Данный проект демонстрирует пример использования библиотеки CAA SerialCOM для реализации обмена данными с внешним устройством через последовательный порт ПЛК NLScon-RSB. В примере рассматривается возможность отправки запроса и приема ответа от устройства. В качестве подключенного к ПЛК устройства рассматривался модуль RealLab! NLS-16DI, настроенный для работы по протоколу DCON (9600, 8N1, адрес 09hex).

Для разработки проекта использована среда Codesys 3.5.16 patch 4 и установочные пакеты компонентов для модулей RealLab! CoDeSys Linux package.

Для работы в проект Codesys понадобится добавить библиотеку CAA SerialCOM. Эта библиотека реализует 4 функциональных блока, выполняющих основные функции с COM портом: открытие, запись, чтение и закрытие (см. рис. 1).

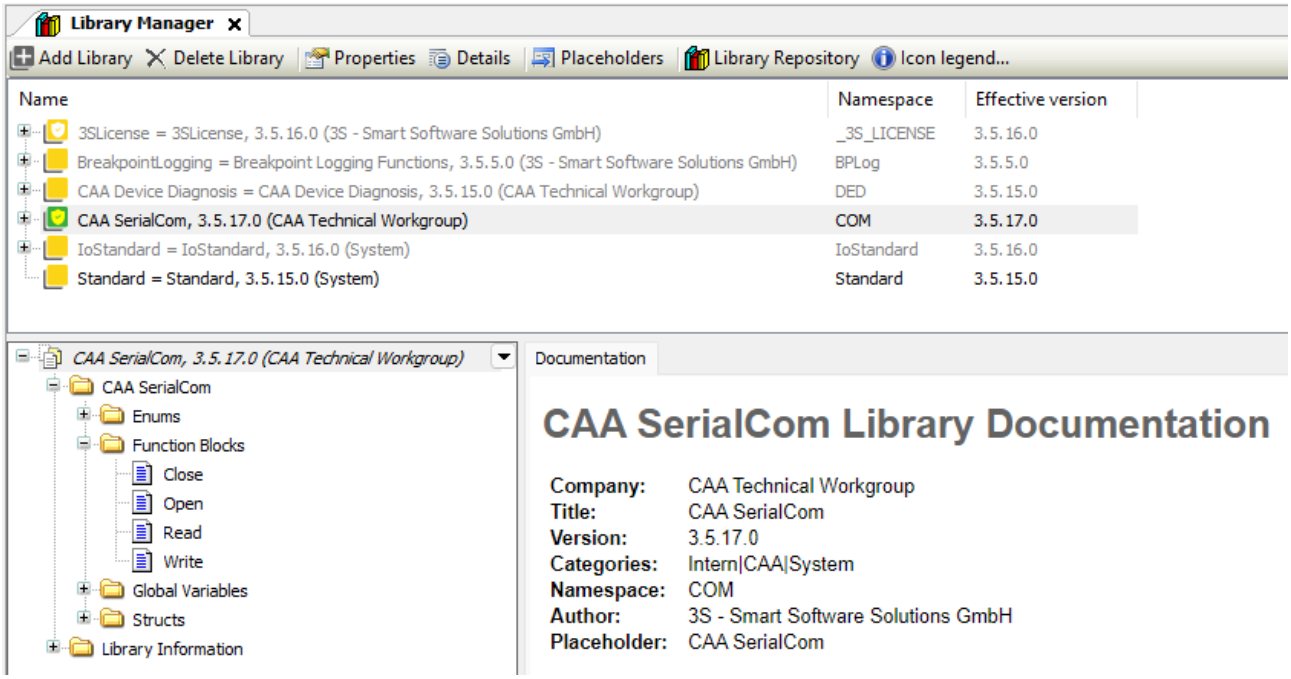

Рис.1 Библиотека CAA SerialCOM

ФБ COM.Open

Scope	Name	Type	Comment
Input	xExecute	BOOL	Нарастающий фронт (перевод из FALSE в TRUE): ФБ начинает выполняться Падающий фронт (из TRUE в FALSE): сброс выходов в 0.
Output	xDone	BOOL	TRUE: ФБ успешно завершил работу
	xBusy	BOOL	TRUE: ФБ в состоянии выполнения
	xError	BOOL	TRUE: Произошла ошибка, ФБ прекращает выполняться FALSE: Ошибок не произошло
Input	usiListLength	USINT	Количество входных параметров в массиве pParameterList
	pParameterList	CAA.PVOID	Указатель на массив с параметрами COM порта
Output	eError	ERROR	Ошибка выполнения ФБ
	hCom	CAA.HANDLE	Возвращаемый ФБ обработчик открытого COM порта

ФБ COM.Write

Scope	Name	Type	Comment
Input	xExecute	BOOL	Нарастающий фронт (перевод из FALSE в TRUE): ФБ начинает выполняться Падающий фронт (из TRUE в FALSE): сброс выходов в 0.
	xAbort	BOOL	TRUE: ФБ сразу прерывает работу, все его выходы устанавливаются в значения по умолчанию.
	udiTimeOut	UDINT	Определяет время (µs), по истечении которого ФБ прерывает работу из-за таймаута с сообщением об ошибке
Output	xDone	BOOL	TRUE: ФБ успешно завершил работу
	xBusy	BOOL	TRUE: ФБ в состоянии выполнения
	xError	BOOL	TRUE: Произошла ошибка, ФБ прекращает выполняться FALSE: Ошибок не произошло
	xAborted	BOOL	TRUE : Работа ФБ прервана пользователем
Input	hCom	CAA.HANDLE	Обработчик открытого COM порта (предварительно получен как результат выполнения COM.Open)
	pBuffer	CAA.PVOID	Указатель на буфер данных для отправки в COM порт
	szSize	CAA.SIZE	Количество байт данных, подготовленных для отправки в COM порт
Output	eError	ERROR	Ошибка выполнения ФБ

ФБ COM.Read

Scope	Name	Type	Comment
Input	xExecute	BOOL	Нарастающий фронт (перевод из FALSE в TRUE): ФБ начинает выполняться Падающий фронт (из TRUE в FALSE): сброс выходов в 0.
	xAbort	BOOL	TRUE: ФБ сразу прерывает работу, все его выходы устанавливаются в значения по умолчанию.
	udiTimeOut	UDINT	Определяет время (µs), по истечении которого ФБ прерывает работу из-за таймаута с сообщением об ошибке
Output	xDone	BOOL	TRUE: ФБ успешно завершил работу
	xBusy	BOOL	TRUE: ФБ в состоянии выполнения
	xError	BOOL	TRUE: Произошла ошибка, ФБ прекращает выполняться FALSE: Ошибок не произошло
	xAborted	BOOL	TRUE : Работа ФБ прервана пользователем
Input	hCom	CAA.HANDLE	Обработчик открытого COM порта (предварительно получен как результат выполнения COM.Open)
	pBuffer	CAA.PVOID	Указатель на буфер для сохранения считанных данных из COM порта
	szBuffer	CAA.SIZE	Максимальный размер в байтах буфера для считываемых данных
Output	eError	ERROR	Ошибка выполнения ФБ
	szSize	CAA.SIZE	Возвращает количество принятых в pBuffer байт данных

ФБ COM.Close

Scope	Name	Type	Comment
Input	xExecute	BOOL	Нарастающий фронт (перевод из FALSE в TRUE): ФБ начинает выполняться Падающий фронт (из TRUE в FALSE): сброс выходов в 0
Output	xDone	BOOL	TRUE: ФБ успешно завершил работу
	xBusy	BOOL	TRUE: ФБ в состоянии выполнения

	xError	BOOL	TRUE: Произошла ошибка, ФБ прекращает выполняться FALSE: Ошибок не произошло
Input	hCom	CAA.HANDLE	Обработчик открытого COM порта (предварительно получен как результат выполнения COM.Open)
Output	eError	ERROR	Ошибка выполнения ФБ

Работу с COM портом удобнее всего описать по шагам с использованием оператора CASE:

```
CASE eState OF
    INIT: [10 lines]
    OPEN:   // открываем COM порт [20 lines]
    WRITE:  // выполняем запись в COM порт [32 lines]
    READ:   // выполняем чтение из COM порта [28 lines]
    CLOSE:  // закрываем порт [15 lines]
END_CASE
```

Рис. 2. Содержание основной программы проекта

Шаг INIT отвечает за сброс используемых функциональных блоков и обнуление вспомогательных переменных. Остальные шаги соответствуют вызываемым ФБ библиотеки CAA SerialCOM.

Перед вызовом ФБ COM.Open нужно сформировать настройки COM порта, которые затем передаются на вход ФБ, — pParameterList (рис. 3).

```
// параметры COM-порта
aParamsB57600 : ARRAY [1..7] OF COM.PARAMETER := [
    (udiParameterId := COM.CAA_Parameter_Constants.udiPort,          udiValue := 2),
    (udiParameterId := COM.CAA_Parameter_Constants.udiBaudrate,      udiValue := 57600),
    (udiParameterId := COM.CAA_Parameter_Constants.udiParity,        udiValue := COM.PARITY.NONE),
    (udiParameterId := COM.CAA_Parameter_Constants.udiStopBits,      udiValue := COM.STOPBIT.ONESTOPBIT),
    (udiParameterId := COM.CAA_Parameter_Constants.udiTimeout,       udiValue := 0),
    (udiParameterId := COM.CAA_Parameter_Constants.udiByteSize,      udiValue := 8)
];
```

Рис.3. Настройка параметров COM порта

Вызов экземпляра ФБ COM.Open показан на рис. 4. В качестве входных параметров вызова указываются настройки порта pParameterList и количество настроечных параметров usiListLength.

```
OPEN:   // открываем COM порт
    fbCOM_Open(
            xExecute := TRUE,
            usiListLength   :=  UINT_TO_USINT(SIZEOF(aParamsB57600) / SIZEOF(COM.PARAMETER)),
            pParameterList  :=  ADR(aParamsB57600)
            );

    hCOM := fbCOM_Open.hCom;    // получаем дескриптор COM порта
```

Рис.4. Вызов ФБ открытия COM порта

Результатом является обработчик открытого порта.

Для передачи данных в открытый COM порт используем экземпляр ФБ COM.Write. Предварительно потребуется сформировать буфер данных для записи. Поскольку данные в нашем примере отправляются в NLS-16DI, работающий по протоколу DCON, сформируем в символьном виде команду чтения имени модуля:

^09Mcr → 5Eh 30h 39h 4Dh 0Dh

Здесь cr - возврат каретки (ASCII код 0Dh). Пример вызова ФБ для записи в порт приведен на рис. 5.

```
WRITE:  // выполняем запись в COM порт
    // подготовим данные для отправки в COM порт
    // ^09Mcr - считать RLDA имя модуля
    byWriteData[0]:=16#5E;
    byWriteData[1]:=16#30;
    byWriteData[2]:=16#39;
    byWriteData[3]:=16#4D;
    byWriteData[4]:=16#0D;

    // запись в COM порт
    fbCOM_Write(
                xExecute :=  TRUE,
                hCom     :=  hCOM,
                pBuffer  :=  ADR(byWriteData),
                szSize   :=  5
                );
```

Рис.5. Вызов ФБ записи в COM порт

При вызове экземпляра ФБ COM.Write требуется указывать обработчик открытого COM порта, начальный адрес буфера данных для записи и количество байт данных для записи.

Чтобы прочитать ответ модуля, нужно выполнить вызов экземпляра ФБ COM.Read, указав во входных параметрах обработчик открытого порта, начальный адрес буфера, куда будут записаны считанные данные, и максимальный размер буфера.

```
READ:   // выполняем чтение из COM порта
    fbCOM_Read(
                xExecute :=  TRUE,
                hCom     :=  hCOM,
                pBuffer  :=  ADR(byReadData),
                szBuffer :=  SIZEOF(byReadData)
                );
```

Рис.6. Вызов ФБ чтения из COM порта

В результате работы данного ФБ в массив byReadData был записан ответ модуля (см. рис 7), что соответствует символам !09NL16DI2cr

byReadData	ARRAY [0..19] OF BYTE	
byReadData[0]	BYTE	16#21
byReadData[1]	BYTE	16#30
byReadData[2]	BYTE	16#39
byReadData[3]	BYTE	16#4E
byReadData[4]	BYTE	16#4C
byReadData[5]	BYTE	16#31
byReadData[6]	BYTE	16#36
byReadData[7]	BYTE	16#44
byReadData[8]	BYTE	16#49
byReadData[9]	BYTE	16#32
byReadData[10]	BYTE	16#0D
byReadData[11]	BYTE	16#00

Рис.7. Вызов ФБ чтения из COM порта

Для завершения работы с портом используем экземпляр ФБ COM.Close (см. рис. 8). При вызове потребуется указать обработчик открытого порта.

```
CLOSE:  // закрываем порт
    fbCOM_Close(xExecute := TRUE, hCOM := hCOM);
```

Рис.8. Вызов ФБ чтения из COM порта

В проекте используется функция _BUFFER_CLEAR() библиотеки OSCAT Basic 3.31 для очистки байтового массива для хранения считанных из COM порта данных.