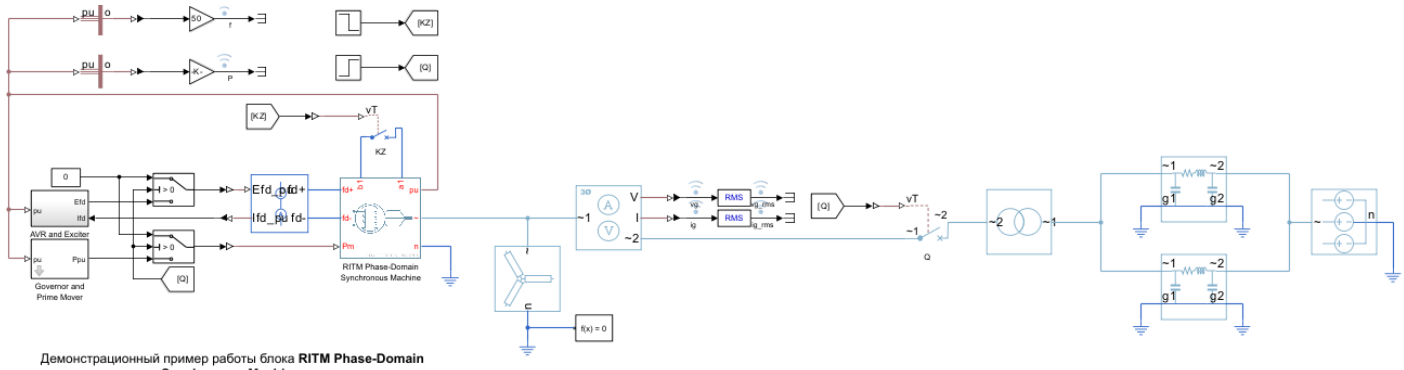
[1] A. B. Dehkordi, P. Neti, A.M Gole, and T.L. Maguire, ”Development and Validation of a ComprehensiveSynchronous Machine Model for a Real-Time Environment,” IEEE Trans. Energy Conversion, Vol. 25, No. 1, March 2010.

Примеры и демонстрации RITM Phase-Domain Synchronous Machine

Модель имитирует работу сети следующего типа: блок генератор-трансформатор, который работает на обобщенную энергосистему через две линии электропередачи. В примере для такой сети заложен следующий опыт:
1. На 10 секунде происходит внутреннее повреждение в обмотке статора синхронного генератора между фазами A и B на заданном удалении от нейтральной точки;
2. После повреждения через заданное время происходит имитация работы защит с отключением генераторного выключателя, гашением поля машины и сброса мощности.

Вы можете изменить вид повреждения и повторить опыт.
Обратите внимание, что пример уже настроен для запуска в реальном времени на КПМ “РИТМ”.

```
open_system('ritmdemo_block_synchronous_machine_pdsm.slx')
```



Демонстрационный пример работы блока **RITM Phase-Domain Synchronous Machine**:

Модель имитирует работу блока генератора - трансформатор на обобщенную энергосистему через две линии электропередачи. В пример заложен следующий опыт:
1. На 10 секунде работы модели происходит внутреннее повреждение в обмотке статора синхронного генератора между фазами A и B;
2. После повреждения через заданное время происходит имитация работы защит с отключением генераторного выключателя, гашением поля и сброса мощности.
Вы можете изменить вид повреждения и повторить опыт.

Обратите внимание, что пример уже настроен для запуска на КПМ РИТМ.

Вспомогательные функции

В данном разделе приводится информация о различных вспомогательных функциях.

- **ritm.energy.convertToComtrade**: преобразование данных в формат COMTRADE (создание файлов *.cfg* и *.dat*).

Вспомогательная функция ritm.energy.convertToComtrade

Данная функция используется для преобразования в формат COMTRADE (создания файлов *.cfg* и *.dat*) данных из Simulink - моделей, записанных в MAT-файл с помощью блока To File, или аналогичных по структуре данных. Преобразование происходит в соответствии с IEEE Std. C37.111-1999 или IEEE Std. C37.111-2013 в зависимости от входных параметров.
Примечание: Работа функции занимает некоторое время, после преобразования в командной строке будет выведено сообщение: *“Конвертация в формат Comtrade прошла успешно!”*.

Синтаксис

```
ritm.energy.convertToComtrade(matFileData, channelNumbers, comtradeFileName, option, value)
```

Входные параметры

- 'matFileData': массив данных для перевода в формат COMTRADE. Обратите внимание, что при записи MAT-файла в Simulink с помощью блока To File в виде массива, первая строка всегда равна времени модели. *Формат - числовой.*
- 'channelNumbers': вектор из двух чисел, где первое значение - это количество аналоговых сигналов, второе - дискретных. *Формат - числовой.*
- 'comtradeFileName': имя файлов *.cfg* и *.dat. Формат - строковый.*
- 'option': опциональный параметр для настройки.
- 'value': значение опционального параметра.

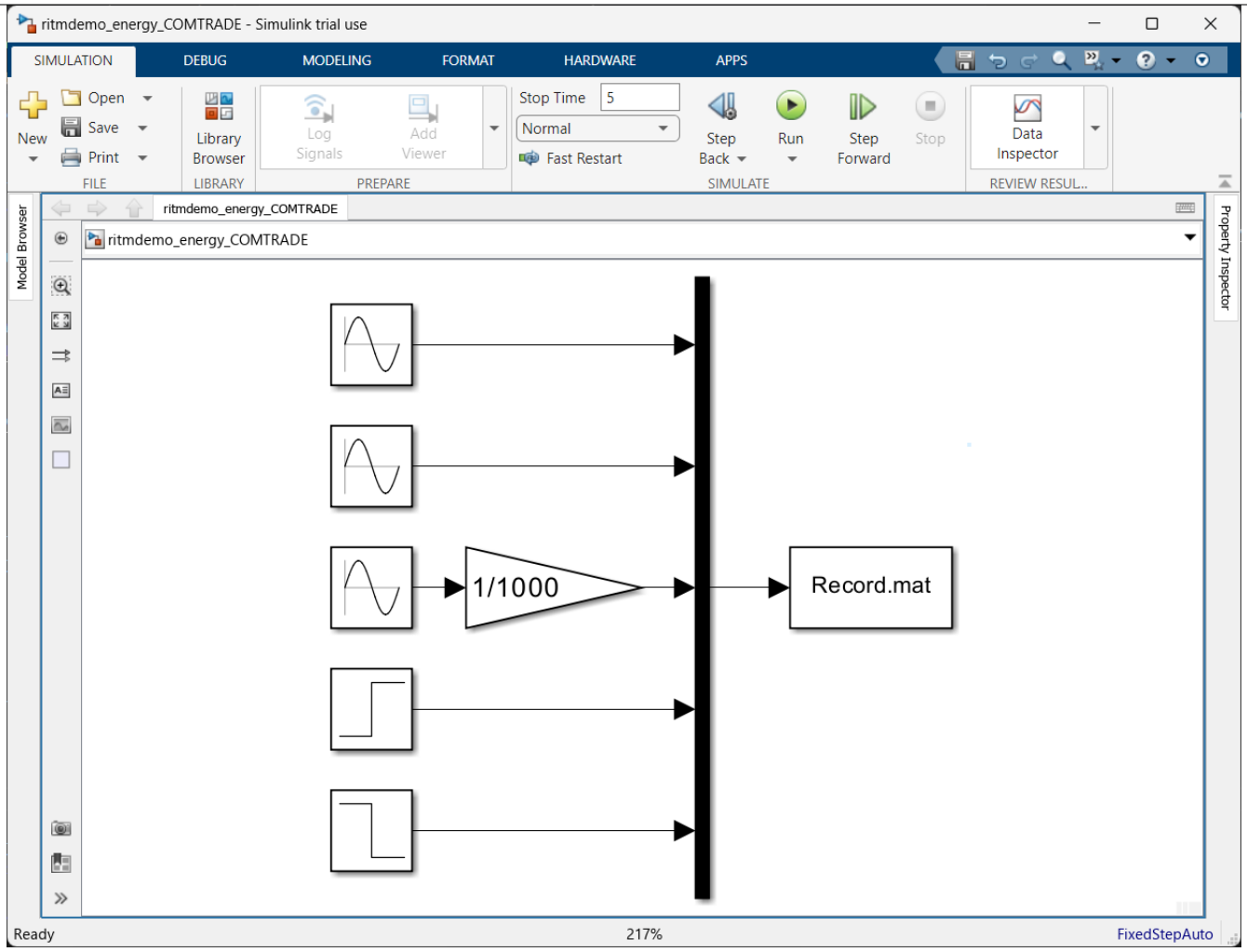
Допустимые параметры и их значения

Параметр	Значение	Описание
rev_year	‘1999’ или ‘2013’	Значение версии стандарта, по умолчанию используется ‘2013’
ft	Один из вариантов: ‘ASCII’, ‘BINARY’, ‘BINARY32’, ‘FLOAT32’	Тип кодирования данных в *.dat* файл, по умолчанию используется ‘BINARY’
ch_id	Вектор из двух ячеек, содержащие строковые векторы	Значения имен аналоговых и дискретных каналов, по умолчанию каналам без заданного имени присваивается следующее значение: для аналогового - “Analog signal” + его номер по порядку, для дискретного “Discrete signal” + его номер по порядку
uu	Строковый вектор	Значения единиц измерений аналоговых каналов, по умолчанию “NONE” для всех каналов
primary	Числовой вектор	Первичные номинальные значения аналоговых каналов, по умолчанию 1 для всех каналов
secondary	Числовой вектор	Вторичные номинальные значения аналоговых каналов, по умолчанию 1 для всех каналов
PS	Строковый вектор вектор из “P” или “S”	Идентификаторы масштабирования аналоговых каналов, в случае записи исходных данных в первичных величинах устанавливается значение “P”, во вторичных значение “S”

Пример использования
Предварительно сформируем массив данных для перевода в формат COMTRADE с помощью демо-модели:

```
open_system('ritmdemo_energy_COMTRADE.slx')
```

Данная модель записывает несколько синусоид и дискретных сигналов в MAT-файл с именем *Record.mat* в рабочую область с шагом 125 мкс.



Преобразование записанного массива в файлы *.cfg* и *.dat* в соответствии с IEEE Std. C37.111:

```
load('Record.mat');
ritm.energy.convertToComtrade(Record, [3 2], 'ExampleCOMTRADE');
```

Преобразование записанного массива в файлы *.cfg* и *.dat* в соответствии с IEEE Std. C37.111 - 1999 и учетом дополнительных параметров:
Примечание: Дополнительные параметры необязательны и могут передаваться на вход функции в произвольном порядке.

```
load('Record.mat');

% Опциональные параметры:
primary = 1000*ones(1, 3);
secondary = ones(1, 3);
PS = ["P" "P" "S"];
units = ["A" "A"];
analogChannelNames = {["I1" "I2"]};
discreteChannelNames = {["D1"]};
channelNames = [analogChannelNames discreteChannelNames];

ritm.energy.convertToComtrade(Record, [3 2], 'ExampleCOMTRADE', 'ft', 'ASCII', 'rev_year', '1999', 'primary', primary, 'secondary', secondary, ...
    'uu', units, 'PS', PS, 'ch_ID', channelNames);
```

Обратите внимание, что недостающие параметры имен и единиц измерения были заменены значениями по умолчанию.