Акционерное общество "СМС-Автоматизация"
(АО "СМС-Автоматизация")

УТВЕРЖДАЮ

Технический директор

АО "СМС-Автоматизация"

А.А. Сидоров

2016 г.

Библиотека CMC_S7_IEC101 для реализации протокола МЭК 870-5-101 в контроллерах SIMATIC S7

СЕРИЯ СМС-КОМКОН

Руководство по настройке и использованию

На 22 листах

2016

СОДЕРЖАНИЕ

1 ОБЩЕЕ ОПИСАНИЕ

1.1 Общая информация

Библиотека CMC_S7_IEC101 предоставляет контроллерам Siemens S7 300/400(H) возможность обмениваться данными с различными устройствами, используя протокол МЭК 870-5-101.

Передаваемая информация может иметь бинарный или числовой формат данных, опционально с меткой времени.

Работа драйвера возможна через:

- Переферийный интерфейс UART, модулей CP-341, CP340;
- Переферийный интерфейс Ethernet, модулей CP-343;
- интегрированный профинет интерфейс CPU 31xPN, 41xPN;
- профинет интерфейс ET200S IM151-8PN/DP CPU.

Библиотека CMC_S7_IEC101 разработана под проекты STEP 7 и PCS7

1.2 Описание составных частей

Библиотека представляет собой совокупность FC, FB, DB, UDT. Для корректной интеграции библиотеки в проект необходимо скопировать все необходимые блоки, кроме экземплярного блока данных, и в случае необходимости, изменить номер блока данных с параметрами. Экземплярный блок данных IEC101_DB необходимо создать вручную с любым доступным номером.

Таблица 1 – Набор блоков для реализации менеджера (клиента)

Символьное имя	Тип (номер)	Описание
Основные составляющие		
IEC101_FB	FB 251	Общий блок реализации менеджера
IEC101_DB	DB 251	Экземпляр **IEC101_FB**
IEC101_ IN	DB 252	Блок данных с принимаемыми параметрами
IEC101_ OUT	DB 253	Блок данных с отправляемыми параметрами
IEC101_Buffer	DB 254	Блок данных для временного хранения отправляемых параметров
IEC101_DT_Converter	FC 251	Вспомогательная функция для преобразования метки времени
Вспомогательные общие стандартные блоки		
AD_DT_TM	FC 1*	См. Справку
GT_DT	FC 14*	См. Справку
SB_DT_DT	FC 34*	См. Справку

Символьное имя	Тип (номер)	Описание
SB_DT_TM	FC 35*	См. Справку
Транспортные блоки для соединений через CP-340		
P_RCV	FB 2*	Системная функция для отправки в сокет для внутреннего порта CPU
P_SEND	FB 3*	Системная функция для приема из сокета для внутреннего порта CPU
UART_CP340	FB 297	Транспортный уровень обмена данными через модули CP-340
UART_CP340_DB	DB 297	Экземплярный блок данных FB299
Транспортные блоки для соединений через CP-341		
P_RCV_RK	FB 7*	Системная функция для отправки в сокет для внутреннего порта CPU
PSEND_RK	FB 8*	Системная функция для приема из сокета для внутреннего порта CPU
UART_CP341	FB 299	Транспортный уровень обмена данными через модули CP-341
UART_CP341_DB	DB 299	Экземплярный блок данных FB299
Транспортные блоки для соединений через встроенные в CPU порты Ethernet		
TSEND	FB 63*	Системная функция для отправки в сокет для внутреннего порта CPU
TRCV	FB 64*	Системная функция для приема из сокета для внутреннего порта CPU
TCON	FB 65*	Системная функция для установки соединения для внутреннего порта CPU
TDISCON	FB 66*	Системная функция для терминации соединения для внутреннего порта CPU
TCON_PAR	UDT 65*	Структура с параметрами соединения
TCP_T_Block	FB 298*	Транспортный уровень обмена данными через встроенные Ethernet порты CPU
TCP_T_Block_DB	DB 298*	Экземплярный блок данных FB298

*В случае наличия блоков под данными номерами с иным назначением, необходимо либо номера существующих блоков и скопировать блоки для библиотеки, либо сделать Rewire внутри библиотеки и скопировать блоки в проект включая IEC101_FB.

1.3 Структурная схема драйвера протокола

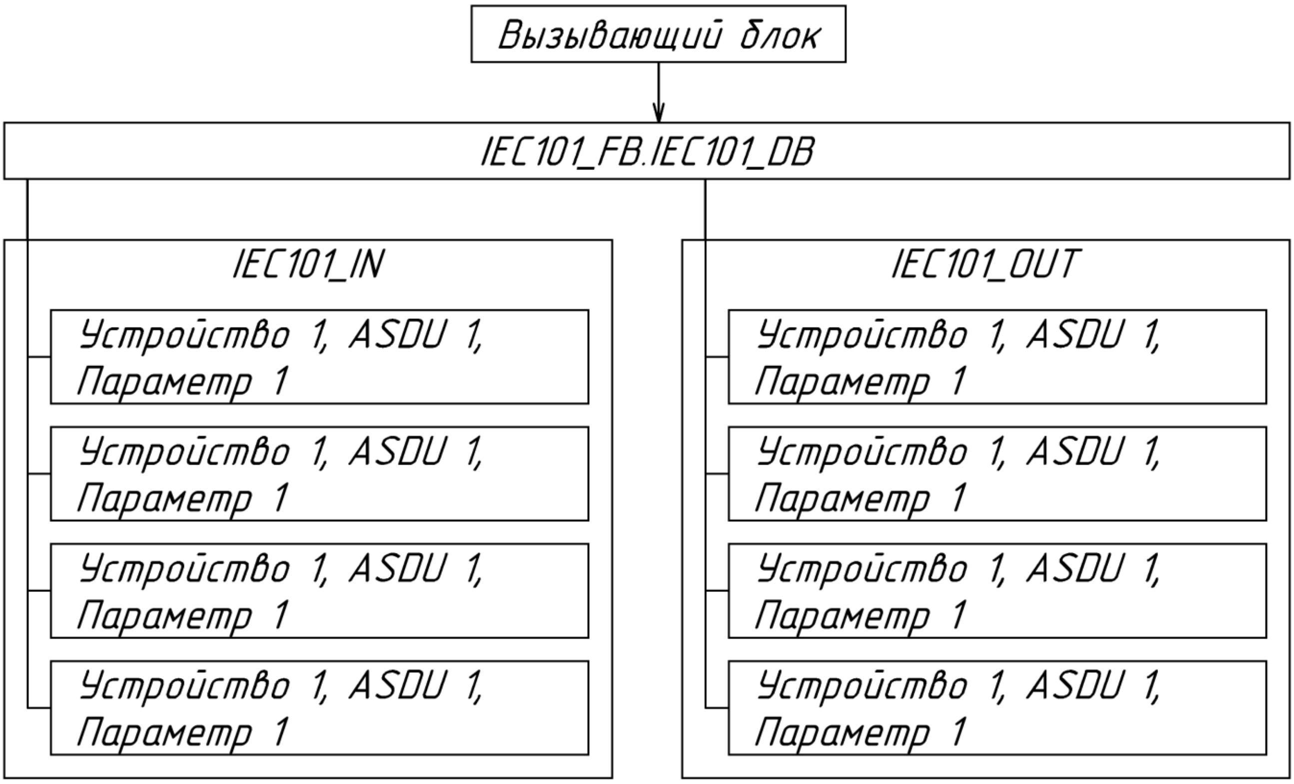

Рисунок 1 - Структурная схема

1.4Интеграция в проект

Последовательность действий для начала работы

Последовательность действий для начала работы:

1 Скопировать все необходимые блоки, перечисленные в таблице 1 (все блоки из нужного примера в проекте).

2 Если блок удобно вызвать как функцию с входами и выходами, то рекомендуется добавить в проект исходный файл IEC 101 *** IO, и указать все необходимые входа и выхода. Либо использовать второй вариант и настроить все параметры в области констант исходного файла и скомпилировать FC250.

3 В программе вызвать функцию FC250 с необходимыми параметрами (или без них, в зависимости от выбранной в предыдущем пункте). (Внимание! Скорость работы библиотеки будет зависеть от частоты вызова, на примем или отдачу одной команды требуется до 4 вызовов)

4 Заполнить IEC101_IN и IEC101_OUT соответственно.

5 Загрузить все необходимые блоки в контроллер.

Примеры использования библиотеки

Пример вызова FC250.

```
FC250();
Или
FC250(
        ,inLaddr :=  1          // Номер первого байта Input области, занимаемой CP, из hardware
        ,inLinkAddress := 1                 // Размер в байтах, занимаемый под сетевой адрес
        ,inLinkAddrSize := 2              // Размер в байтах, занимаемый под сетевой адрес
        ,inCASDUSize := 1              // Размер в байтах, занимаемый под ASDU
        ,inIOASize := 1                   // Размер в байтах, занимаемый под IOA
        ,inBalancedMode := false  // Работать в балансном режиме
        ,inItIsMain := true    // В балансном режиме – станция А; в небалансном – мастер.
        ,inReqAck := true // запрашивать подтверждение каждой посылки
        ,inUseCause10 := true // Отправлять кадр завершения активации команд
        ,inConTimeout := 10.0          // Таймаут обрыва связи
        ,inAckTimeout := 1.0             // Время ожидания подтверждения
        ,inAskStatusPeriod := 5.0; // Период опроса статуса соединения
        ,inAskClass2Period := 5.0; // Период запроса данных класса 2
        );
```

Пример заполнения IEC101_IN.

```
DATA_BLOCK "IEC101_IN"
        STRUCT
          Object1 : STRUCT
             Link_Addr: INT := 1;  // Сетевой адрес устройства
             ASDU : INT  := 1;   // Адрес блока данных прикладного уровня
             IOA  : DINT := L#128;  // Идентификационный адрес объекта
             Type_obj : INT := 1; // Тип объекта
             Value    : DWORD;      // Значение
             Bools    : BYTE; // Булевые параметры принятого сигнала
             Params   : BYTE; // Настройки данного параметра
             Time_rec : DATE_AND_TIME; // Время из метки
             Time_confirm : DATE_AND_TIME;
          END_STRUCT;
          Object2 : STRUCT
             Link_Addr: INT := 1;  // Сетевой адрес устройства
             ASDU : INT  := 1;   // Адрес блока данных прикладного уровня
             IOA  : DINT := L#130;   // Идентификационный адрес объекта
             Type_obj : INT := 13;  // Тип объекта
             Value    : Real := 123.45; // Начальное значение
             Bools    : BYTE:= 16#00; // Булевые параметры принятого сигнала
             Params   : BYTE; // Настройки данного параметра
             Time_rec : DATE_AND_TIME; // Время из метки
             Time_confirm : DATE_AND_TIME;
          END_STRUCT;
        END_STRUCT
        BEGIN
END_DATA_BLOCK
```

Пример заполнения IEC101_OUT.

```
DATA_BLOCK "IEC101_OUT"
        STRUCT
            Group : STRUCT     // Групповой запрос
               Link_addr: INT := 1;    // Сетевой адрес устройства (Обязателен для небалансного режима)
               ASDU     : INT := 1;    // Адрес блока данных прикладного уровня    (Обязательно)
               IOA      : DINT:= L#0;  // Идентификационный адрес объекта     (Обязательно)
               Type_obj : INT := 100;    // Тип объекта (Обязательно)
               Value    : DWORD;         // Передаваемое значение (не используется)
               Bools    : BYTE;          // Как правило байт качества
               Params   : BYTE;          // Параметры
               Time_snd : DATE_AND_TIME; // Время последнего изменения (для метки времени)
               Time_left: REAL;    // Время до обновления
               Period   : REAL;    // Период обновления
            END_STRUCT;
            object1 : STRUCT
```

```
          Link_addr: INT := 1;    // Сетевой адрес устройства (Обязателен для небалансного режима)
          ASDU  : INT := 1;    // Адрес блока данных прикладного уровня  (Обязательно)
          IOA : DINT:= L#1;  // Идентификационный адрес объекта    (Обязательно)
          Type_obj : INT := 1;    // Тип объекта (Обязательно)
          Value    : DWORD;      // Передаваемое значение
          Bools    : BYTE;       // Как правило байт качества
          Params   : BYTE; // Параметры
          Time_snd : DATE_AND_TIME; // Время последнего изменения, для метки времени
          Time_left: REAL;    // Время до обновления
          Period   : REAL;   // Период обновления
        END_STRUCT;
        object2 : STRUCT
          Link_addr: INT := 1;    // Сетевой адрес устройства ( Обязателен для небалансного режима)
          ASDU     : INT := 1;    // Адрес блока данных прикладного уровня    (Обязательно)
          IOA      : DINT:= L#2;  // Идентификационный адрес объекта      (Обязательно)
          Type_obj : INT := 1;    // Тип объекта (Обязательно)
          Value    : DWORD;      // Передаваемое значение
          Bools    : BYTE;           // Как правило байт качества
          Params   : BYTE;          // Параметры
          Time_snd : DATE_AND_TIME; // Время последнего изменения, для метки времени
          Time_left: REAL;          // Время до обновления
          Period   : REAL;          // Период обновления
        END_STRUCT;
      END_STRUCT
      BEGIN
END_DATA_BLOCK
```

Описание входов вызываемого блока IEC101_FB

IEC101_FB.

inCon_laddr - номер первого байта Input области, занимаемой CP, из hardware

inDB_in - блок IEC101_IN

inDB_out - блок IEC101_OUT

inLink_Addr - размер в байтах, занимаемый под сетевой адрес

inLink_Addr_sz - размер в байтах, занимаемый под сетевой адрес

inCASDU_sz - размер в байтах, занимаемый под ASDU

inIOA_sz - размер в байтах, занимаемый под IOA

inBalanced_mode - работать в балансном режиме

inIt_is_main - DIR=1 или признак мастера

inReqAck - запрашивать подтверждение каждой посылки

inUseCause10 - отправлять кадр завершения активации команд

inAll_reset - перезапуск всех соединений

inCon_timeout - таймаут обрыва связи

inAck_timeout - время ожидания подтверждения

inAsk_status_period - период опроса статуса соединения

inAsk_class2_period - период запроса данных класса 2

inBuffer - указатель на буфер для хранения меток времени спонтанных данных

inDriverDB – экземплярный блок транспорта

Настройка списка параметров

В блоках IEC101_IN и IEC101_OUT перечисляются объекты. Каждый параметр должен быть структурой. Имя структуры может быть любым, по усмотрению пользователя.

Таблица 2 – Структура принимаемого объекта в IEC101_IN

Данные о параметре	Тип	Описание
Link_Addr	INT	Сетевой адрес устройства
ASDU	INT	Адрес блока данных прикладного уровня
IOA	DINT	Идентификационный адрес объекта
Type_obj	INT	Тип объекта
Value	DWORD REAL DINT	Значение. Используется только в объектах, размер данных у которых превышает 1 байт или не содержит байт качества
Bools	BYTE	Как правило байт качества
Params	BYTE	Настройки данного параметра: 0..2й байты – группа (1-8) 3й байт используется для фонового сканирования 4й байт выставляется во время группового опроса 5й байт выставляется во время общего опроса 6й байт выставляется для спонтанной передачи 7й байт – разрешение на периодическую отправку объекта
Time_rec	DATE_AND_TIME	Время получения объекта
Time_confirm	DATE_AND_TIME	Время подтверждения команды

Таблица 3 – Структура принимаемого объекта в IEC101_OUT

Данные о параметре	Тип	Описание
Link_Addr	INT	Сетевой адрес устройства
ASDU	INT	Адрес блока данных прикладного уровня
IOA	DINT	Идентификационный адрес объекта
Type_obj	INT	Тип объекта
Value	DWORD REAL DINT	Значение. Используется только в объектах, размер данных у которых превышает 1 байт или не содержит байт качества
Bools	BYTE	Как правило байт качества
Params	BYTE	Настройки данного параметра: 0..2й байты – группа (1-8) 3й байт используется для фонового сканирования 4й байт выставляется во время группового опроса 5й байт выставляется во время общего опроса 6й байт выставляется для спонтанной передачи 7й байт – разрешение на периодическую отправку объекта
Time_snd	DATE_AND_TIME	Время изменения объекта
Time_left	REAL	Оставшееся время до обновления
Period	REAL	Период обновления

1.5 Диагностика библиотеки

Таблица 4 – Диагностические переменные блока IEC101_DB

Имя переменной в IEC101_DB	**Тип**	**Описание**
outStatus	DWORD	Статус работы или код ошибки
outError	BOOL	Бит индикации ошибки

Наиболее распространенные состояния менеджера перечислены в таблице 5.

Таблица 5 – Расшифровка статуса переменной STATUS

Статус	**Описание**
0000 xxxx	Внутренняя неисправность. Обнаружена ошибка входных параметров.
0000 0000	Блок не вызывается
0000 0001	Указанный блок данных принимаемых параметров не существует
0000 0002	Указанный блок данных отправляемых параметров не существует
0000 0003	Неверно указан параметр inLink_Addr_sz
0000 0004	Неверно указан параметр inCASDU_sz
0000 0005	Неверно указан параметр inIOA_sz
0000 0006	Ошибка в параметрах

Статус	Описание
0000 000A	Принят параметр, отсутствующий в IEC101_IN
0000 000B	Принята незавершенная посылка
0000 000C	Пакет содержит нулевое количество объектов
0000 000D	Соединение 101 отсутствует
0000 000E	Ошибка подсчета CRC принятого пакета
0000 000F	Принят мусор
0001 xxxx	Нормальная работа
0002 xxxx	Ошибка от P_RCV_RK
0003 xxxx	Ошибка от P_SND_RK

Настройка транспорта для соединений через CP-340

Для корректной работы данного вида транспорта необходимо выполнить следующие условия.

В проекте должны присутствовать следующие блоки:

P_RCV	FB 2*	Системная функция для отправки в сокет для внутреннего порта CPU
P_SEND	FB 3*	Системная функция для приема из сокета для внутреннего порта CPU
UART_CP340	FB 297	Транспортный уровень обмена данными через модули CP-340
UART_CP340_DB	DB 297	Экземплярный блок данных FB299

В Hardware должен присутствовать CP340, возможно несколько (Рисунок 2)

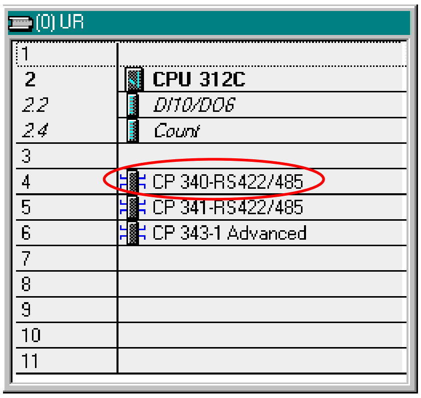

Рисунок 2

Далее необходимо настроить добавленный коммуникационный процессор в соответствии с Рисунок 3, Рисунок 4, Рисунок 5.

В настройках коммуникационного процессора заходим в параметры. Для того, чтобы кнопка была активна, необходимо установить модуль «*CP PtP* Param».

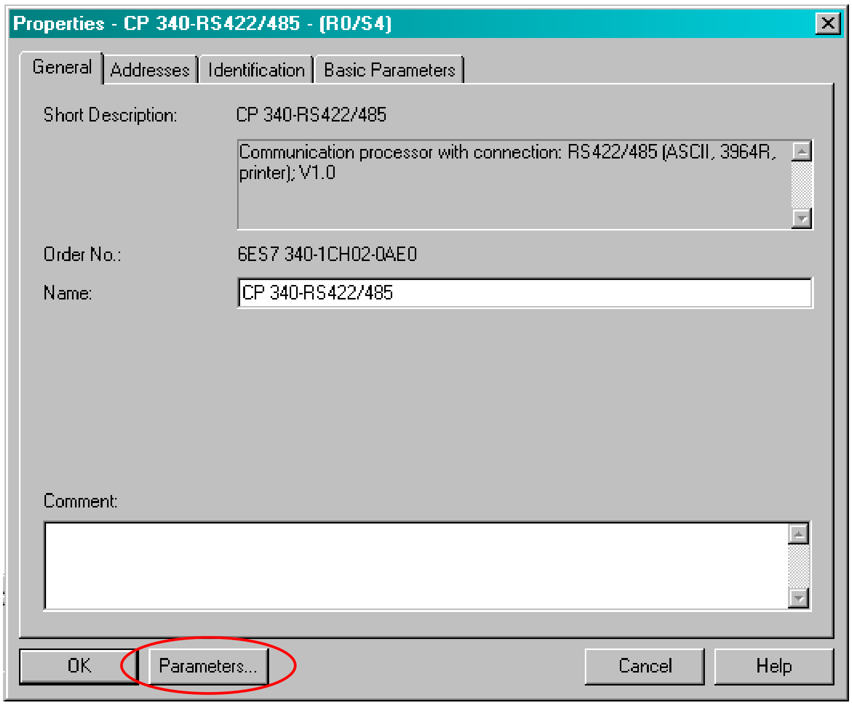

Рисунок 3

В качестве типа связи выбрать ASCII (Рисунок 4)

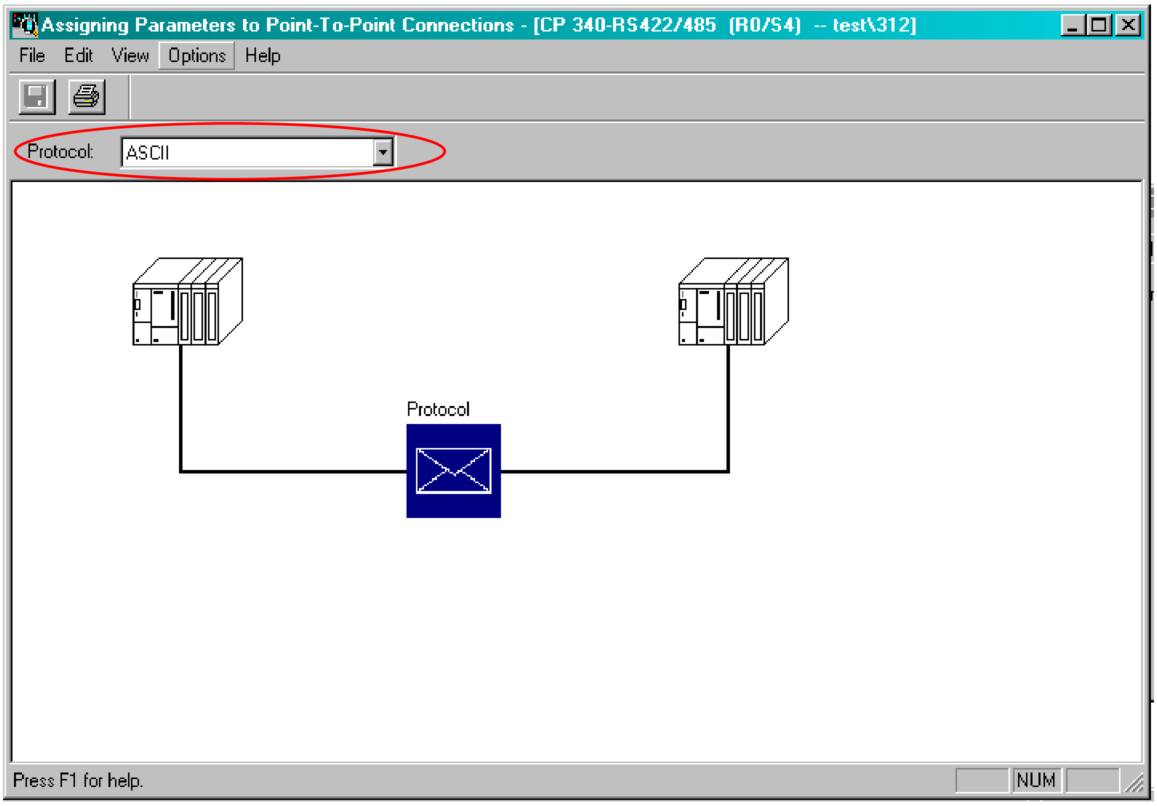

Рисунок 4

Открыть настройки протокола (двойным нажатием на синий конверт из «Рисунок 4»). Произвести настройки канала в соответствии с требуемыми, остальные параметры оставить как на рисунке 6.

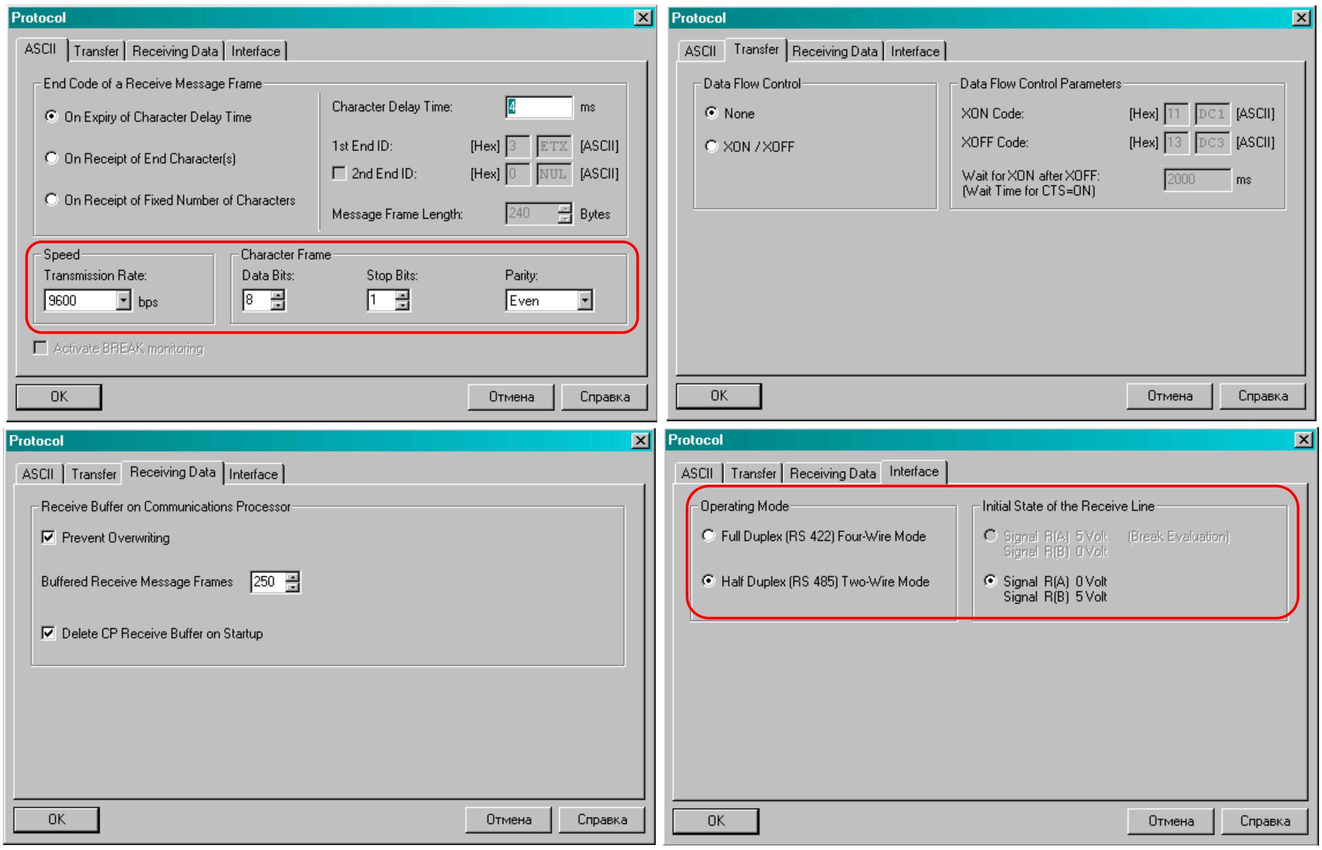

Рисунок 5

При настройке вызова библиотеки, нужно будет указать LADDR. Здесь требуется указать адрес модуля. Данный адрес определяется из Hardware (Рисунок 6)

Slot	Module	Order number	Firmware	MPI address	I address	Q address	Comment
1							
2	CPU 312C	6ES7 312-5BF04-0AB0	V3.3	2			
2.2	DI10/DO6				124...125	124	
2.4	Count				768...783	768...783	
3							
4	CP 340-RS422/485	6ES7 340-1CH02-0AE0			256...271	256...271	
5							

Рисунок 6

Также необходимо убедиться в корректном вызове блока UART_CP340.

```
// Пример вызова
UART_CP340.UART_CP340_DB(inLADDR := 256);
```

В поле обработчика должен быть указан экземплярный блок UART_CP340_DB.

```
// Пример указания экземплярного блока
,inDriverDB := UART_CP340_DB
```

Настройка транспорта для соединений через СР-341

Для корректной работы данного вида транспорта необходимо выполнить следующие условия.

В проекте должны присутствовать следующие блоки:

P_RCV_RK	FB 7*	Системная функция для отправки в сокет для внутреннего порта CPU
PSEND_RK	FB 8*	Системная функция для приема из сокета для внутреннего порта CPU
UART_CP300	FB 299	Транспортный уровень обмена данными через модули СР-341
UART_CP300_DB	DB 299	Экземплярный блок данных FB299

В Hardware должен присутствовать CP341, возможно несколько (Рисунок 7)

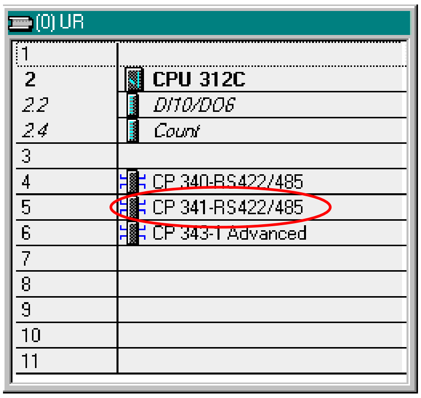

Рисунок 7

Далее необходимо настроить добавленный коммуникационный процессор в соответствии с «Рисунок 8», «Рисунок 9», «Рисунок 10».

В настройках коммуникационного процессора заходим в параметры. Для того, чтобы кнопка была активна, необходимо установить модуль «*CP PtP* Param».

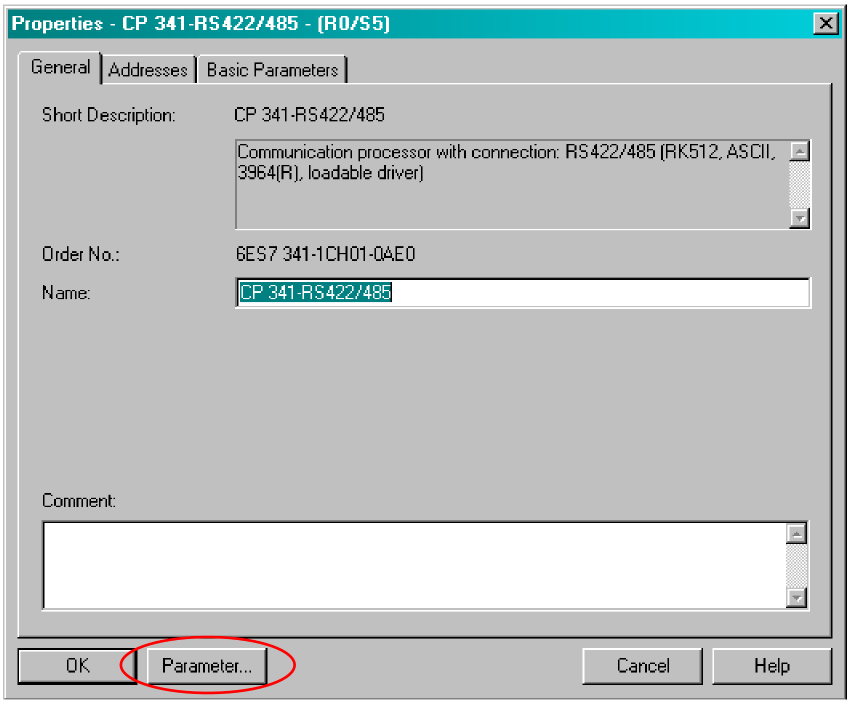

Рисунок 8

В качестве типа связи выбрать ASCII (Рисунок 9)

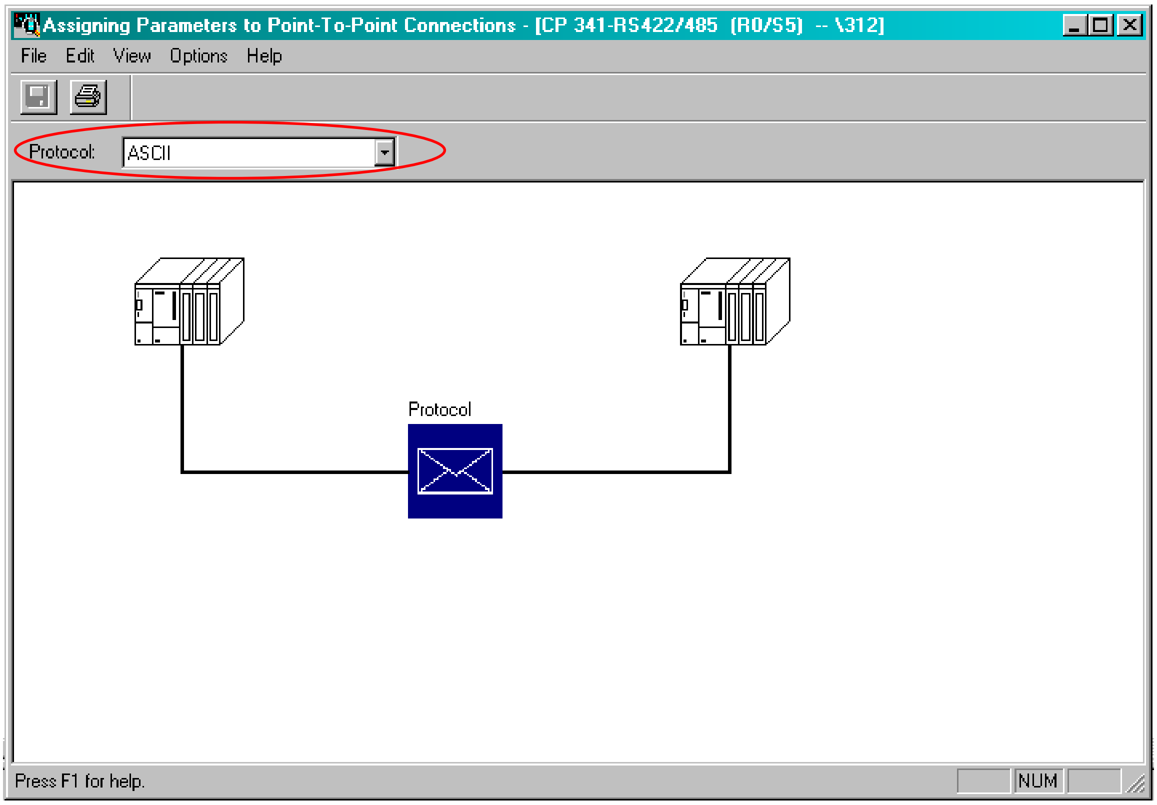

Рисунок 9

Открыть настройки протокола (двойным нажатием на синий конверт из «Рисунок 9»). Произвести настройки канала в соответствии с требуемыми, остальные параметры оставить как на «Рисунок 10».

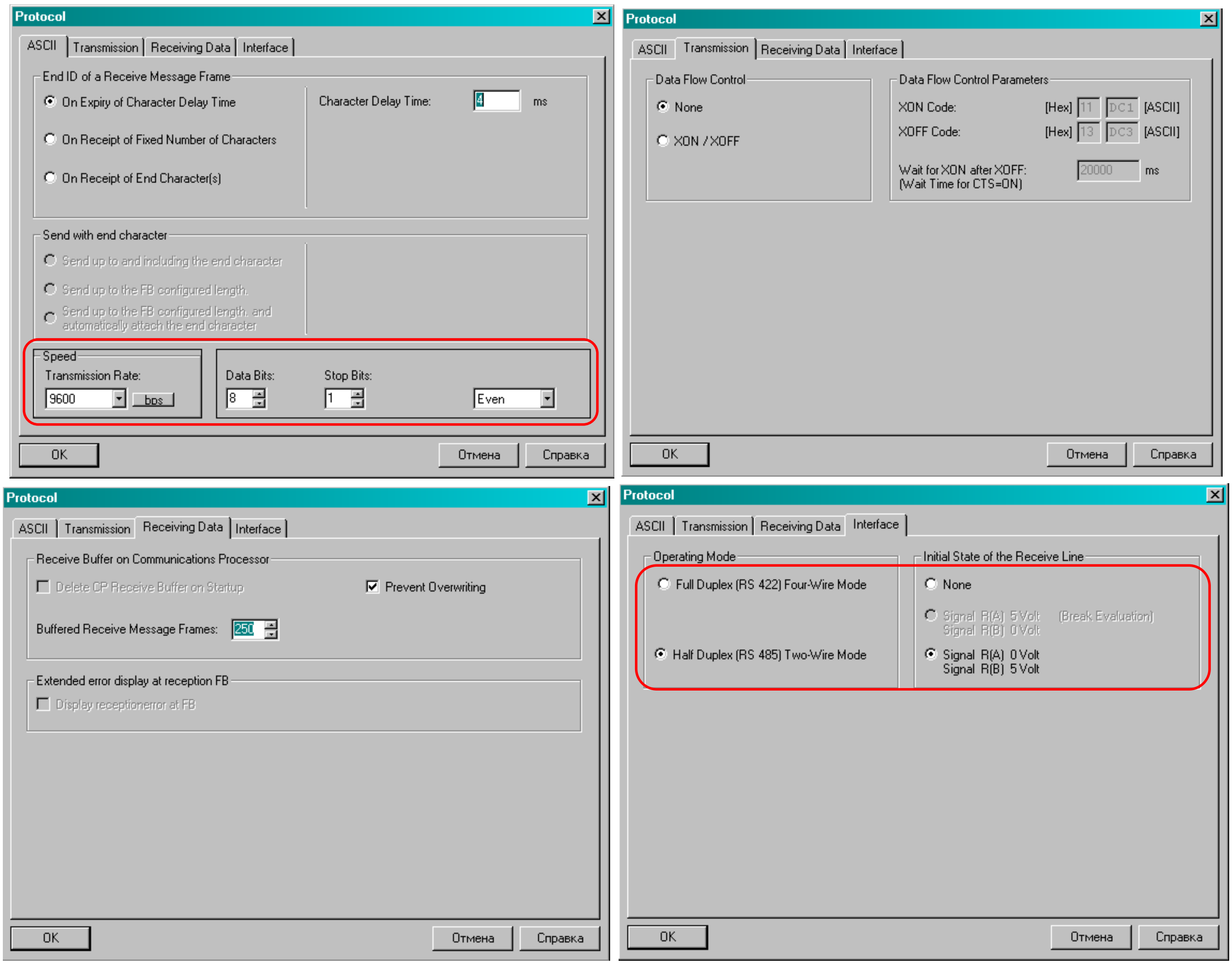

Рисунок 10

При настройке вызова библиотеки, нужно будет указать LADDR(адрес модуля). Данный адрес определяется из Hardware (Рисунок 11)

Slot	Module	Order number	Firmware	MPI address	I address	Q address	Comment
1							
2	CPU 312C	6ES7 312-5BF04-0AB0	V3.3	2			
2.2	DI10/DO6				124...125	124	
2.4	Count				768...783	768...783	
3							
4	CP 340-RS422/485	6ES7 340-1CH02-0AE0			256...271	256...271	
5	CP 341-RS422/485	6ES7 341-1CH01-0AE0			272...287	272...287	
6	CP 343-1 Lean	6GK7 343-1CX10-0XE0	V2.0	3	288...303	288...303	
7							

Рисунок 11

Также необходимо убедиться в корректном вызове блока UART_CP341.

```
// Пример вызова
UART_CP341.UART_CP341_DB(inLADDR := 256);
```

В поле обработчика должен быть указан экземплярный блок UART_CP341_DB.

```
// Пример указания экземплярного блока
,inDriverDB := UART_CP341_DB
```

Настройка транспорта для соединений через встроенные в CPU порты Ethernet

Для корректной работы данного вида транспорта необходимо выполнить следующие условия.

В проекте должны присутствовать следующие блоки:

TSEND	FB 63*	Системная функция для отправки в сокет для внутреннего порта CPU
TRCV	FB 64*	Системная функция для приема из сокета для внутреннего порта CPU
TCON	FB 65*	Системная функция для установки соединения для внутреннего порта CPU
TDISCON	FB 66*	Системная функция для терминации соединения для внутреннего порта CPU
TCON_PAR	UDT 65*	Структура с параметрами соединения
TCP_T_Block	FB 298*	Транспортный уровень обмена данными через встроенные Ethernet порты CPU
TCP_T_Block_DB	DB 298*	Экземплярный блок данных FB298

Необходимо убедится в правильном указании параметра Local_device_id из Hardware, номер напротив Ethernet периферии в CPU (по умолчанию PN-IO). (Рисунок 12).

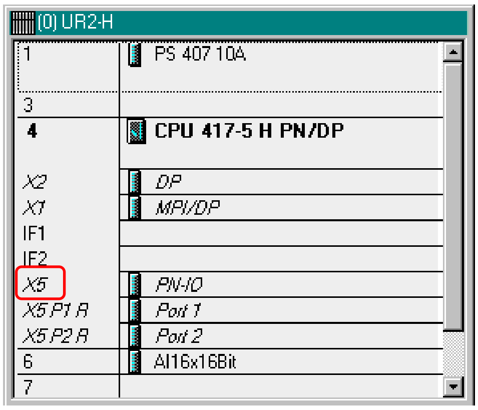

Рисунок 12

При наличии нескольких TCP соединений в контроллере необходимо указать свободный ID соединения (параметр inConnect_ID).

Параметр inLocal_port не указывается, так как пассивное соединение еще не реализовано.

Также необходимо убедиться в корректном вызове блока TCP_T_block.

```
// Пример вызова
TCP_T_block.TCP_T_block_DB(inIPAddress_1 := 192
                        ,inIPAddress_2 := 168
                        ,inIPAddress_3 := 1
                        ,inIPAddress_4 := 13
                        ,inPort       := 508
                        ,inLocal_device_id := 1
                        //,inConnect_ID := W#16#0063    // Connection ID (Оставляем по умолчанию)
                        //,inLocal_port := 2000;     // Local Port (2000) (При активном соединении не используется)
                      );
```

В поле обработчика должен быть указан экземплярный блок TCP_T_block_DB.

```
// Пример указания экземплярного блока
,inDriverDB := TCP_T_block_DB
```

2 ЗАГРУЗКА ПРОЕКТА В КОНТРОЛЛЕР

После завершения конфигурирования всех блоков, связанных с IEC101, необходимо скомпилировать проект и прогрузить все необходимые блоки.

3 ЛИЦЕНЗИРОВАНИЕ

Функция лицензирования на даный момент не предусмотерена

Библиотеки IEC104 защищены от несанкционированного использования с привязкой к карте памяти. Ядро логики FB104 "IEC104" скрыто под флагом Know_How_Protect. Для использования библиотеки необходимо провести активацию:

1 В название проекта ввести в скобках девятизначный код:

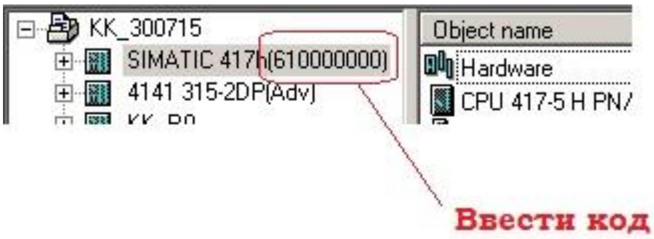

Рисунок 1 – Ввод девятизначного ключа в название проекта

2 В исходном SCL IEC104_COM ввести десятизначные ключи активации для каждого соединения. Для одного соединения вводится 3 ключа под 3 карты памяти (Листинг 1).

Листинг 1:

```
DATA_BLOCK "IEC104_DB1" IEC104
BEGIN
    Password[0] := b#16#FC; Password_Red[0] := b#16#FC; Password_Res[0] := b#16#FC;
    Password[1] := b#16#AC; Password_Red[1] := b#16#AC; Password_Res[1] := b#16#AC;
    Password[2] := b#16#E8; Password_Red[2] := b#16#E8; Password_Res[2] := b#16#E8;
    Password[3] := b#16#DD; Password_Red[3] := b#16#DD; Password_Res[3] := b#16#DD;
    Password[4] := b#16#ED; Password_Red[4] := b#16#ED; Password_Res[4] := b#16#ED;
    Password[5] := b#16#DD; Password_Red[5] := b#16#DD; Password_Res[5] := b#16#DD;
    Password[6] := b#16#EB; Password_Red[6] := b#16#EB; Password_Res[6] := b#16#EB;
    Password[7] := b#16#DE; Password_Red[7] := b#16#DE; Password_Res[7] := b#16#DE;
    Password[8] := b#16#E8; Password_Red[8] := b#16#EB; Password_Res[8] := b#16#E8;
    Password[9] := b#16#D0; Password_Red[9] := b#16#D2; Password_Res[9] := b#16#D0;
END_DATA_BLOCK
```

При неправильном вводе ключа более 4 раз происходит блокировка. Для разблокировки нужно прогрузить экземплярный блок данных, например, IEC104_DB1, и снова ввести ключ.

Если все ключи введены корректно, то библиотека работает в штатном режиме. При некорректном вводе ключей работа невозможна.

Для корректной работы функции лицензирования в проекте должны быть обязательно FC1 – LICENSE_CHECK, DB9 – LICENSE_DB. При использовании актуальных FC, FB из библиотеки, в DB9.DBX0.0 взводится бит лицензирования. Этот признак никак не связан ни с активацией библиотеки по ключам в названии проекта, ни с активацией по ключу каждого ядра IEC104.

4 КОНТАКТЫ

Если у Вас есть вопросы по поводу работы библиотек CMC_S7_IEC101_UART или Вы хотите получить совет по телефону, мы Вам поможем.

По вопросам приобретения библиотек и техническим вопросам обращайтесь по контактам:

Тел./факс: (846) 993-83-83

E-mail: license.plc@sms-a.ru

Web: www.sms-automation.ru

ПРИЛОЖЕНИЕ А

Перечень терминов, сокращений и нормативной документации

Перечень терминов и сокращений

– CP – коммуникационный процессор;

– CPU – центральный процессор;

– ET200S – станции распределенного ввода-вывода;

– FC, FB, DB, UDT – логические блоки/блоки данных;

– S7-300/ S7-400 – программируемые логические контроллеры Siemens Simatic;

– SCL – текстовый язык программирования основанный на стандарте IEC 1131-3;

– Simatic STEP 7 – базовый пакет программ, включающий в свой состав весь спектр инструментальных средств, необходимых для программирования и эксплуатации систем автоматизации, построенных на основе программируемых контроллеров Simatic S7/C7, а также систем компьютерного управления Simatic WinAC;

Лист регистрации изменений									
Изм.	**Номера листов (страниц)**				**Всего листов (страниц) в документе**	**Номер документа**	**Входящий номер сопроводительного документа и дата**	**Подпись**	**Дата**
	измененных	**замененных**	**новых**	**аннулированных**					