Справочное руководство по настройке обмена данными между МК и дисплеем Nextion в среде FLProg с помощью блоков ecoins

---

Руководство пользователя
2022

Оглавление

1. Общие положения

Даное руководство составлено и обобщено на основе информации, изложенной в теме форума сообщества FLProg, посвященной разработкам ecoins для дисплея Nextion.

Использование штатных блоков FLProg для обмена данными между микроконтроллером (МК) и дисплеем Nextion (Nxt) имеет свои недостатки, а именно: значительное падение производительности МК при наличии хотя бы одного блока для работы с Nxt.

Были проведены эксперименты с альтернативными решениями (создание своих блоков для отправки текстовых команд). Примеры таких решений можно посмотреть здесь и здесь. Результаты оказались не очень приемлемыми как по производительности, так и по удобству применения, особенно при большом количестве переменных.

Командой ecoins была проведена разработка своего протокола обмена:

1. **Из контроллера в дисплей (NXT)** в текстовой форме: va0.val=0x12FA45DE, или va1.val=5 --> Запись in32t_t в NXT (без контроля достоверности). В NXT должны быть зарезервированы переменные va0,va1...и так с запасом, скажем до 32 (размер памяти позволяет). Если передается не существующая переменная, то запись игнорируется.

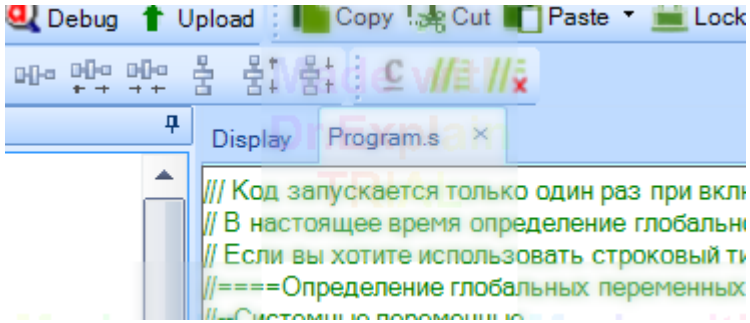
2. **Из дисплея в контроллер** упрощенное подобие ModBus ASCII:
1-байт - идентификатор начала посылки =':'
2-байт - тип посылки =1 передача байт; =2 передача uint16_t; =3 передача uint32_t;
3-байт - кол-во данных
.... передаваемые данные
n-байт - контрольная сумма
n+1 13 10 (коды завершения посылки).
В блоке FLProg должны будут зарезервированы необходимые буфера. Если данных больше чем буфер, то лишние данные отбрасываются.
Разработанный протокол - компромисс между скоростью и надежностью обмена, а также простотой реализации (в NXT большие программы писать не очень удобно).
---
3. Дисплей посылает данные в контроллер , если они данные изменились или периодически(например 1раз в сек) для поддержания активного обмена.
Контроллер работает аналогично.
Устройства не ожидают подтверждения об отправленных посылок и потому обмен быстрый.
Прием и передача (RX,TX) независимы.

Для настройки протокола обмена необходимо:

1. Прописать конфигурацию протокола в дисплее (вкладка *Program.s* в среде программирования Nextion Editor)
2. Использовать соответсвующую библиотеку ecoins (см. ссылку на раздел форума)
3. Сконфигурировать блоки ecoins отправки и приема данных в FLProg
4. Прописать задачи в дисплее Nextion.

2. Конфигурация дисплея

Конфигурация прописывается во вкладке Program.s в Nextion Editor



В общем виде она выглядит следующим образом:

```
//========================================================================================
//                    1.СИСТЕМНЫЕ ПЕРЕМЕННЫЕ РЕДАКТОРА Nextion
// sys0,sys1,sys2 - переменные для использования пользователем;
// dp     - код текущей страницы;
// dim    - уровень подсветки (0-100);
// thc    - цвет кисти для рисования. Допустимые значения BALACK,BLUE,BROWN,GREEN,YELLOW,RED,GRAY,WHITE, числовое значение RGB565;
// bkcmd- управление возвращаемыми данными (0,1,2,3);
// sleep- переход в режим сна(=1);
// wup    - страница после выхода из режима сна;
// ussp    - интервал в секундах по истечении которого, при отсутствии данных из UART, произойдет переход в режим сна (0-отключено,3-65535сек);
// thsp    - интервал в секундах по истечении которого, при отсутствии касания экрана  произойдет переход в режим сна (0-отключено,3-65535сек);
// thup    - выход из режима сна при отсутствии касания: =0-не выходить из сна при касании; =1-выход из сна при касании;
// thc2,tch3 - последнии координаты (x,y) точки касания (только на чтение);
//-------Параметры для дисплев с расширенными возможностями(K,P)----------------------------------
// rtc0,rtc1,rtc2,rtc3,rtc4,rtc5,rtc6-год,месяц,число,час,минута,секунда,день недели часов реального времени;
//========================================================================================
int sys0=0,sys1=0,sys2=0         //--Пользовательские значения;
//========================================================================================
```

```
//                  2.ОБЩИЕ ПОЛЬЗОВАТЕЛЬСКИЕ ПЕРЕМЕННЫЕ
//=============================================================================================
int agoPage                         //--Номер предыдущей страницы (для фиксации изменения номера страницы);
int agoSec,changeTime               //--Буфер для контроля изменения времени (по rtc5(секунды) или rtc4(мин);
int vReadySystem                    //--Флаг готовности для отправки системных значений из Nextion в MCU;
int userSys0,userSys1               //--Зарезервированные пользовательские переменные;
int vBlink,cntBlink,numBlink        //--Переменные для мерцания(анимации) выбранным изображением на каждой страницы;
int buf,buf1,buf2,buf3              //--Рабочие буфера;
int idx,idx1,idx2,idx3              //--Рабочие индексы;
//---------------------------------------------------------------------------------------------
int idGrafic,chanelGrafic       //--id и канал(0-3) графика;
int cmdByte,cmdLong                 //--Рабочие команды для работы с графиками;
int dataByte,dataLong,dataRun   //--Рабочие параметры для работы с графиками;
int cmndGrafic,limGrafic,valGrafic //--Рабочие параметры для построения графиков;

//=============================================================================================
//                  3.ПОЛЬЗОВАТЕЛЬСКИЕ ПЕРЕМЕННЫЕ
//              для формирования транзакций из Nextion в контроллер
//                    (протокол - клон ModBus)
//=============================================================================================
int head       //--Заголовок посылки
int func      //--Функция: 6-отправка регистра,16 отправка нескольких регистров, 56-отправка системных регистров;
int adr       //--Буфер для адреса регистра отправляемой посылки (0xFFFF);
int qntReg  //--Кол-во отправляемых регистров(0xFF)
int qntByte //--Кол-во отправляемых байт(0xFF)
int crc       //--Контрольная сумма отправляемой посылки (0xFF);
//---------------------------------------------------------------------------------------------
int cmdPack    //--Команда управления отправки посылки;
int cntPack
//=============================================================================================

//=============================================================================================
//                  3.ВХОДНЫЕ ПОЛЬЗОВАТЕЛЬСКИЕ ПЕРЕМЕННЫЕ
```

```
//      (для записи параметров из MCU в Nextion командами inx=123456789 или inx=0xFFFFFFFF)
// ПРИМЕЧАНИЕ: кол-во параметров и их имена могут быть измененены пользователем под конкретный проект.
//=========================================================================================
int in0,  in1,  in2,  in3,  in4,  in5,   in6,   in7,   in8,  in9
int in10, in11, in12, in13, in14, in15,  in16,  in17,  in18, in19
int in20, in21, in22, in23, in24, in25,  in26,  in27,  in28, in29
int in30, in31, in32, in33, in34, in35,  in36,  in37,  in38, in39
int in40, in41, in42, in43, in44, in45,  in46,  in47,  in48, in49
int in50, in51, in52, in53, in54, in55,  in56,  in57,  in58, in59
//=========================================================================
//              4.ИНСТРУКЦИИ ДЛЯ РАБОТЫ С ГРАФИКАМИ
// 1.Управление байтами через переменную grB(xxB):
//    grB=0xYYZZ, ZZ-данные, YY-команда(=70-заполнение массива, =71-сдвиг массива);
//    Примеры: №1 grB=0x703F; №2 grB=0x7177;
// 2. Управление 4-байтными переменными=grL через переменную grR
//      grR(=70-заполнение массива, =71-сдвиг массива, =90-заполнение треугольником);
//    Примеры: №1 grL=0x703F; grR=0x70
//           №2 grL=0x716D; grR=0x71
//           №3 grR=0x90;
//=========================================================================================

//=========================================================================
//                5.ИСХОДНЫЕ ДАННЫЕ ДЛЯ ИНДИВИДУАЛЬНОГО ГРАФИКА gr
//=========================================================================
int  grSize=512   //--Количество точек в графике. Рекомендуется выбирать кратным 64 (128,192,256,320,384,448,512 и т.д.);
int  grB=0,grL=0,grR=0,grEnd //--Параметры управления(grB,grL,grR), вспомогательный параметр GrEnd
int  gr0,gr1,gr2,gr3,gr4,gr5,gr6,gr7,gr8,gr9,gr10,gr11,gr12,gr13,gr14,gr15
//=========================================================================

//=========================================================================
//           10.ПОЛЬЗОВАТЕЛЬСКИЕ ПАРАМЕТРЫ ДЛЯ РАБОТЫ СО СТРАНИЦАМИ
//ПРИМЕЧАНИЕ: кол-во параметров и их имена могут быть измененены пользователем под конкретный проект.
//=========================================================================
```

```
//-------------------------------------------------------------------------
//            10.0.ПОЛЬЗОВАТЕЛЬСКИЕ ПАРАМЕТРЫ ДЛЯ СТРАНИЦЫ 0
//-------------------------------------------------------------------------
int  out00, out01, out02, out03 ,out04, out05, out06, out07, out08, out09
int  ago00, ago01, ago02, ago03, ago04, ago05, ago06, ago07, ago08, ago09
//-------------------------------------------------------------------------
//            10.1.ПОЛЬЗОВАТЕЛЬСКИЕ ПАРАМЕТРЫ ДЛЯ СТРАНИЦЫ 1
//-------------------------------------------------------------------------
int  out10, out11, out12, out13, out14, out15, out16, out17, out18, out19
int  ago10, ago11, ago12, ago13, ago14, ago15, ago16, ago17, ago18, ago19
//-------------------------------------------------------------------------
//            10.2.ПОЛЬЗОВАТЕЛЬСКИЕ ПАРАМЕТРЫ ДЛЯ СТРАНИЦЫ 2
//-------------------------------------------------------------------------
int  out20, out21, out22, out23, out24, out25, out26, out27, out28, out29
int  ago20, ago21, ago22, ago23, ago24, ago25, ago26, ago27, ago28, ago29
//-------------------------------------------------------------------------
//            10.3.ПОЛЬЗОВАТЕЛЬСКИЕ ПАРАМЕТРЫ ДЛЯ СТРАНИЦЫ 3
//-------------------------------------------------------------------------
int  out30, out31, out32, out33, out34, out35, out36, out37, out38, out39
int  ago30, ago31, ago32, ago33, ago34, ago35, ago36, ago37, ago38, ago39
//=========================================================================================
//                    20.ОБЩИЙ КОД ПРОГРАММЫ
//Примечание: во избежании редких случаев настройки скорости рекомендуется
//           в секции прединициализации page 0 установить параметр bauds=11500
//=========================================================================================
dim=100                          //--Установка яркости;
thc=BLUE                        //--Цвет кисти для рисования;
bkcmd=2                          //--Отправлять данные только когда последняя команда НЕ была выполнена;
//-----------------------------------------------------------------------------------------
wup=255                          //--После выхода из сна будет загружена последняя страница;  wup=0 -- после пробуждения будет загружена страница с I
ussp=0                          //--Переход в режим сна при отсутствии данных из UART отключен;
thsp=0                          //--Переход в режим сна при отсутствии касания экрана отключен;
```

```
thup=1                          //--Выход из сна при касании;
//-------------------------------------------------------------------------------------------
page 0                          //--Установка начальной страницы;
baud=115200                         //--Установка скорости обмена;
//=============================================================================================
```

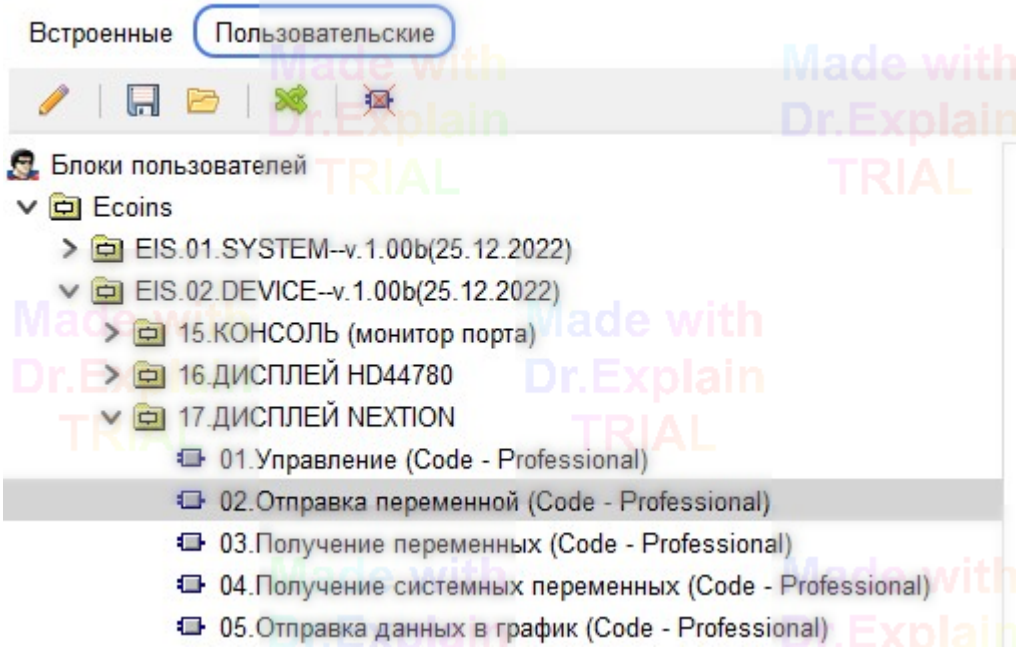
Здесь определяются системные и пользовательские параметры работы.

Количество переменных условно не ограничено; Главное условие - количество пользовательских переменных (in0, in1, in2... , out00, out01, out02... , ago00, ago01, ago02...) должно быть не меньше чем используется в вашем проекте. Имена переменных могут быть изменены; работа программы Nxt производитеся по этим именам переменных.

В общем случае, можно скопировать и вставить всю эту инструкцию без изменения.

3. Отправка данных из МК в Nextion

Обмен данными между МК и Nxt осуществляется с помощью пользовательских блоков ecoins для дисплея Nextion.

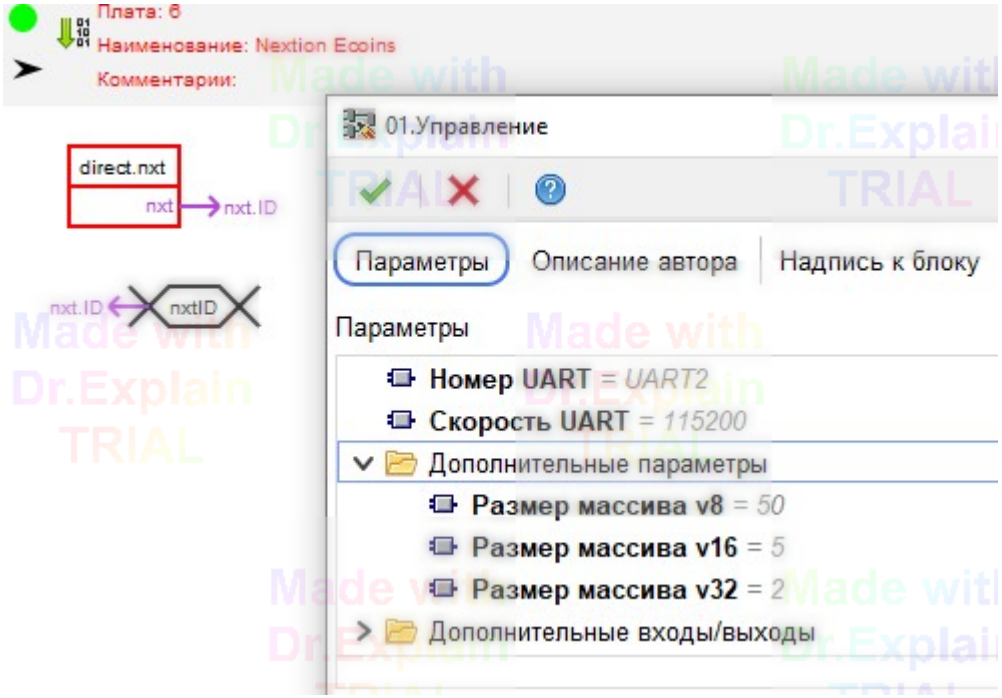


Необходимо перенести блок "Управление" (direct.nxt) на плату и настроить его.

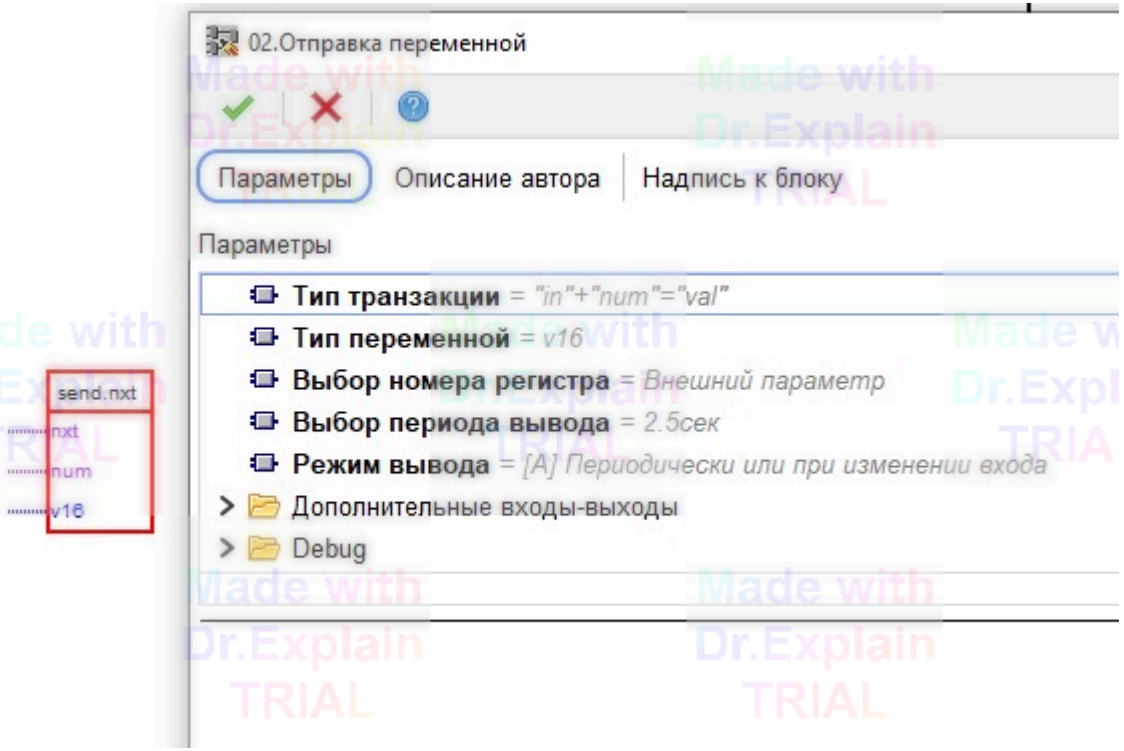
• Указать номер последовательного порта

• Указать скорость обмена данными (скорость должна совпадать со скоростью в конфигурации дисплея Nxt (*baud=115200 //--Установка скорости обмена;*))

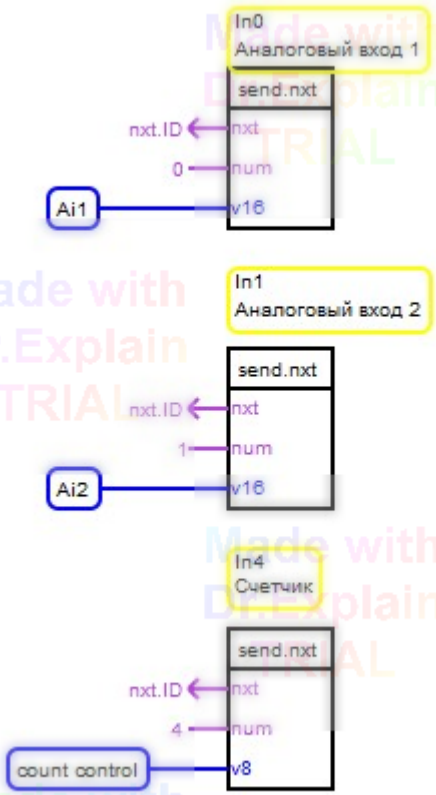
• Можно изменить размеры массивов данных, если это требуется. Массив - это количество переменных того или иного типа данных, которые используются при передаче.



Вынести блок "Отправка переменной" (send.nxt) на плату.
В основном, достаточно настроек по умолчанию.
Возможно, потребуется изменить тип переменной (v8 - byte, v16 - word, v32 - double word) и период вывода данных.



Соединить входы **nxt** блоков **send.nxt** с выходом **nxt** блока **direct.nxt**.
num - это номер переменной (регистра) в пакете отправки. В базовом случае, для in0 num=0, in1 num=1 и т.д.
На вход v8 (v16, v32) подается передаваемое значение.



На этом конфигурация пакета отправки в FLProg завершена.

3.1. Прием данных в Nextion

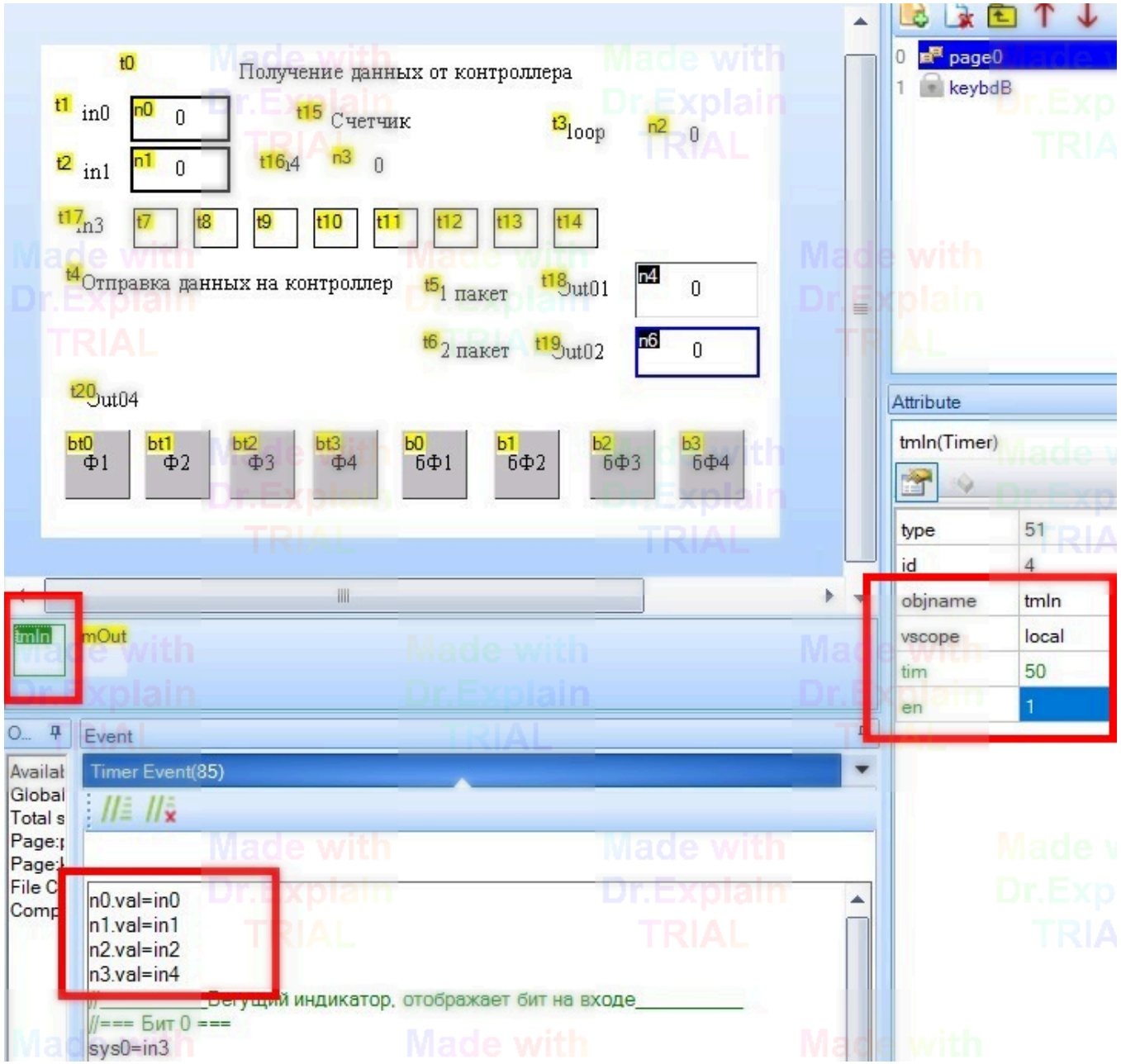
Для приема переменных в Nxt необходимо на страницах, где будут задействованы принимаемые переменные, создать таймеры.

Для удобства работы, назовем **tmIn** (timer input).

Период таймера 50 мс, но время может быть изменено по усмотрению пользователя. Это период опроса нновых посылок. Делать чаще не имеет смысла, больше 1000мс - будет видна задержка обновления данных.

В поле событий таймера **Timer Event** необходимо указать, какие входящие переменные каким элементам дисплея соответствуют.

Это может быть значение числового параметра, текст, значение цвета элемента, id картинки и т.п.


t0
Получение данных от контроллера
t1 in0
n0 0
t15 Счетчик
t3 loop
n2 0
t2 in1
n1 0
t16 4
n3 0
t17 in3
t7 t8 t9 t10 t11 t12 t13 t14
t4 Отправка данных на контроллер
t5 1 пакет
t18 Out01
n4 0
t6 2 пакет
t19 Out02
n6 0
t20 Out04
bt0 Ф1
bt1 Ф2
bt2 Ф3
bt3 Ф4
b0 бФ1
b1 бФ2
b2 бФ3
b3 бФ4
page0
keybdB
Attribute
tmIn(Timer)
type 51
id 4
objname tmIn
vscope local
tim 50
en 1
tmIn
Event
Timer Event(85)
n0.val=in0
n1.val=in1
n2.val=in2
n3.val=in4
//_____Бегущий индикатор, отображает бит на входе_____
//=== Бит 0 ===
sys0=in3

Прием битовых состояний будет описан в разделе 5

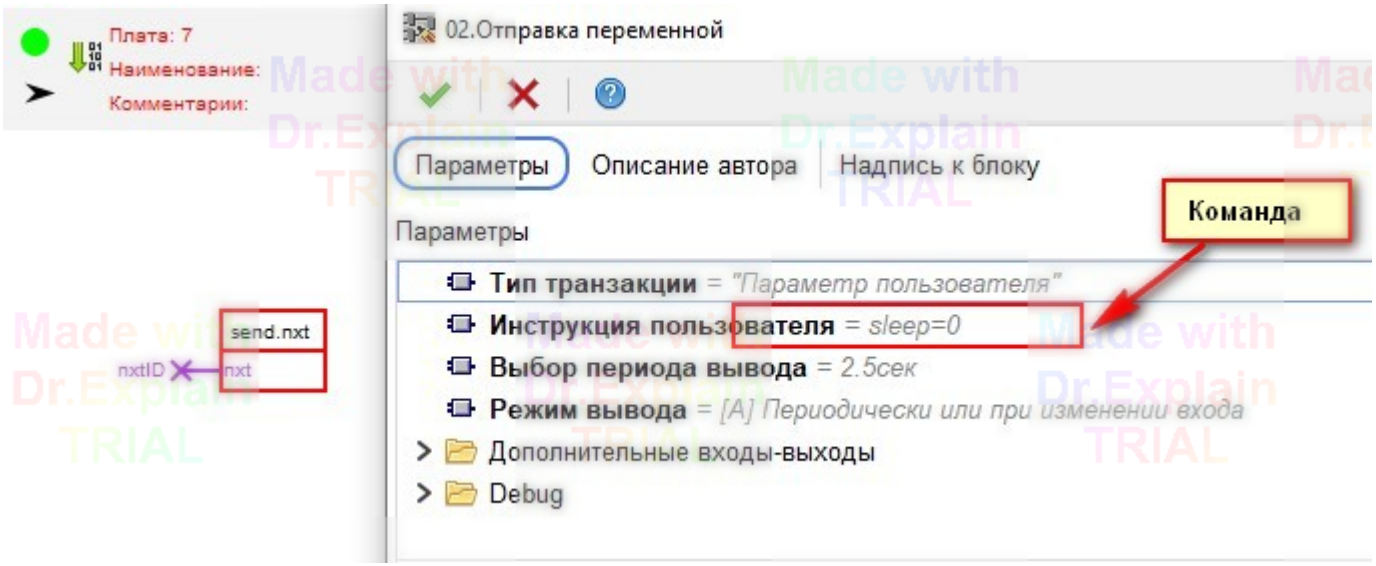
3.2. Отправка текстовой команды

Иногда может пригодиться отправка из МК в Nxt текстовой команды.

В частности, у меня была необходимость выводить дисплей из состояния сна по внешней команде (не касанием дисплея).

Для этого в отдельной плате, вызываемой по условию, нужно разместить блок send.nxt и в нем прописать команду как внешний параметр.

(команда на пробуждение дисплея **sleep=0**)

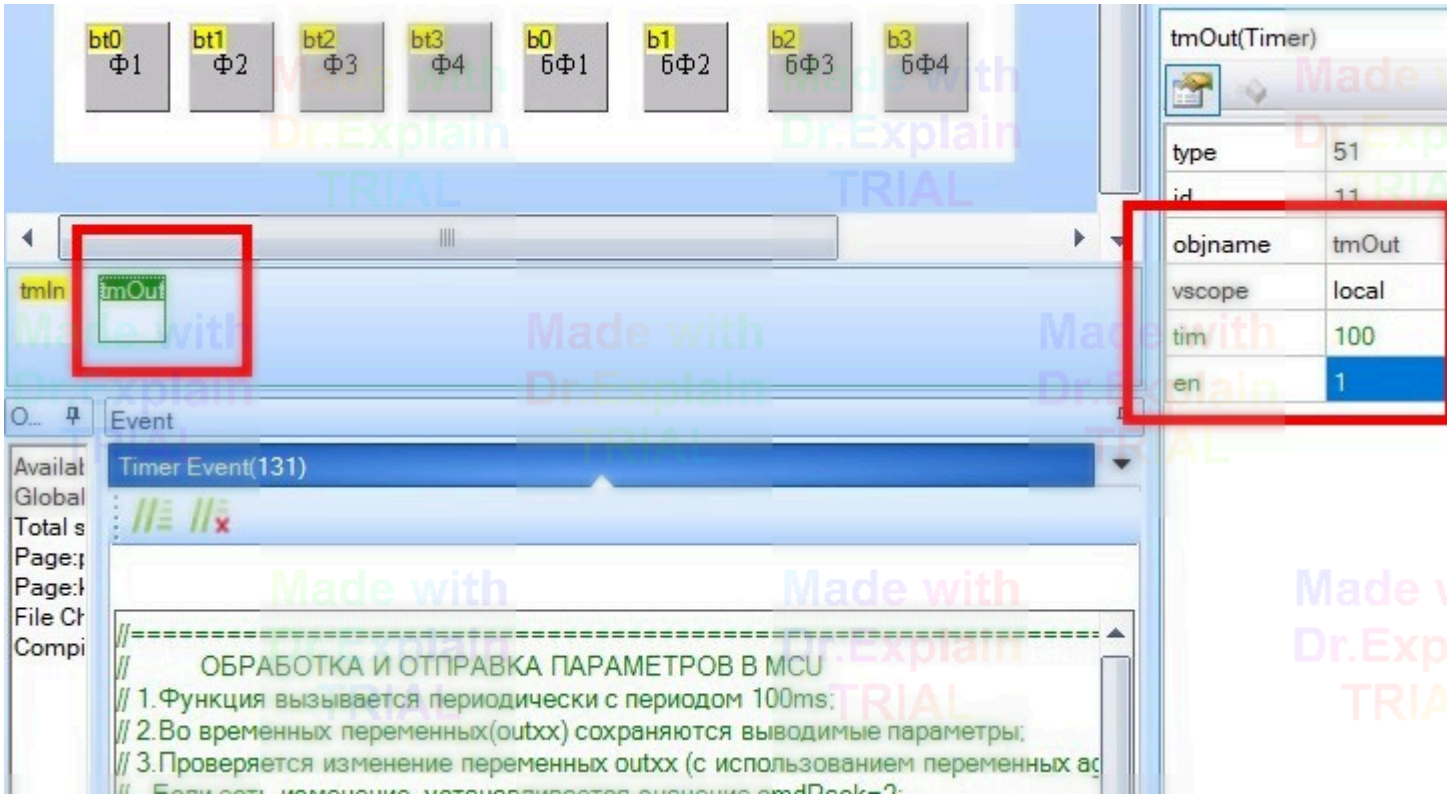


4. Отправка данных из Nextion в МК (код задачи Main)

Для отправки переменных из Nxt необходимо на страницах, где будут задействованы передаваемые переменные, создать таймеры отправки.

Для удобства работы, назовем **tmOut** (timer output).

Период таймера 100 мс, это время привязано к периодичности отправки (см. далее).



В поле событий таймера **Timer Event** необходимо прописать следующий код (**Main**):

```
//=============================================================================================
```

```
//         ОБРАБОТКА И ОТПРАВКА ПАРАМЕТРОВ В MCU
// 1.Функция вызывается периодически с периодом 100ms;
// 2.Во временных переменных(outxx) сохраняются выводимые параметры;
// 3.Проверяется изменение переменных outxx (с использованием переменных agoxx).
//   Если есть изменение, устанавливается значение cmdPack=2;
// 4.Проверяется счетчик cntPack>10 для контроля достижения 1сек.
//   Если cntPack>10, устанавливается значение cmdPack=1;
//=============================================================================================
//=============================================================================================
//      1.СОХРАНЕНИЕ ВЫВОДИМЫХ ПАРАМЕТРОВ В ПОЛЬЗОВАТЕЛЬСКИХ ПЕРЕМЕННЫХ outxx
//=============================================================================================
//out00=bt0.val
out01=n4.val
out02=n6.val
out03=7
// out04 -устанавливается по событиям (нажатие) кнопки (4 с фиксацией, 4 без фиксации)
//=============================================================================================
//      2.ПРОВЕРКА НА ИЗМЕНЕНИЕ ПЕРЕМЕННЫХ outxx
//=============================================================================================
if(ago00!=out00)
{
  cmdPack=2
  ago00=out00
}
//------
if(ago01!=out01)
{
  cmdPack=2
  ago01=out01
}
//------
if(ago02!=out02)
{
```

```
  cmdPack=2
  ago02=out02
}
//------
if(ago04!=out04)
{
  cmdPack=2
  ago04=out04
}
//=============================================================================================
//      3.ПРОВЕРКА НА ДОСТИЖЕНИЕ ВРЕМЕНИ 1 СЕК ПРИ ОТСУТСТВИИ ИЗМЕНЕНИЯ outxx
//=============================================================================================
cntPack++
if(cntPack>=10)     //-- Увеличение счетчика с каждым проходом таймера, 100мс * 10 проходов = 1с
{
  cmdPack=1
}
//=============================================================================================
//      4.ОТПРАВКА ПЕРЕМЕННЫХ В MCU
//=============================================================================================
if(cmdPack>0)     //--Проверка команды отправки (по изменению или 1 раз в сек)
{
  cmdPack=0       //--Очистка команды управления;
  cntPack=0     //--Очистка счетчика времени;
  head=0x3A       //--Заголовок
  func=0x06       //--Отправка нескольких регистров байтовых регистров
  adr=1           //--Адрес первого отправляемого регистра
  qntReg=2        //--Кол-во отправляемых регистров
  qntByte=2       //--Кол-во отправляемых байт
  crc=head        //--Код заголовка (':') в КС
  crc+=func       //--Параметр func
  crc+=adr        //--Параметра adr
  crc+=qntReg      //-- Параметр qntReg
```

```
  crc+=qntByte     //-- Параметр qntByte
  //----Добавление в crc передаваемых регистров----------------------------------------------------
  crc+=out00
  crc+=out01
  //  crc+=out02
  //  crc+=out03
  //------------------------
  crc=crc&0xFF     //--Нормализация crc до байта:
  //-----Вычисление  crc с двумя дополнениями-------------------------------------------------------
  crc=0xFF-crc
  crc+=1
  crc&=0xFF
 //=============================================================================
//         5. Отправка посылки в UART-----------------------------------------------------------
//=============================================================================
  prints head,1      //--Отправка заголовка;
  prints func,1      //--Отправка кода функции;
  prints adr,1      //-- Адрес первого отправляемого регистра в пакете
  prints qntReg,1   //--Кол-во записываемых регистров;
  prints qntByte,1   //--Кол-во отправляемых байт;
  prints out00,1     //--Отправка байта регистра 1;
  prints out01,1    //--Отправка байта регистра 2;
  //  prints out02,1    //--Отправка байта регистра 3;
  //  prints out03,1    //--Отправка байта регистра 4;
  prints crc,1     //--Отправка байта crc;
  printh FF FF FF   //--Отправка кодов завершения посылки
  //=============================================================================
  //       6.   Отправка второго пакета данных ==================
  //=============================================================================
  head=0x3A       //--Заголовок
  func=0x06       //--Отправка нескольких регистров байтовых регистров
  adr=5           //--Адрес первого отправляемого регистра
  qntReg=3        //--Кол-во отправляемых регистров
```

```
  qntByte=3       //--Кол-во отправляемых байт
  crc=head        //--Код заголовка (':') в КС
  crc+=func       //--Параметр func
  crc+=adr        //--Параметра adr
  crc+=qntReg     //-- Параметр qntReg
  crc+=qntByte    //-- Параметр qntByte
  //----Добавление в crc передаваемых регистров-----------------------------------------------------
  //  crc+=out00
  //  crc+=out01
  crc+=out02
  crc+=out03
  crc+=out04
  //-------------------------
  crc=crc&0xFF    //--Нормализация crc до байта:
  //-----Вычисление  crc с двумя дополнениями-------------------------------------------------------
  crc=0xFF-crc
  crc+=1
  crc&=0xFF
  //-----Отправка посылки в UART--------------------------------------------------------------------
  prints head,1     //--Отправка заголовка;
  prints func,1     //--Отправка кода функции;
  prints adr,1      //--Адрес записи
  prints qntReg,1   //--Кол-во записываемых регистров;
  prints qntByte,1  //--Кол-во отправляемых байт;
  //  prints out00,1    //--Отправка байта регистра 1;
  //  prints out01,1    //--Отправка байта регистра 2;
  prints out02,1    //--Отправка байта регистра 3;
  prints out03,1    //--Отправка байта регистра 4;
  prints out04,1    //--Отправка байта регистра 5;
  prints crc,1    //--Отправка байта crc;
  printh FF FF FF   //--Отправка кодов завершения посылки
}
//=============================================================================================
```

Параметры, которые необходимо изменить:

1.СОХРАНЕНИЕ ВЫВОДИМЫХ ПАРАМЕТРОВ В ПОЛЬЗОВАТЕЛЬСКИХ ПЕРЕМЕННЫХ outxx

В этом разделе указывается, какие значения или переменные на странице соответствуют выходным переменным out**

2.ПРОВЕРКА НА ИЗМЕНЕНИЕ ПЕРЕМЕННЫХ outxx

В этойм разделе необходимо прописать код опредделения изменения значения для каждой передаваемой переменной

```
if(ago00!=out00)    // ago00 - зарезервированная в Program.s переменная, out00 - зарезервированная выходная переменная, которая отправляется в MK
{
  cmdPack=2
  ago00=out00
}
```

код должен выглядеть в точности так, но с вашими переменными/номерами переменных, которые используются на текущей странице.

3.ПРОВЕРКА НА ДОСТИЖЕНИЕ ВРЕМЕНИ 1 СЕК ПРИ ОТСУТСТВИИ ИЗМЕНЕНИЯ outxx

копируется без изменений.

4.ОТПРАВКА ПЕРЕМЕННЫХ В MCU

```
func=0x06      //тип передаваемых данных. 0x06 - byte (v8), 0x16 - word (v16), 0x32 - double word (v32)
adr=1          //--Адрес первого отправляемого регистра
```

Здесь нужно быть внимательным. Если переменные передаются с нескольких страниц, то необходимо следить, чтобы каждая переменная была на своем месте в передаваемом пакете данных. Например, на первой странице передаются **2** переменные: **out00** и **out01**, на второй странице **4** переменных: **out02**, **out03**, **out04**, **out05**, и на третьей странице **1** переменная **out06**. Тогда для первой стрвницы **adr=1**, для второй **adr=3**, для третьей **adr=7**.

Общий вид этой части пакета передачи будет выглядеть так:

adr	1	2	3	4	5	6	7
переменная	out00	out01	out02	out03	out04	out05	out06

```
qntReg=2        //--Кол-во отправляемых регистров
 qntByte=2      //--Кол-во отправляемых байт
```

число - количество передаваемых переменных

```
//----Добавление в crc передаваемых регистров----------------------------------------------------
 crc+=out00
 crc+=out01
```

здесь необходимо в указанном формате прописать имена всех передаваемых переменных (которые отправляются с этой страницы)

```
 5. Отправка посылки в UART
prints out00,1    //--Отправка байта регистра 1;
 prints out01,1   //--Отправка байта регистра 2;
```

так же прописать все отправляемые переменные.

Всё остальное копируется без изменений.

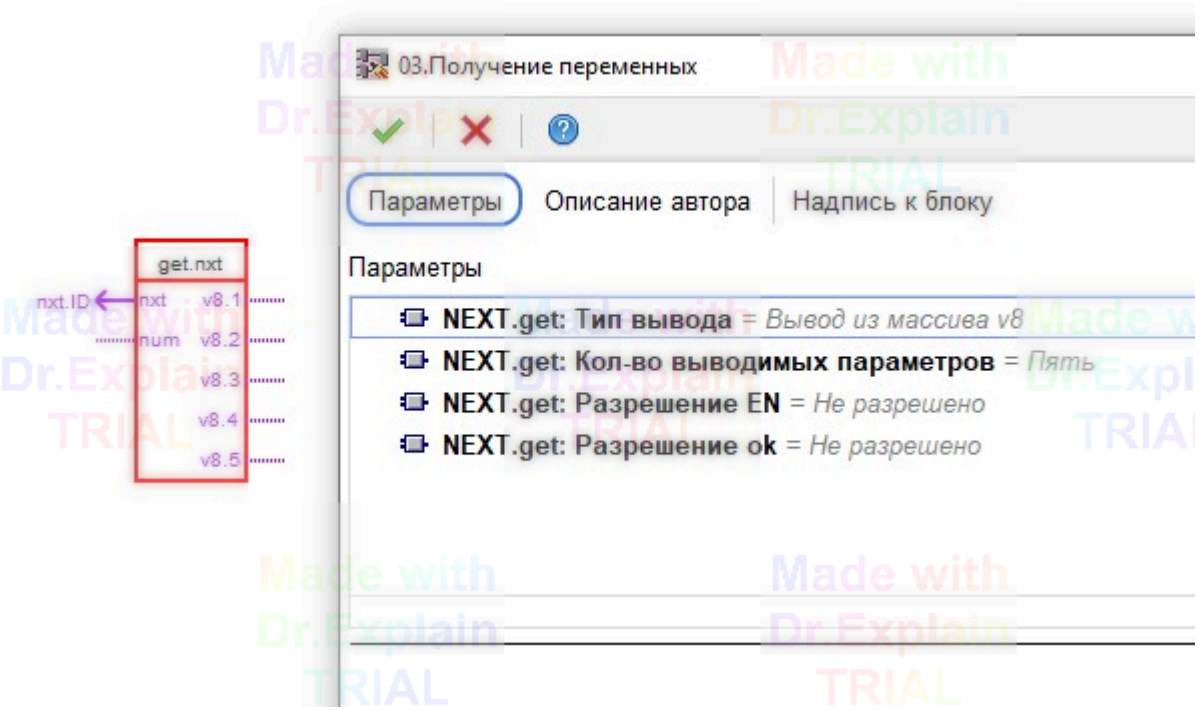
6. Отправка второго пакета данных

Этот раздел приведен для примера, как можно отправить второй пакет данных (например, если необходимо отправить данные другого типа)

Данный код задачи прописывается на каждой странице, где требуется отправка данных из Nxt в МК.

4.1. Прием данных в МК

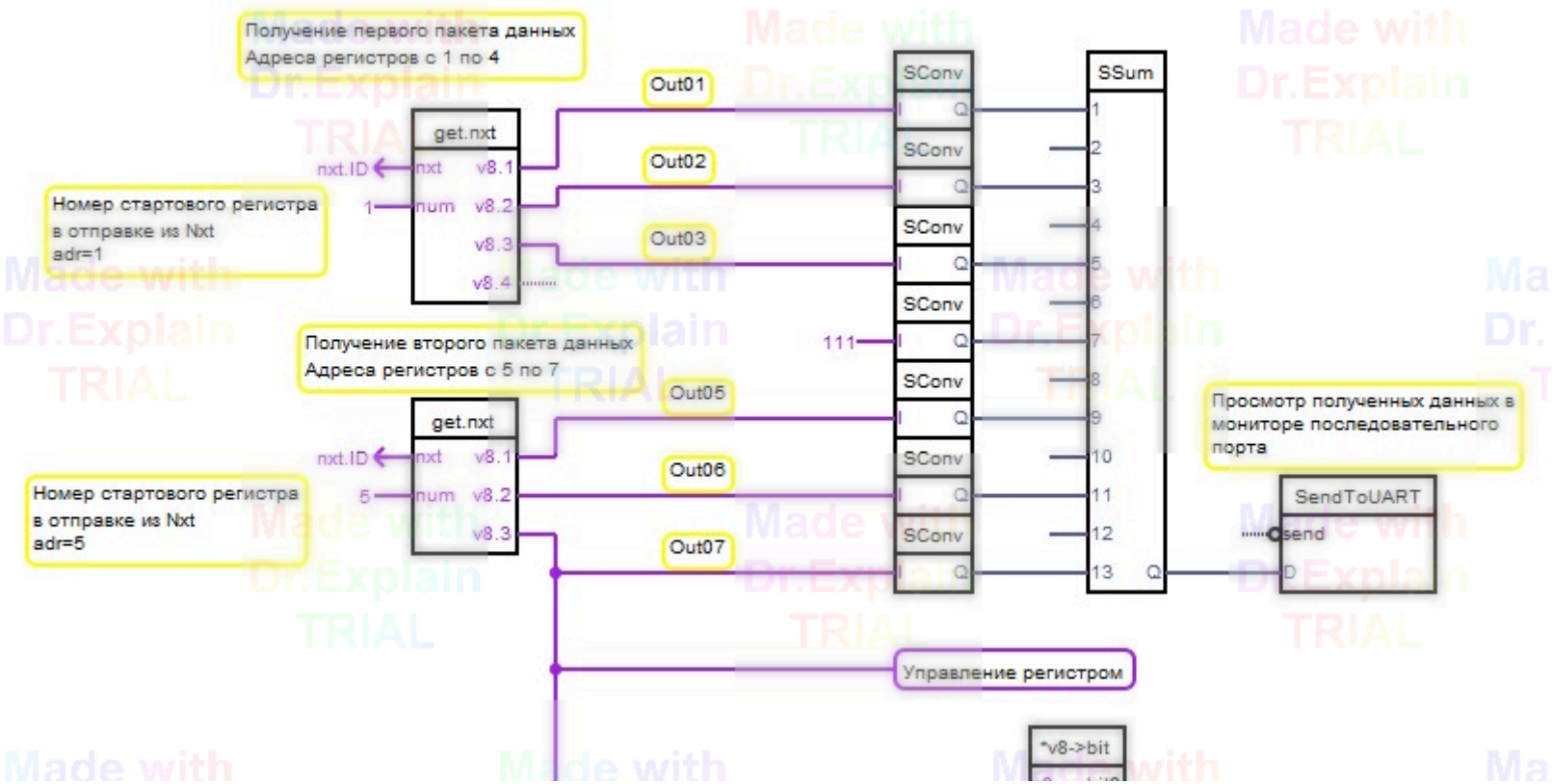
Для получения данных, переданных из Nxt в МК необходимо вынести на плату блок **get.nxt**



Вход блока **nxt** соединить с выходом блока **direct.nxt**

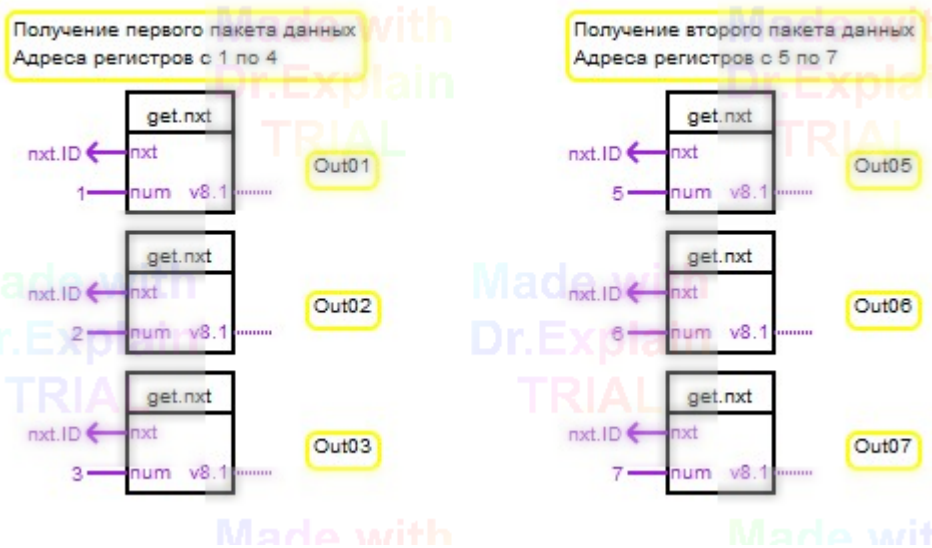
Вход **num** - стартовый адрес, т.е. в блоке выходы v8.1, v8.2 и т.д. будут проиндексированы, начиная с этого адреса.
В общем случае **num** соответствует **adr** в коде задачи Main в Nxt.

В настрйоках блока можно указать тип принимаемых данных (он должен соответствовать типу отправляемых из Nxt данных)
Количество выводимых параметров = количесству переменных, которые принимаются данным блоком.



При увеличении количества переменных можно добавить дополинтельные блоки get.nxt с указанием соответствующего стартового адреса (num). Должна соблюдаться сквозная нумерция переменных. Если адреса переменных (регистров) наложатся, то первая поученная переменная по этому адресу затрётся следующей придедшей по этому же адресу.

Для общего понимания, следующая конфигурация блоков равносильна приведенной выше:



5. Передача битовых состояний

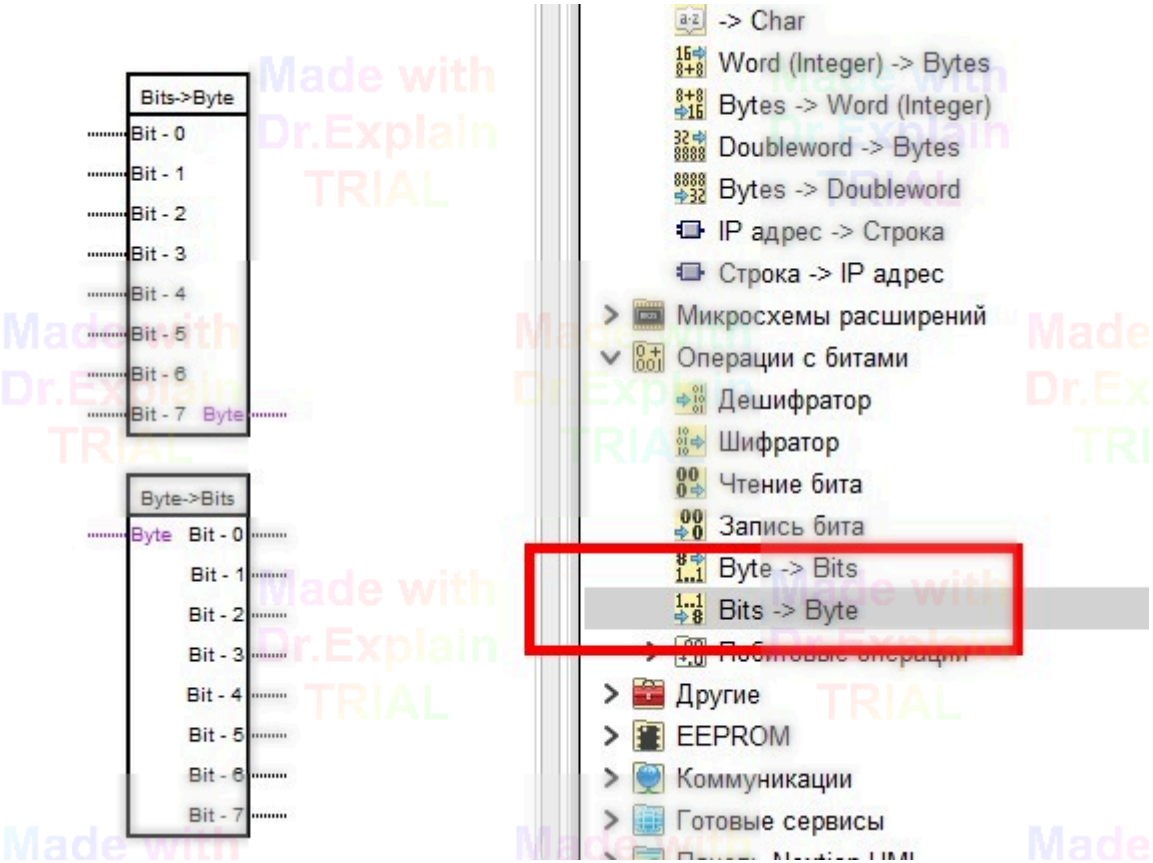
Для передачи состояния булевых переменных (состояния кнопок, переключателей, индикаторов событий и т.п.) следует провести процедуру сборки байта из наборов битов, передать из МК в Nxt или обратно, и на другой стороне разложить байт на составляющие биты.

Эти процедуры отписаны в разделах 5.1 Прием/передача в МК и 5.2 Прием/передача в Nextion

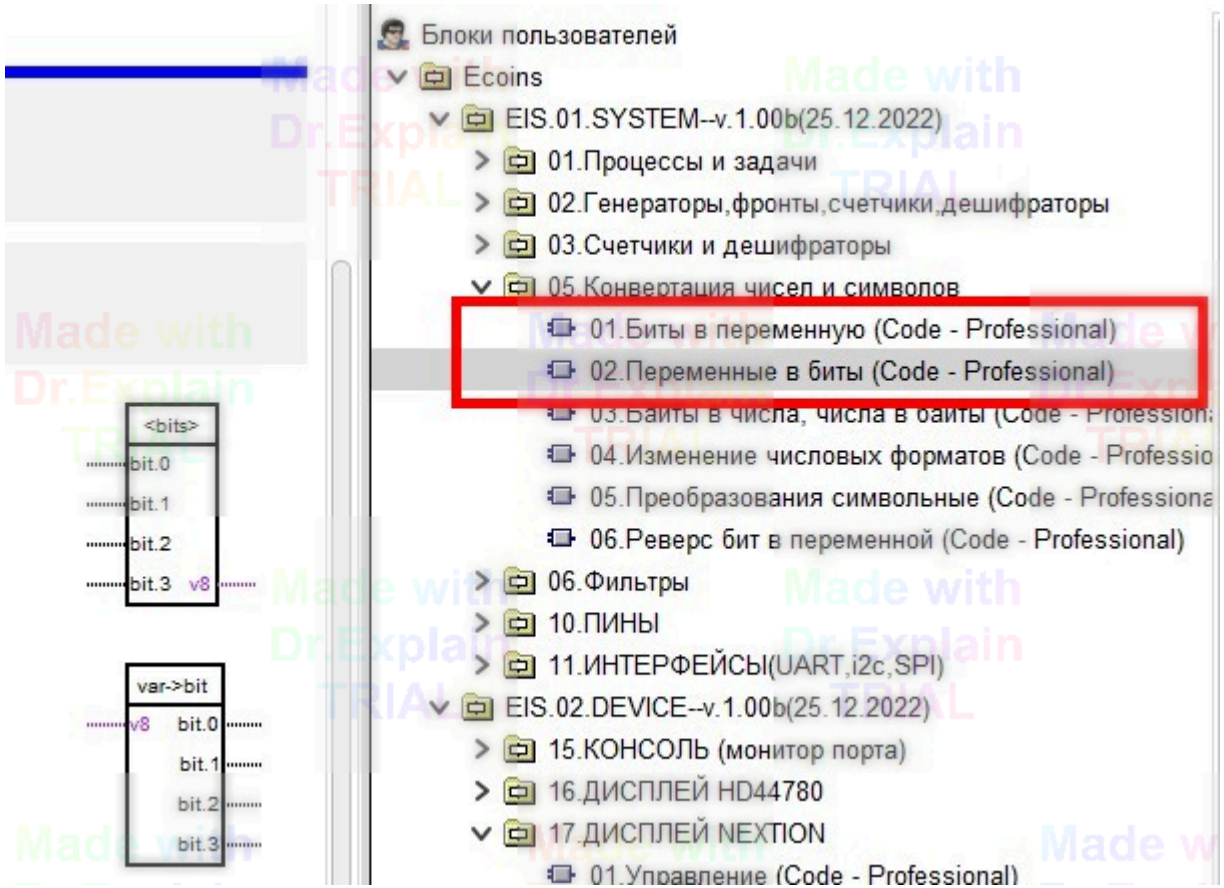
5.1. Прием/передача в МК

Сборка и отправка байта в контроллере

Сборка байта и разложение байта на биты в FLProg реализцется достаточно просто.
Используются или штатные блоки работы с битовыми операциями

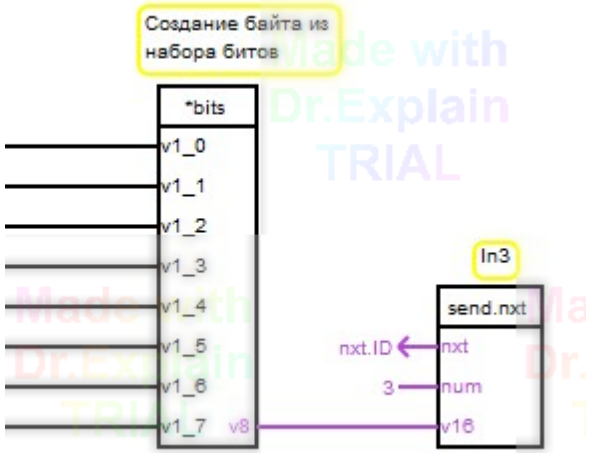


Или сторонние пользовательские блоки. Например, ecoins



Для сборки байта необходимо на входы блока **Bits->Byte** или **<bits>** подключить фискретные сигналы или переменные, и на выходе получим byte.

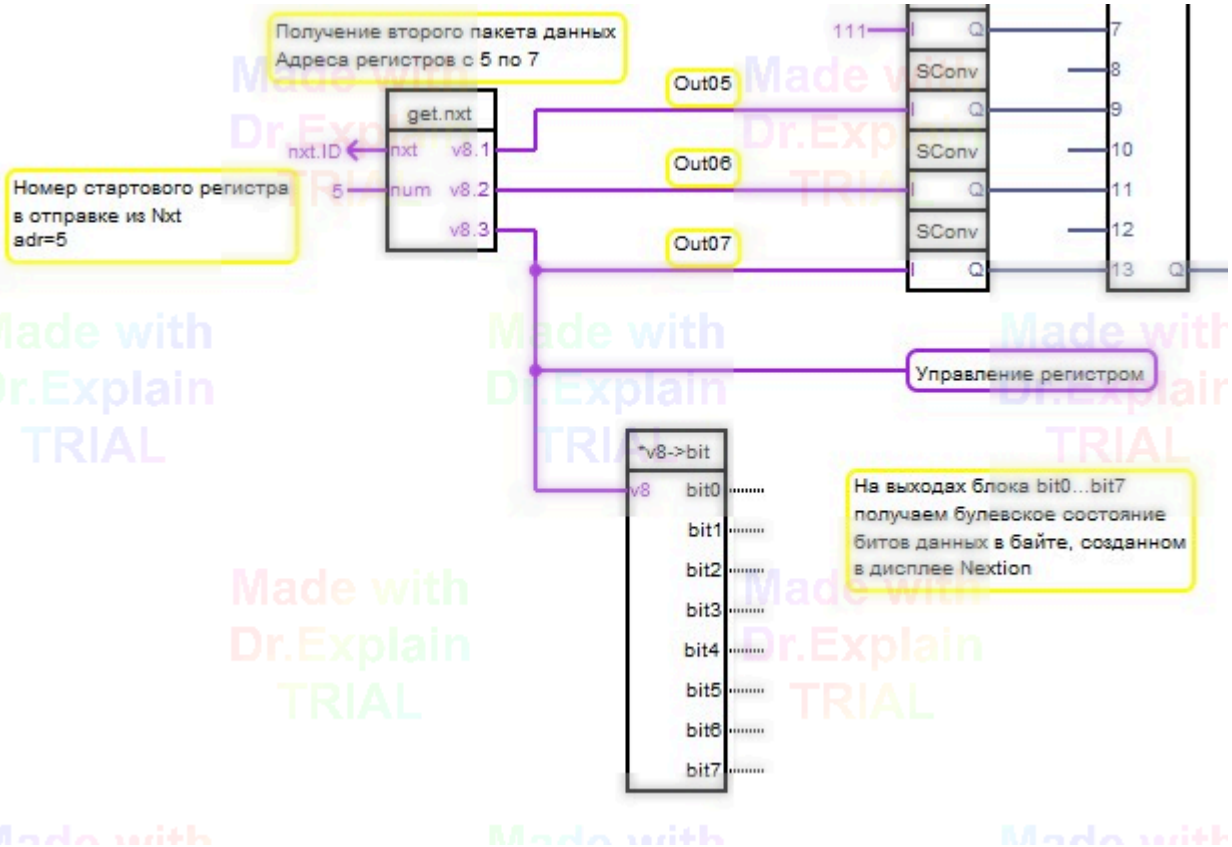
Его и отправляем на блок **send.nxt**.



Получение байта и его разложение на биты

Процедура разложения обратна предыдущей операции.

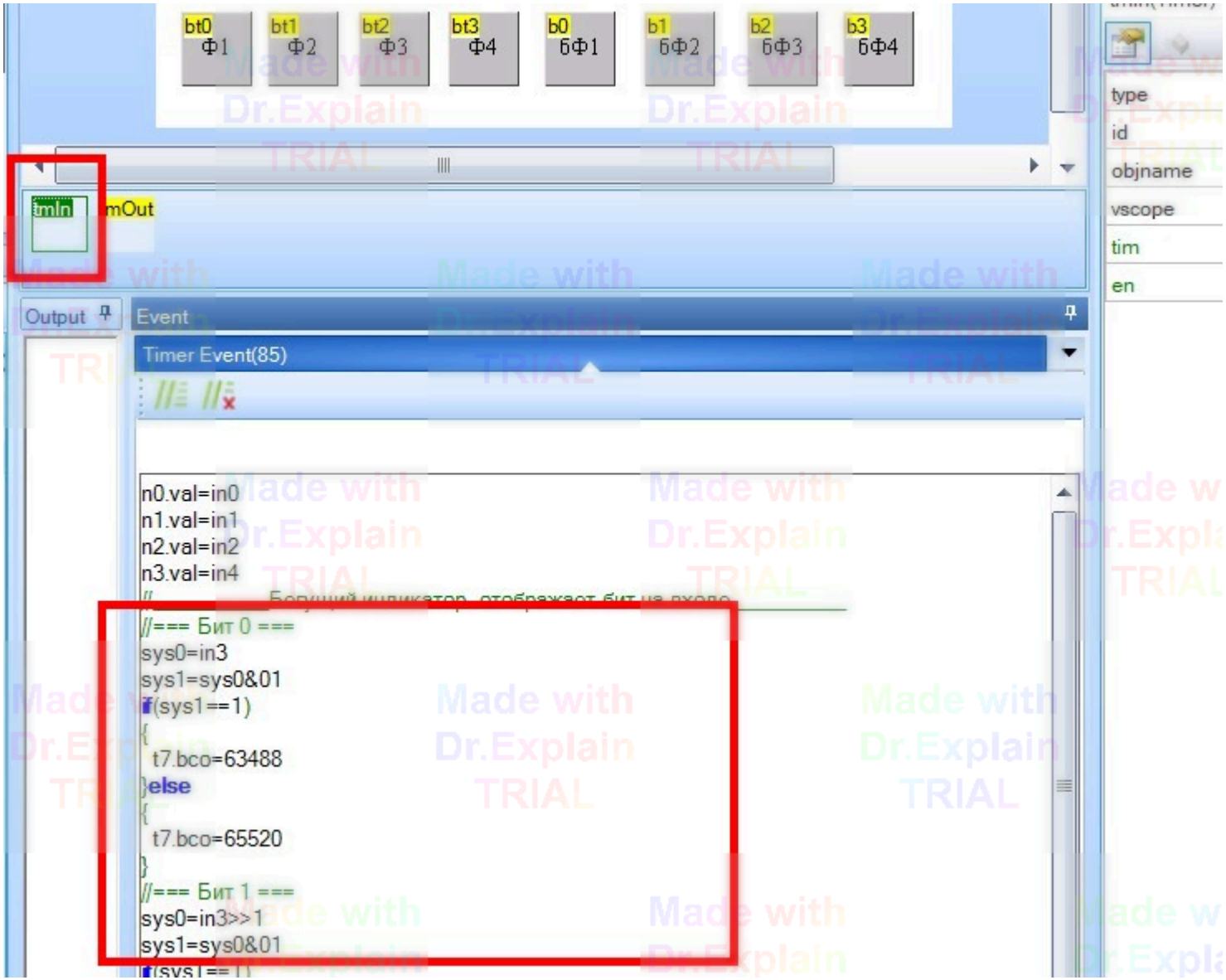
Используя блок **var->bit** или **Byte->Bits** из полученного из Nxt байта состояний кнопок получаем биты, которые далее можем использовать в программе.



5.2. Прием/передача в Nextion

Получение байта и его разложение на биты

Для разложения полученного из МК байта состояний в Nxt необходимо в поле **Timer Event** таймера **tmIn** прописать код:


bt0 Ф1
bt1 Ф2
bt2 Ф3
bt3 Ф4
b0 бФ1
b1 бФ2
b2 бФ3
b3 бФ4
type
id
objname
vscope
tim
en
Output
Event
Timer Event(85)
n0.val=in0
n1.val=in1
n2.val=in2
n3.val=in4
//=== Бит 0 ===
sys0=in3
sys1=sys0&01
if(sys1==1)
{
t7.bco=63488
}else
{
t7.bco=65520
}
//=== Бит 1 ===
sys0=in3>>1
sys1=sys0&01

```
//_________Разложение байта на биты_________
//=== Бит 0 ===
sys0=in3                  // берем переменную, в которой собраны дискретные состояния, по умолчанию берется нулевой бит
sys1=sys0&01          // операция логического умножения для определения состояния бита. Если бит =0, то результат =0, если бит=1, то результат =1
if(sys1==1)               // далее необходимое условие. В даном случае, если бит =1
{
  t7.bco=63488          // то назначем текстовому полю t7 цвет 63488. Здесь может быть любой элемент. Например, n0.val=1 или t7.txt="УРА!" И т.п.
}else
{
  t7.bco=65520          // иначе назначем цвет текстового поля 65520. или другой текст, или другое значение элемента "номер"
}
//=== Бит 1 ===
sys0=in3>>1          // делаем сдвиг в переменной in3 на 1 позицию.
sys1=sys0&01      // логическое умножение
if(sys1==1)
{
  t8.bco=63488
}else
{
  t8.bco=65520
}
//=== Бит 2 ===
sys0=in3>>2      // продолжаем сдвигать на 1 позицию, и так доя 7-го бита.
sys1=sys0&01
if(sys1==1)
{
  t9.bco=63488     //прописываем нужные нам условия
}else
```

```
{
  t9.bco=65520
}
//=== Бит 3 ===
sys0=in3>>3
sys1=sys0&01
if(sys1==1)
{
  t10.bco=63488
}else
{
  t10.bco=65520
}
//=== Бит 4 ===
sys0=in3>>4
sys1=sys0&01
if(sys1==1)
{
  t11.bco=63488
}else
{
  t11.bco=65520
}
//=== Бит 5 ===
sys0=in3>>5
sys1=sys0&01
if(sys1==1)
{
  t12.bco=63488
}else
{
  t12.bco=65520
}
```

```
//=== Бит 6 ===
sys0=in3>>6
sys1=sys0&01
if(sys1==1)
{
  t13.bco=63488
}else
{
  t13.bco=65520
}
//=== Бит 7 ===
sys0=in3>>7
sys1=sys0&01
if(sys1==1)
{
  t14.bco=63488
}else
{
  t14.bco=65520
}
```

Таким образом, с помощью такой конструкции можно получить битовые состояния.

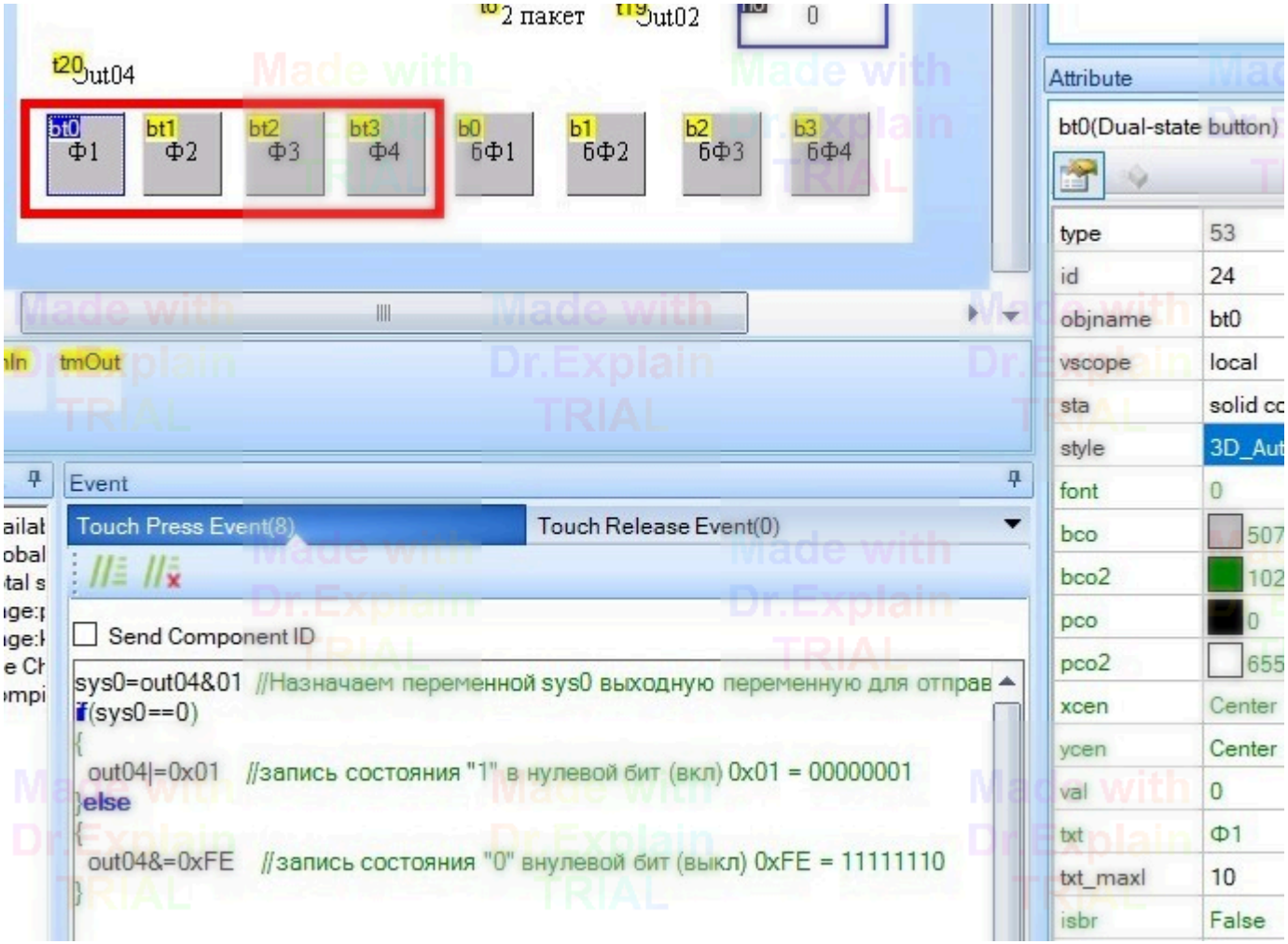
Сборка и отправка байта в Nxt

Сборку байта из битов состояния рассмотримс на примере переключателей.

В Nxt есть два типа параелючателей: с фиксацией (dual state button, Switch, Checkbox, Radio) и без фиксации (Button, либо надатие на любой элемент интерфейса).

В общем случае запись бита в определенный байт происходит одинаково.

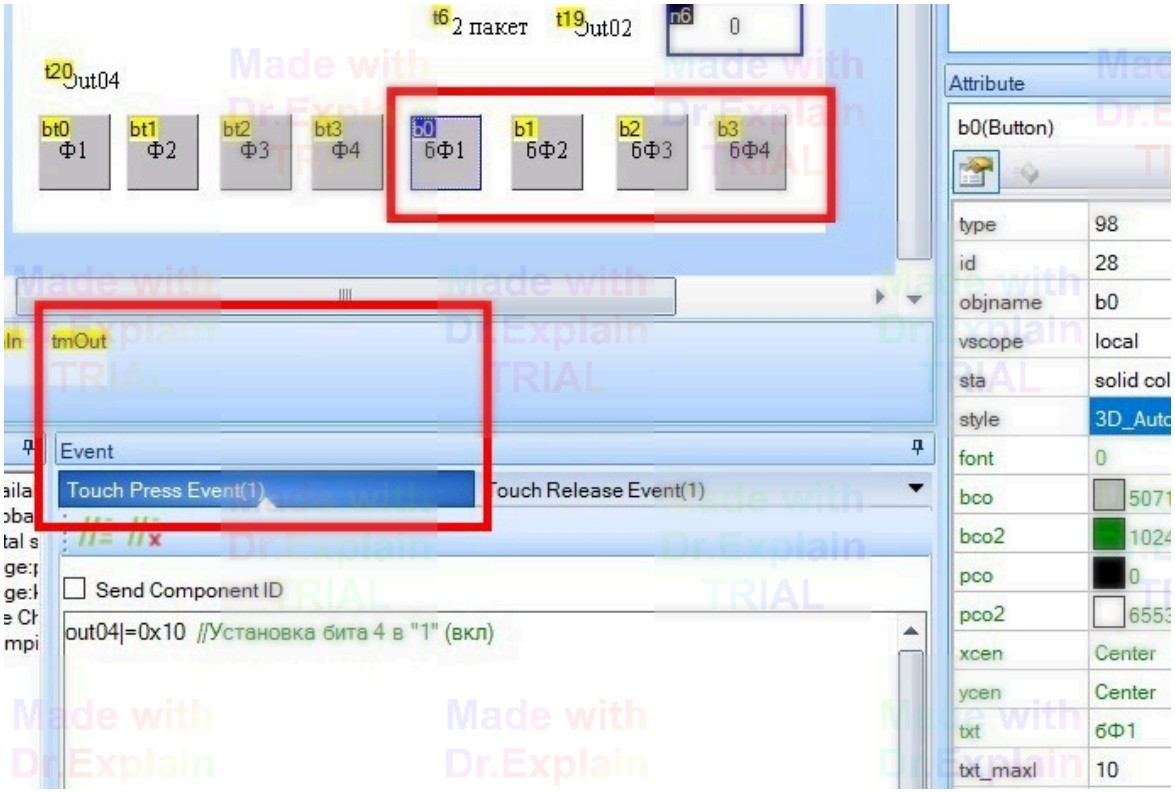
Передача состояния переключателя (кнопка с фиксацией)

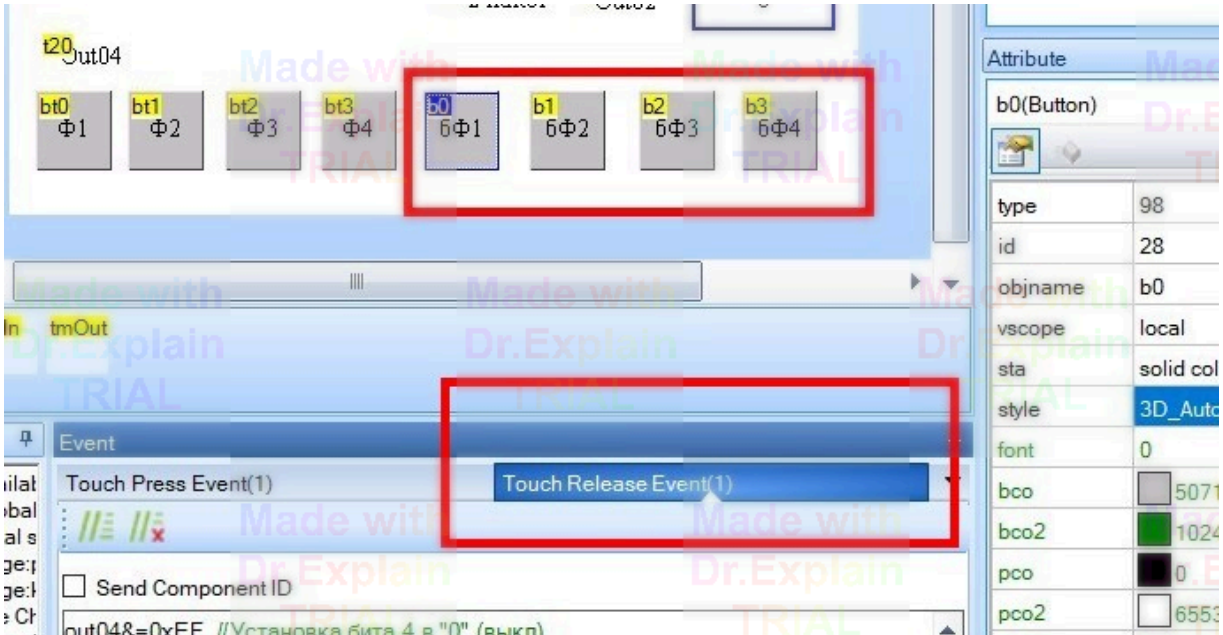


код в **Touch Press Event** на выбранном элементе.

```
sys0=out04&01     //Назначаем системной переменной sys0 выходную переменную out04 для отправки
if(sys0==0)
{
  out04|=0x01          //запись состояния в первый бит (вкл) 0x01 = 00000001
  OnOffSW.pic=58     //установка картинки состояния
t15.txt="Вкл"             // любое связанное событие интерфейса: изменение цвета, смена картинки т .п.
}else
{
  out04&=0xFE          //запись состояния в первый бит (выкл) 0xFE = 11111110
  OnOffSW.pic=57
t15.txt="Выкл"              // любое связанное событие интерфейса: изменение цвета, смена картинки т .п.
}
```

Передача состояния кнопки без фиксации


t6 2 пакет
t19 out02
n6 0
t20 out04
bt0 Ф1
bt1 Ф2
bt2 Ф3
bt3 Ф4
b0 бФ1
b1 бФ2
b2 бФ3
b3 бФ4
tmOut
Event
Touch Press Event(1)
Touch Release Event(1)
Send Component ID
out04|=0x10 //Установка бита 4 в "1" (вкл)
Attribute
b0(Button)
type 98
id 28
objname b0
vscope local
sta solid col
style 3D_Auto
font 0
bco 5071
bco2 1024
pco 0
pco2 6553
xcen Center
ycen Center
txt бФ1
txt_maxl 10


t20 out04
bt0 Ф1
bt1 Ф2
bt2 Ф3
bt3 Ф4
b0 бФ1
b1 бФ2
b2 бФ3
b3 бФ4
tmOut
Event
Touch Press Event(1)
Touch Release Event(1)
Send Component ID
Attribute
b0(Button)
type 98
id 28
objname b0
vscope local
sta solid col
style 3D_Auto
font 0
bco 5071
bco2 1024
pco 0
pco2 6553

код в **Touch Press Event** на выбранном элементе:

out04|=0x10 //запись состояния в 4 бит переменной **out04** (**вкл**) 0x10 = 00010000

код в **Touch Release Event**

out04&=0xEF //запись состояния в 4 бит (**выкл**) 0xEF = 11101111

Таблица адресов битов для ВКЛ и ВЫКЛ состояний:

номер бита	вкл HEX вкл 2	выкл HEX выкл 2
0	0x01 00000001	0xFE 11111110
1	0x02 00000010	0xFD 11111101
2	0x04 00000100	0xFB 11111011
3	0x08 00001000	0xF7 11110111
4	0x10 00010000	0xEF 11101111
5	0x20 00100000	0xDF 11011111
6	0x40 01000000	0xBF 10111111

7		0x80 10000000	0x7F01111111

Переменная (в данном примере **out04**) в таймере отправки **tmOut** прописывается точно так же, как и другие переменные (т.е. проверка изменения и т.д),

часть таймера отправки

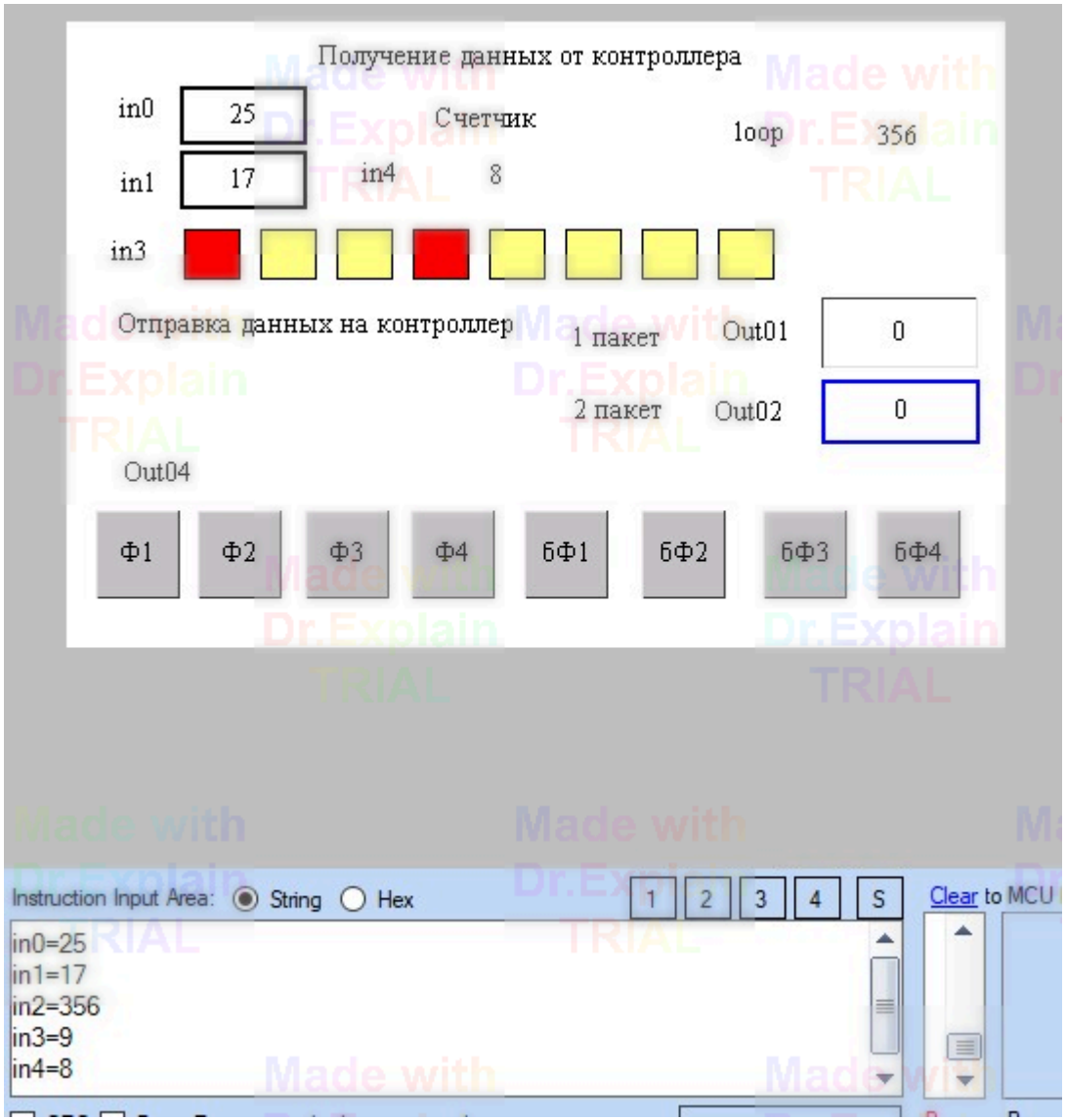
```
out01=n4.val
out02=n6.val
out03=7
// out04 - устанавливается по событиям (нажатие) кнопки (4 с фиксацией, 4 без фиксации) Должно быть закомментировано
//=============================================================================================
//      2.ПРОВЕРКА НА ИЗМЕНЕНИЕ ПЕРЕМЕННЫХ outxx
//=============================================================================================
if(ago00!=out00)
{
  cmdPack=2
  ago00=out00
}
//------
if(ago01!=out01)
{
  cmdPack=2
  ago01=out01
}
//------
if(ago02!=out02)
{
  cmdPack=2
  ago02=out02
}
```

```
//------
if(ago04!=out04)
{
  cmdPack=2
  ago04=out04
}
```

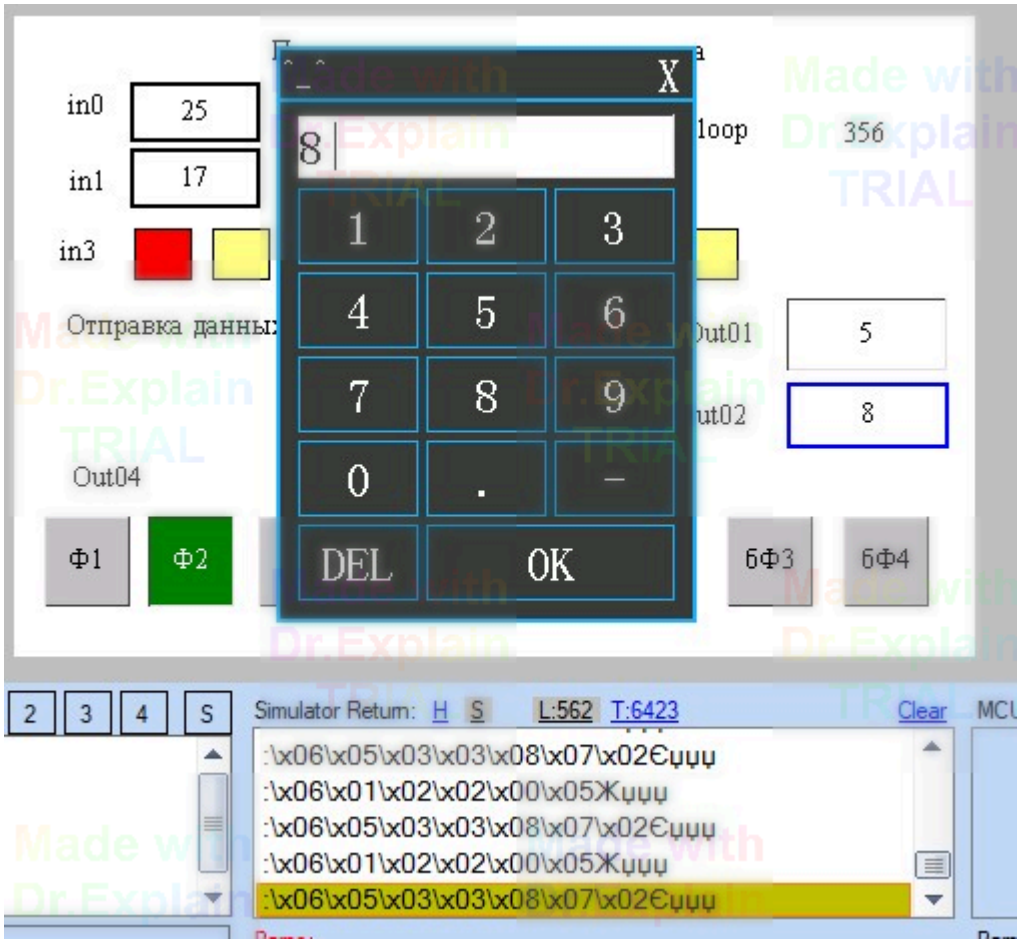
и так далее,

6. Отладка

Для проверки корректности отображения принимаемых данных из МК и отправляемых из Nxt проект можно запустить в Debug Nextion Editor.



в поле ввода команд можно напрямую прописывать входящие переменные.



Отправку переменных можно контролировать в поле вывода. Строка представляет собой сформированный пакет отправки в формате, описанно в разделе 1 Общие положения

Для систематизации и последующего упрощения отладки рекомендуется составить таблицу передаваемых переменных между МК и Nxt.

Пример таблицы:

7. Примеры кодов задач

Здесь представлены некоторые типовые примеры кодов задач.

```
//
//==========================================================================================
//                                КОД ЗАДАЧИ Main (100ms)
// Рекомендации: устанавливать задачу на каждой странице
//
//==========================================================================================
```

```
//                      1.Периодическое мерцания(анимация) индикатором
//=========================================================================================
cntBlink++
if(cntBlink>=4)  //--Событие 100мс(100мс*4)
{
  cntBlink=0
//----Изображение 0-----------------------------------------------
  if(numBlink==0)
  {
     blink.pic=2
  }
//----Изображение 1-----------------------------------------------
  if(numBlink==1)
  {
      blink.pic=3
  }
//----Изображение 2-----------------------------------------------
  if(numBlink==2)
  {
      blink.pic=2
  }
```

```
//----Изображение 3-------------------------------------------------
  if(numBlink==3)
  {
     blink.pic=3
  }
//----Изменение счетчика анимаций-----------------------------------
  numBlink++
  if(numBlink>3)
  {
     numBlink=0
  }
//-----------------------------------------------------------------------------------------------
}
```

```
//                          2.ПРОВЕРКА ИЗМЕНЕНИЯ СИСТЕМНЫХ ПЕРЕМЕННЫХ
// Примечание:
//  1.Функция только для Nextion с расширенными возможностями(K,P).
//  2.Если в MCU у
//=================================================================================================
//-----Очистка флага изменения системных переменных-----------------------------------------------
vReadySystem=0
//-----Проверка на изменение номера страницы------------------------------------------------------
if(agoPage!=dp)
{
  agoPage=dp
  vReadySystem=1
}
//-----Проверка на изменение времени--------------------------------------------------------------
sys0=rtc5          //--Контроль сек;
//sys0=rtc4          //--Контроль минут;
if(changeTime!=sys0)
{
```

```
  changeTime=sys0
  vReadySystem=1
}
```

```
//                          3.ОТПРАВКА ИЗ NEXTION СИСТЕМНЫХ ПАРАМЕТРОВ
// Рекомендации: включать в главную задачу Main (period=100ms)
// Примечание: функция для Nextion с расширенными возможностями(K,P).

//=========================================================================================
if(vReadySystem>0)  //--Проверка флага свежих данных
{
  //-----Выделение двух младших разрядов года-------------------------------------------------
  sys1=rtc0%100          //-
  head=0x3A              //--Заголовок
  func=0x56          //--Отправка нескольких регистров байтовых регистров
  adr=1              //--Адрес первого отправляемого регистра
  qntReg=10          //--Кол-во отправляемых регистров
  qntByte=10         //--Кол-во отправляемых байт
  //-----Расчет контрольной суммы-------------------------------------------------------------
  crc=head           //--Код заголовка (':') в КС;
  crc+=func          //--Параметр func;
  crc+=adr           //--Параметра adr;
  crc+=qntReg        //--Параметр qntReg;
  crc+=qntByte       //--Параметр qntByte;
  crc+=dp            //--Номер текущей страницы;
  crc+=sys1              //--Два младших разряда года;
  crc+=rtc1              //--Месяц;
  crc+=rtc2          //--Число;
  crc+=rtc3          //--Часы;
  crc+=rtc4          //--Минуты;
  crc+=rtc5          //--Секунды;
  crc+=rtc6          //--День недели;
```

```
  crc+=0              //--Резервный байт;
  crc+=0              //--Резервный байт;
  crc=crc&0xFF          //- Нормализация CRC до байта:
  //-----Вычисление  crc с двумя дополнениями----------------------------------------------------
  crc=0xFF-crc
  crc+=1
  crc&=0xFF
  //-----Отправка посылки в UART-----------------------------------------------------------------
  prints head,1       //--
  prints func,1       //--Команда записи нескольких регистров
  prints adr,1        //--Адрес записи
  prints qntReg,1   //--Кол-во записываемых регистров
  prints qntByte,1     //--Кол-во отправляемых байт
  prints dp,1         //--Номер страницы
  prints sys1,1       //--Два младших разряда даты
  prints rtc1,1       //--Отправка регистра v1 (0xFF)
  prints rtc2,1       //--Отправка регистра v2 (0xFF)
  prints rtc3,1       //--Отправка регистра v3 (0xFF)
  prints rtc4,1       //--Отправка регистра v4 (0xFF)
  prints rtc5,1       //--Отправка регистра v4 (0xFF)
  prints rtc6,1       //--Отправка регистра v4 (0xFF)
  prints 0,1          //--Отправка регистра v4 (0xFF)
  prints 0,1          //--Отправка регистра v4 (0xFF)
  prints crc,1      //--Отправка crc (0xFF)
  printh FF FF FF   //--Отправка кодов завершения посылки
}
```

```
//                    4.ФОРМИРОВАНИЕ СТРОКОВЫХ ПЕРЕМЕННЫХ ДАТЫ И ВРЕМЕНИ
// Необходимые параметры:
// vSTR - переменная на странице, определенная на странице компонетом Variable;
// txtWeek- поле, опредленное компонентом Text с параметрами: txt_maxl=15;
// txtData - поле, опредленное компонентом Text с параметрами: txt_maxl=10;
```

```
// txtTime - поле, опредленное компонентом Text с параметрами: txt_maxl=10.
// Примечание: функция для Nextion с расширенными возможностями(K,P).
//=========================================================================================
if(agoSec!=rtc5)
{
  agoSec=rtc5
  //-----Формирование даты и дня недели-------------------------------------------------------------
  cov rtc2,vSTR.txt,2
  txtData.txt=vSTR.txt  //--день
  txtData.txt+="."
  cov rtc1,vSTR.txt,2
  txtData.txt+=vSTR.txt //--месяц
  txtData.txt+="."
  cov rtc0,vSTR.txt,4
  txtData.txt+=vSTR.txt //--год
  //-----Формирование времении----------------------------------------------------------------------
  cov rtc3,vSTR.txt,2
  txtTime.txt=vSTR.txt  //--час
  txtTime.txt+=":"
  cov rtc4,vSTR.txt,2
  txtTime.txt+=vSTR.txt  //--минута
  txtTime.txt+=":"
  cov rtc5,vSTR.txt,2
  txtTime.txt+=vSTR.txt  //--секунда
  //-----Формирование дня недели-------- ------------------------------------------------------------
  sys0=rtc6
  txtWeek.txt="день"
  if(sys0==0)
  {
    txtWeek.txt="Воскресение"
  }
  if(sys0==1)
  {
```

```
  txtWeek.txt="Понедельник"
 }
 if(sys0==2)
 {
  txtWeek.txt="Вторник"
 }
 if(sys0==3)
 {
  txtWeek.txt="Среда"
 }
 if(sys0==4)
 {
  txtWeek.txt="Четверг"
 }
 if(sys0==5)
 {
  txtWeek.txt="Пятница"
 }
 if(sys0==6)
 {
  txtWeek.txt="Суббота"
 }
}
//================================================================================
```

```
//                    5.ИНДИКАЦИЯ БИТОВАЯ
// Необходимые параметры:
// ledMCU - поле созданное компонентом Picture с установленным свойством .pic соответствущим 0 (в примере=2).
//================================================================================
sys0=in0
sys2=2
sys1=sys0&01
ledMCU.pic=sys1+sys2
```

```
//                    6.ПОБИТОВАЯ ИНДИКАЦИЯ ПЕРЕМЕННОЙ ТИПА БЕГУЩАЯ "1"
// Необходимые параметры:
// run0-run7 - поле созданное компонентом Picture с установленным свойством .pic соответствущим 0 (в пример=4).
//=============================================================================================
sys0=in1     //--Сохранение входной переменной
sys2=4         //--Номер изображения для значения бита=0;
//-----------------------------------------
sys1=sys0&01
run0.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>1
sys1=sys1&01
run1.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>2
sys1=sys1&01
run2.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>3
sys1=sys1&01
run3.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>4
sys1=sys1&01
run4.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>5
sys1=sys1&01
run5.pic=sys1+sys2
//-------------------------------------------------------------------------------------------
sys1=sys0>>6
sys1=sys1&01
```

```
run6.pic=sys1+sys2
//-----------------------------------------------------------------------------------------------
sys1=sys0>>7
sys1=sys1&01
run7.pic=sys1+sys2
```

```
//                    7.ОБРАБОТКА КРУГОВОГО ИНДИКАТОРА
// Необходимые параметры:
// adcGauge - поле созданное компонентом Gauge   с установленным свойством .val=0
// adcNum   - поле созданное компонентом Numeric с установленным свойством .val=0
//=================================================================================================
sys0=in5
sys1=in5
//-----------------------------------------------------------------------------------------------
sys0*=270
sys0/=4095
sys0=sys0-45
if(sys0<0)
{
  sys0+=360
}
//-----------------------------------------------------------------------------------------------
adcGauge.val=sys0
adcNum.val=sys1
```

```
//                  8.  УПРАВЛЕНИЕ ДАННЫМИ МАССИВА ГРАФИКА
//  Рекомендации: включить функцию в каждую страницу для динамического отслежживания изменения данных.
//=================================================================================================
//               Команды управления байтами через grB, где
// байт 0 - данные(dataByte);
// байт 1 - команда управления(cmdByte);
//  cmdByte=0    - нет действий
//  cmdByte=0x70 - заполнение массива байтом;
```

```
//  cmdByte=0x71 - запись байта со сдвигом всего массива;
// (cmdByte>0 & cmdByte<=64) - запись байта в массив по индексу=cmnd-1;
// По завершению обработки слово grB очищается.
//=============================================================================
//                Команды управления словами через xxL и xxR
// grL - данные(слово);
// grR - команда управлени(cmnd);
//  cmdLong=0    - нет действий
//  cmdLong=0x70 - заполнение массива словом;
//  cmdLong=0x71 - запись слова со сдвигом всего массива;
// (cmdLong>0 & cmdLong<=16) - запись слова в массив по индексу=cmnd-1;
// По завершению команда grR очищается.
//=============================================================================

//=============================================================================
//                1.Сохранение команд управления и данных
//=============================================================================
dataByte=grB
dataLong=grL
dataRun=grR
//=============================================================================
//                2.Выделение кодов управления (cmdByte,cmdLong)
//=============================================================================
cmdByte=dataByte>>8     //--Выделения байта 1 команды yKод управления
cmdByte&=0xFF           //--Код управления для команд управления байтами;
cmdLong=dataRun           //--Код управления для команд управления словами;
//=============================================================================
//                3.Заполнение массива байтом
//=============================================================================
if(cmdByte==0X70)
{
//-----------------------------------------------------------------------------------------
  sys0=dataByte&0xFF
```

```
  sys1=sys0
  sys1<<=8
  sys1|=sys0
  sys1<<=8
  sys1|=sys0
  sys1<<=8
  sys1|=sys0
//-------------------------------------------------------------------------------------------
  gr0=sys1
  gr1=sys1
  gr2=sys1
  gr3=sys1
  gr4=sys1
  gr5=sys1
  gr6=sys1
  gr7=sys1
  gr8=sys1
  gr9=sys1
  gr10=sys1
  gr11=sys1
  gr12=sys1
  gr13=sys1
  gr14=sys1
  gr15=sys1
  grEnd=sys1 //--Заполнение вспомогательного слова;
} //================END if(cmndGrafic==1)=============================================

//=============================================================================================
//                  4.Запись байта со сдвигом всего массива;
//=============================================================================================
if(cmdByte==0x71)
{
//-------------------------------------------------------------------------------------------
```

```
sys0=gr15>>24
sys0=sys0&0xFF
grEnd<<=8
grEnd|=sys0
//---15--------------------------------
sys0=gr14>>24
sys0=sys0&0xFF
gr15<<=8
gr15|=sys0
//---14--------------------------------
sys0=gr13>>24
sys0=sys0&0xFF
gr14<<=8
gr14|=sys0
//---13--------------------------------
sys0=gr12>>24
sys0=sys0&0xFF
gr13<<=8
gr13|=sys0
//---12--------------------------------
sys0=gr11>>24
sys0=sys0&0xFF
gr12<<=8
gr12|=sys0
//---11--------------------------------
sys0=gr10>>24
sys0=sys0&0xFF
gr11<<=8
gr11|=sys0
//---10--------------------------------
sys0=gr9>>24
sys0=sys0&0xFF
gr10<<=8
```

```
gr10|=sys0
//---9--------------------------------
sys0=gr8>>24
sys0=sys0&0xFF
gr9<<=8
gr9|=sys0
//---8--------------------------------
sys0=gr7>>24
sys0=sys0&0xFF
gr8<<=8
gr8|=sys0
//---7--------------------------------
sys0=gr6>>24
sys0=sys0&0xFF
gr7<<=8
gr7|=sys0
//---6--------------------------------
sys0=gr5>>24
sys0=sys0&0xFF
gr6<<=8
gr6|=sys0
//---5--------------------------------
sys0=gr4>>24
sys0=sys0&0xFF
gr5<<=8
gr5|=sys0
//---4--------------------------------
sys0=gr3>>24
sys0=sys0&0xFF
gr4<<=8
gr4|=sys0
//---3--------------------------------
sys0=gr2>>24
```

```
  sys0=sys0&0xFF
  gr3<<=8
  gr3|=sys0
  //---2------------------------------
  sys0=gr1>>24
  sys0=sys0&0xFF
  gr2<<=8
  gr2|=sys0
  //---1------------------------------
  sys0=gr0>>24
  sys0=sys0&0xFF
  gr1<<=8
  gr1|=sys0
  //---0------------------------------
  sys0=dataByte&0xFF
  gr0<<=8
  gr0|=sys0
} //================END if(cmndGrafic==2)=============================================

//=============================================================================================
//                    5.Запись нулевого байта grData по номеру байта массива
//=============================================================================================
if(cmdByte>0)
{
 if(cmdByte<=64)
 {
//-----------------------------------------------------------------------------------------
  sys0=dataByte
  sys1=sys0>>8       //--Сдвиг для выделения номера байта
  sys1&=0xFF         //--Номер байта
  sys1-=1            //          в массиве;
  sys0&=0xFF         //--Значение байта;
//-----------------------------------------------------------------------------------------
```

```
  sys2=sys1/4            //--Номер слова;
  sys2*=4
  sys2=sys1-sys2      //--Номер байта в слове;
  sys2*=8               //--Кол-во сдвигов в слове до требуемого байт
  sys0<<=sys2           //--Маска с данными для записи требуемого байт
//-------------------------------------------------------------------------------------------------
  //--0---------------------
  if(sys1==0)
  {
    gr0|=sys0
  }
  //--1---------------------
  if(sys1==1)
  {
    gr1|=sys0
  }
  //--2---------------------
  if(sys1==2)
  {
    gr2|=sys0
  }
  //--3---------------------
  if(sys1==3)
  {
    gr3|=sys0
  }
  //--4---------------------
  if(sys1==4)
  {
    gr4|=sys0
  }
  //--5---------------------
  if(sys1==5)
```

```
{
  gr5|=sys0
}
//--6---------------------
if(sys1==6)
{
  gr6|=sys0
}
//--7---------------------
if(sys1==7)
{
  gr7|=sys0
}
//--8---------------------
if(sys1==8)
{
  gr8|=sys0
}
//--9---------------------
if(sys1==9)
{
  gr9|=sys0
}
//--10---------------------
if(sys1==10)
{
  gr10|=sys0
}
//--11---------------------
if(sys1==11)
{
  gr11|=sys0
}
```

```
  //--12---------------------
  if(sys1==12)
  {
    gr12|=sys0
  }
  //--13---------------------
  if(sys1==13)
  {
    gr13|=sys0
  }
  //--14---------------------
  if(sys1==14)
  {
    gr14|=sys0
  }
  //--15---------------------
  if(sys1==15)
  {
    gr15|=sys0
  }
//-------------------------------------------------------------------------------------------
 }
} //================END if(cmndGrafic==3)=============================================

//=========================================================================================
//                    6.Запись слова во все слова массива
//=========================================================================================
if(cmdLong==0x70)
{
  gr0=dataLong
  gr1=dataLong
  gr2=dataLong
  gr3=dataLong
```

```
  gr4=dataLong
  gr5=dataLong
  gr6=dataLong
  gr7=dataLong
  gr8=dataLong
  gr9=dataLong
  gr10=dataLong
  gr11=dataLong
  gr12=dataLong
  gr13=dataLong
  gr14=dataLong
  gr15=dataLong
  grEnd=dataLong //--Заполнение вспомогательного слова;
} //=================END if(cmndGrafic==4)=============================================

//=========================================================================================
//                    7.Запись слова со сдвигом всего массива
//=========================================================================================
if(cmdLong==0x71)
{
  grEnd=gr15
  gr15=gr14
  gr14=gr13
  gr13=gr12
  gr12=gr11
  gr11=gr10
  gr10=gr9
  gr9=gr8
  gr8=gr7
  gr7=gr6
  gr6=gr5
  gr5=gr4
  gr4=gr3
```

```
  gr3=gr2
  gr2=gr1
  gr1=gr0
  gr0=dataLong
} //=================END if(cmndGrafic==5)=============================================

//=============================================================================================
//                    8.Запись слова по номеру слова массива
//=============================================================================================
if(cmdLong>0)
{
 if(cmdLong<=16)
 {
//---------------------------------------------------------------------------------------------
  sys0=dataLong      //
  sys1=cmdLong-1 //--Определение номера слова
  //--0---------------------
  if(sys1==0)
  {
    gr0=sys0
  }
  //--1---------------------
  if(sys1==1)
  {
    gr1=sys0
  }
  //--2---------------------
  if(sys1==2)
  {
    gr2=sys0
  }
  //--3---------------------
  if(sys1==3)
```

```
{
  gr3=sys0
}
//--4---------------------
if(sys1==4)
{
  gr4=sys0
}
//--5---------------------
if(sys1==5)
{
  gr5=sys0
}
//--6---------------------
if(sys1==6)
{
  gr6=sys0
}
//--7---------------------
if(sys1==7)
{
  gr7=sys0
}
//--8---------------------
if(sys1==8)
{
  gr8=sys0
}
//--9---------------------
if(sys1==9)
{
  gr9=sys0
}
```

```
  //--10---------------------
  if(sys1==10)
  {
    gr10=sys0
  }
  //--11---------------------
  if(sys1==11)
  {
    gr11=sys0
  }
  //--12---------------------
  if(sys1==12)
  {
    gr12=sys0
  }
  //--13---------------------
  if(sys1==13)
  {
    gr13=sys0
  }
  //--14---------------------
  if(sys1==14)
  {
    gr14=sys0
  }
  //--15---------------------
  if(sys1==15)
  {
    gr15=sys0
  }
 }
} //================END if(cmndGrafic==6)=============================================
```

```
//=============================================================================
//                  10.Заполнение графика данными пила "Треугольник"
//=============================================================================
if(cmdLong==0x90)
{
 gr0=0x18100800
 gr1=0x38302820
 gr2=0x58504840
 gr3=0x78706860
 gr4=0x98908880
 gr5=0xB8B0A8A0
 gr6=0xD8D0C8C0
 gr7=0xF8F0E8E0
 gr8=0xE7EFF7FF
 gr9=0xC7CFD7DF
gr10=0xA7AFB7BF
gr11=0x878F979F
gr12=0x676F777F
gr13=0x474F575F
gr14=0x272F373F
gr15=0x070F171F
grEnd=0
}
//=============================================================================
//                   9.Очистка управляющих параметров
//=============================================================================
grB=0
grR=0
//=============================================================================
```

```
//                  9.ФУНКЦИЯ ВЫВОДА ГРАФИКА
//  Входные параметры:
//  gr0-gr15,grEnd
```

```
//  Рабочие переменнные
//  valGrafic,limGrafic,idx,idx1
//  В функции в командах cle и add приходится вручную
//  корректировать ID графика(в примере ID=9) и chanel графика (в примере chanel=0)
//=============================================================================

//=======================Очистка графика=======================================
cle 9,0
//======Обработка 16-ти слова в цикле=========================================
for(idx=0;idx<16;idx++)
{
//======Обработка 4 х-байт(каждого из 16-ти слова) в цикле====================
  for(idx1=0;idx1<4;idx1++)
  {
//======Выбор текущего и следующего слова=====================================
    if(idx==0)
    {
      valGrafic=gr0
      limGrafic=gr1
    }
    if(idx==1)
    {
      valGrafic=gr1
      limGrafic=gr2
    }
    if(idx==2)
    {
      valGrafic=gr2
      limGrafic=gr3
    }
    if(idx==3)
    {
      valGrafic=gr3
```

```
  limGrafic=gr4
}
if(idx==4)
{
  valGrafic=gr4
  limGrafic=gr5
}
if(idx==5)
{
  valGrafic=gr5
  limGrafic=gr6
}
if(idx==6)
{
  valGrafic=gr6
  limGrafic=gr7
}
if(idx==7)
{
  valGrafic=gr7
  limGrafic=gr8
}
if(idx==8)
{
  valGrafic=gr8
  limGrafic=gr9
}
if(idx==9)
{
  valGrafic=gr9
  limGrafic=gr10
}
if(idx==10)
```

```
    {
      valGrafic=gr10
      limGrafic=gr11
    }
    if(idx==11)
    {
      valGrafic=gr11
      limGrafic=gr12
    }
    if(idx==12)
    {
      valGrafic=gr12
      limGrafic=gr13
    }
    if(idx==13)
    {
      valGrafic=gr13
      limGrafic=gr14
    }
    if(idx==14)
    {
      valGrafic=gr14
      limGrafic=gr15
    }
    if(idx==15)
    {
      valGrafic=gr15
      limGrafic=grEnd
    }
//======Подготовка данных для вывода текущего байта================================================
//------Подготовка данных для вывода 0-го байта-------------------------------------------------
    if(idx1==0)
    {
```

```
      sys0=valGrafic
      sys0&=0xFF
      sys1=valGrafic>>8
      sys1&=0xFF
    }
//------Подготовка данных для вывода 1-го байта-----------------------------------------------------
    if(idx1==1)
    {
      sys0=valGrafic>>8
      sys0&=0xFF
      sys1=valGrafic>>16
      sys1&=0xFF
    }
//------Подготовка данных для вывода 2-го байта-----------------------------------------------------
    if(idx1==2)
    {
      sys0=valGrafic>>16
      sys0&=0xFF
      sys1=valGrafic>>24
      sys1&=0xFF
    }
//------Подготовка данных для вывода 3-го байта-----------------------------------------------------
    if(idx1==3)
    {
      sys0=valGrafic>>24
      sys0&=0xFF
      sys1=limGrafic
      sys1&=0xFF
    }
//======Вывода 8 точек между двумя смежными байта=================================================
    add 9,0,sys0
//-------------------------------------------------------------------------------------------------
    sys2=sys1-sys0
```

```
    sys2=sys2/7
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    sys2=sys1-sys0
    sys2=sys2/6
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    sys2=sys1-sys0
    sys2=sys2/5
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    sys2=sys1-sys0
    sys2=sys2/4
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    sys2=sys1-sys0
    sys2=sys2/3
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    sys2=sys1-sys0
    sys2=sys2/2
    sys0+=sys2
    add 9,0,sys0
//-------------------------------------------------------------------------------------------
    add 9,0,sys1        //--Вывод конечной точки;
 }//======END for(idx1=0;idx1<4;idx1++)
} //======END for(idx=0;idx<16;idx++)=============================================================
//=============================================================================================
```

8. Рабочий пример

Пример рабочего проекта, который рассматривался в данном руководстве, выложен на форуме FLProg https://forum.flprog.ru/viewtopic.php?t=6943&start=160#p119637

автор WildCat