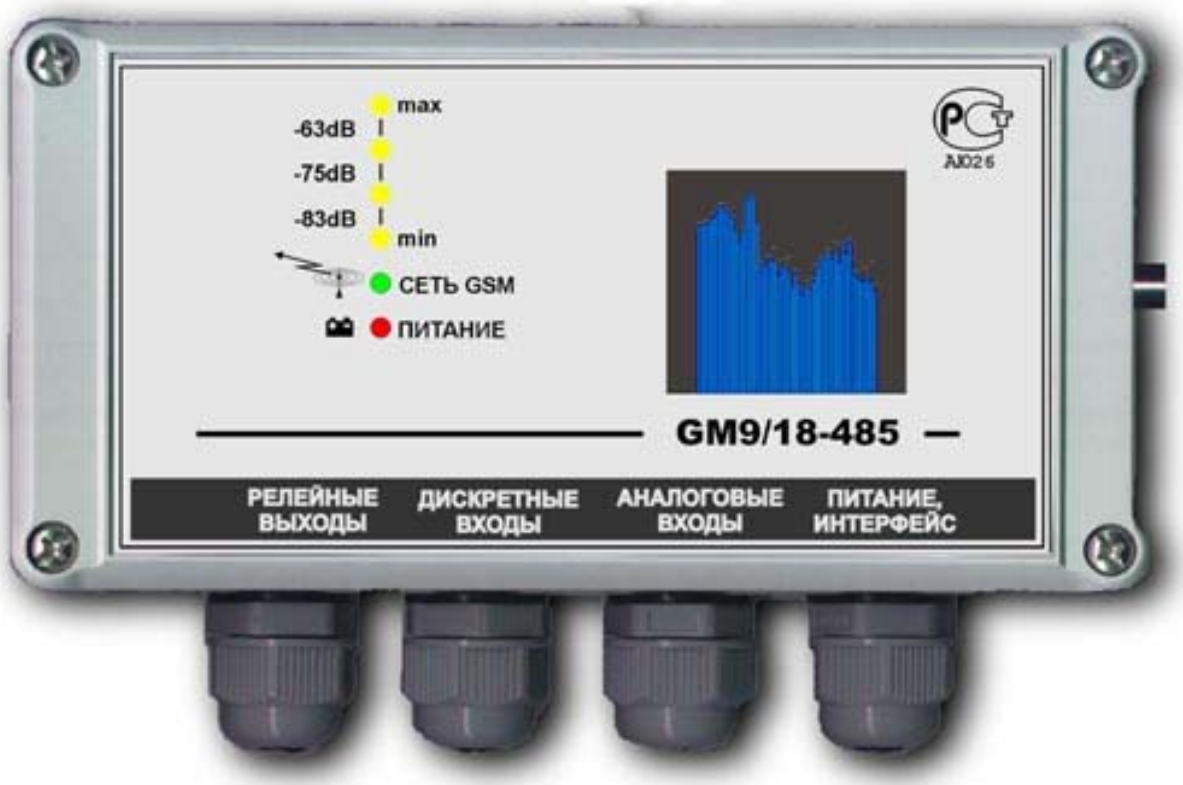
ЗАВОД МОЛНИЯ

МОДУЛЬ ПЕРЕДАЧИ ДАННЫХ GM9/18-485/232

Руководство по программированию
ВЕРСИЯ V.6



644031, г.Омск, ул. Звездова, 128
(3812) 36-74-38, 30-94-30, 51-99-66, 51-86-82
www.molnya.ru mail@molnya.ru

СОДЕРЖАНИЕ

1. Введение

Настоящий документ предназначен для изучения устройства и работы модуля передачи данных GM9/18-485/232 (в дальнейшем **модуль GM**) с точки зрения программиста, осуществляющего самостоятельное программирование модуля GM под конкретную телеметрическую задачу.

Специалистами создана **библиотека базового программного обеспечения (БПО)** (файл **bsw.c**), которая содержит основные функции для работы с портами, таймером, последовательной памятью и другими устройствами модуля GM. Использование БПО позволяет пользователю, знакомому с программированием на языке СИ, быстро начать практическую работу с модулем для решения своих конкретных задач.

Для более углубленного изучения работы модуля GM программисту следует изучить следующие документы:

- **msp430_UG.pdf, MSP430F14_.pdf** (описание процессора MSP430F149 фирмы “Texas Instruments”);
- описание используемого GSM модема;
- **24LC256.pdf** (описание последовательной флэш-памяти);
- **PCF8583.pdf** (описание таймера RTC).

В качестве **демонстрационного** примера передачи информации по каналу данных GSM приводится программа **main_gm.c,** которая позволяет осуществлять связь модуля GM с диспетчерским компьютером, передавать данные с дискретных, и аналоговых входов модуля. При этом на диспетчерском компьютере устанавливается программа **GMTest.exe**.

В модуле GM реализована концепция “программирования в системе”. При этом все программирование и внутрисхемная отладка происходит **по исходному тексту СИ программы,** и осуществляется в режиме реального времени через встроенный в модуль JTAG интерфейс, который подключается через преобразователь к параллельному (LPT) порту PC компьютера.

Для написания и отладки программ на PC компьютере используется рабочая среда “IAR Embedded Workbench” со встроенным компилятором языка СИ, ассемблер процессора MSP430F149, линкер и внутрисхемный эмулятор через JTAG интерфейс (ограничение 30 дней).

Таким образом, имеется **полный комплекс программно-аппаратных средств** для разработки пользователем своего собственного программного обеспечения для модуля GM.

2. Общее описание модуля GM.

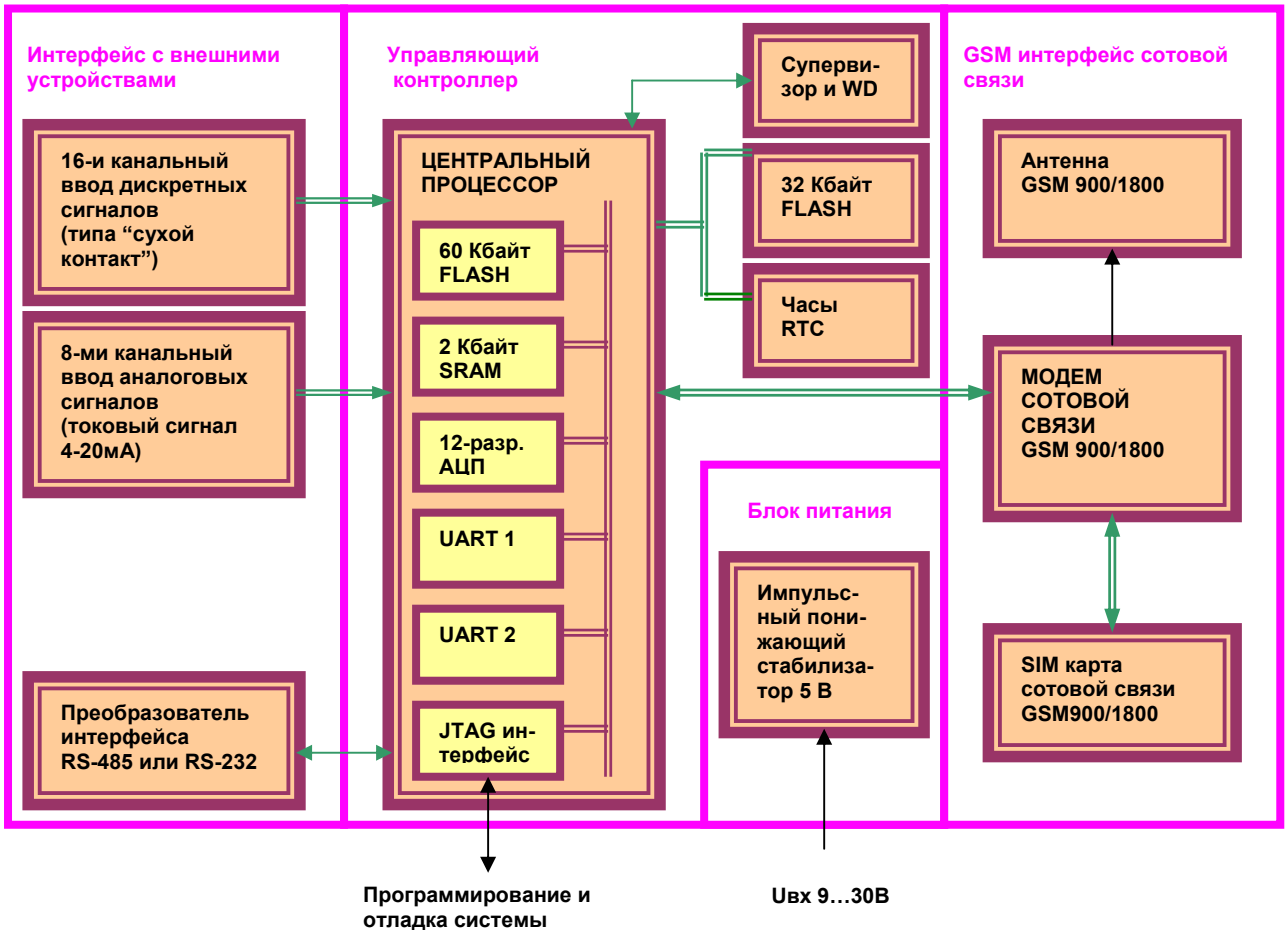
Модуль GM предназначен для передачи данных и SMS сообщений в EGSM900/GSM1800. Модуль содержит: GSM модем WISMO-QUIK, управляющий контроллер на базе процессора MSP430F149 фирмы “Texas Instruments”, блок ввода аналоговых сигналов (8 каналов), блок ввода дискретных сигналов (16 каналов), интерфейс RS-485 или RS-232 (только линии RxD и TxD), импульсный блок питания (DC-DC), внешний супервизор со сторожевым таймером (WD).

Модем WISMO-QUIK позволяет передавать информацию по каналу данных (скорость 9600 бит/сек), по каналу GPRS и по каналу SMS сообщений (текстовый и в PDU режим) в диапазонах 900 или 1800 МГц.

Управляющий контроллер опрашивает входные дискретные и аналоговые контакты модуля GM, управляет его выходными дискретными релейными выходами, осуществляет обмен по последовательным интерфейсам и т.д. Таким образом, применение управляющего контроллера позволяет использовать модуль GM в различных системах автоматики как управляющее или терминальное устройство.

Программирование и отладка производится через JTAG интерфейс непосредственно в системе.

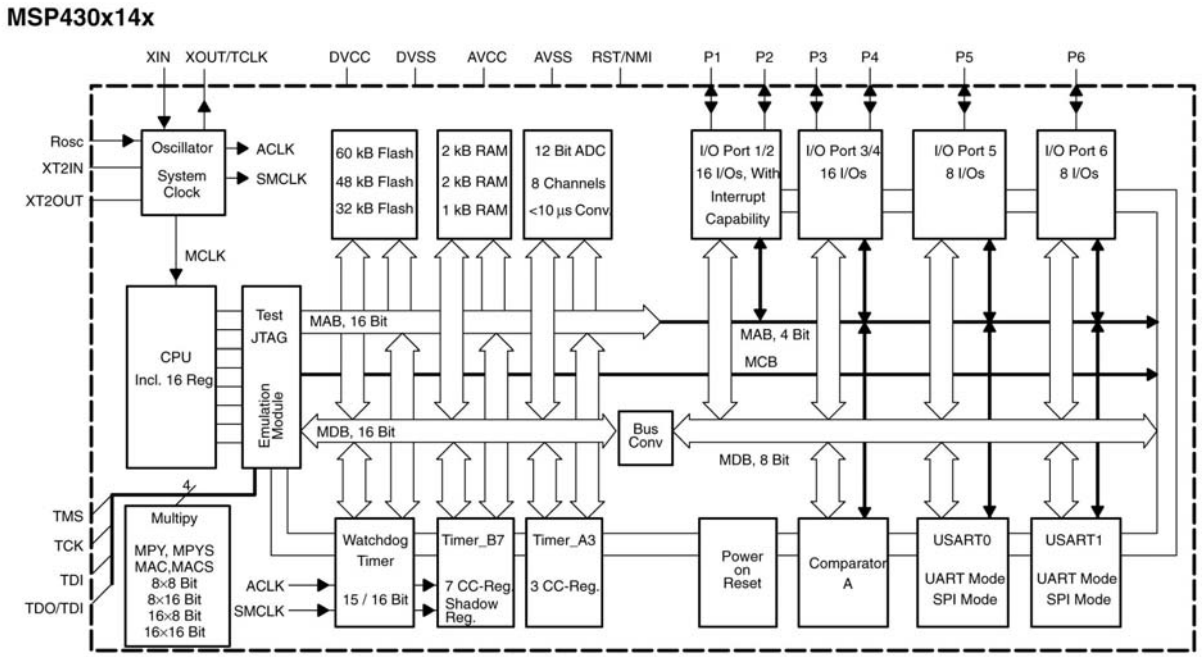
Структурная схема модуля GM.



Управляющий контроллер построен на базе процессора MSP430F149 фирмы “Texas Instruments”. Полное описание процессора содержится в файлах **msp430_UG.pdf, MSP430F14_.pdf** или на сайте фирмы “Texas Instruments” www.ti.com .

Процессор MSP430F149 (далее процессор) имеет 16 разрядную RISC архитектуру и ряд периферийных устройств на своем кристалле.

Ниже показана структура процессора MSP430F149.



Процессор содержит:

- **60 Кбайт** программной FLASH памяти, **2 Кбайта** ОЗУ, **256 байт** FLASH памяти данных;
- два последовательных асинхронных порта;
- 8-ми канальное 12-и разрядное АЦП;
- шесть 8-разрядных портов ввода-вывода;
- два 16-и разрядных таймера с регистрами сравнения;
- модули компаратора, сторожевого таймера, аппаратного умножения, контроля питания и т.д.

Программирование и отладка процессора осуществляется “в системе” через JTAG интерфейс. При этом процессор подключается через преобразователь JTAG интерфейса к параллельному (LPT) порту PC компьютера (см п. 4).

Процессор имеет единое адресное пространство 0…0FFFFh в котором расположены области памяти и регистров показанные ниже.

0FFFFh…0FFE0h	*Область векторов прерывания*
0FFFFh…01100h	*Кодовая память (60 Кбайт)*
010FFh…01000h	*Память данных FLASH (256 байт)*
0FFFh…0C00h	*Загрузчик памяти (зашивается при производстве процессора)*
09FFh…0200h	*Память данных ОЗУ (2 Кбайта)*
01FFh…0100h	*16-и битовые регистры периферийных модулей*
0FFh…010h	*8-и битовые регистры периферийных модулей*
0Fh…00h	*Специальные функциональные регистры (SFR)*

interrupt vector addresses

The interrupt vectors and the power-up starting address are located in the address range 0FFFFh – 0FFE0h. The vector contains the 16-bit address of the appropriate interrupt-handler instruction sequence.

INTERRUPT SOURCE	INTERRUPT FLAG	SYSTEM INTERRUPT	WORD ADDRESS	PRIORITY
Power-up External Reset Watchdog Flash memory	WDTIFG KEYV (see Note 1)	Reset	0FFFEh	15, highest
NMI Oscillator Fault Flash memory access violation	NMIIFG (see Notes 1 & 4) OFIFG (see Notes 1 & 4) ACCVIFG (see Notes 1 & 4)	(Non)maskable (Non)maskable (Non)maskable	0FFFCh	14
Timer_B7 (see Note 5)	BCCIFG0 (see Note 2)	Maskable	0FFFAh	13
Timer_B7 (see Note 5)	BCCIFG1 to BCCIFG6 TBIFG (see Notes 1 & 2)	Maskable	0FFF8h	12
Comparator_A	CAIFG	Maskable	0FFF6h	11
Watchdog timer	WDTIFG	Maskable	0FFF4h	10
USART0 receive	URXIFG0	Maskable	0FFF2h	9
USART0 transmit	UTXIFG0	Maskable	0FFF0h	8
ADC	ADCIFG (see Notes 1 & 2)	Maskable	0FFEEh	7
Timer_A3	CCIFG0 (see Note 2)	Maskable	0FFECh	6
Timer_A3	CCIFG1, CCIFG2, TAIFG (see Notes 1 & 2)	Maskable	0FFEAh	5
I/O port P1 (eight flags)	P1IFG.0 (see Notes 1 & 2) To P1IFG.7 (see Notes 1 & 2)	Maskable	0FFE8h	4
USART1 receive	URXIFG1	Maskable	0FFE6h	3
USART1 transmit	UTXIFG1		0FFE4h	2
I/O port P2 (eight flags)	P2IFG.0 (see Notes 1 & 2) To P2IFG.7 (see Notes 1 & 2)	Maskable	0FFE2h	1
			0FFE0h	0, lowest

NOTES:
1. Multiple source flags
2. Interrupt flags are located in the module.
3. Nonmaskable: neither the individual nor the general interrupt-enable bit will disable an interrupt event.
4. (Non)maskable: the individual interrupt-enable bit can disable an interrupt event, but the general-interrupt enable can not disable it.
5. Timer_B7 in MSP430x14x family has 7 CCRs; Timer_B3 in MSP430x13x family has 3 CCRs; in Timer_B3 there are only interrupt flags CCIFG0, 1, and 2, and the interrupt-enable bits CCIE0, 1, and 2 integrated.

Все SFR регистры, регистры периферийных модулей и вектора прерываний процессора определены в файле **msp430x14x.h** .

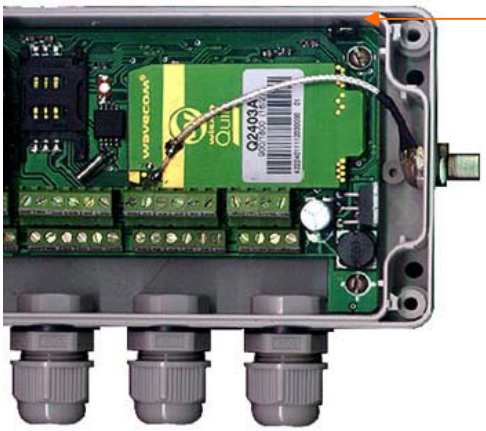
Для программирования модуля GM имеется **библиотека базового программного обеспечения (БПО)** (файл **bsw.c**), которая содержит функции для работы с основными устройствами модуля. Программист может воспользоваться данными функциями, или написать свои собственные функции в зависимости от условий конкретной задачи.

Внешний супервизор служит для повышения надежности работы модуля в автономном режиме. Супервизор содержит сторожевой таймер (WD) и схему контроля напряжения питания. Сигнал RESET с супервизора на процессор поступает через перемычку J1, которая находится в правой верхней части платы модуля GM (вблизи винта крепления).

Сигнал RESET вырабатывается супервизором когда напряжение питания процессора становится меньше 2,64В или когда период импульсов сброса сторожевого таймера становится больше 1,2 сек.

Работу с супервизором рекомендуется производить соответствующими функциями базового программного обеспечения (БПО) (см. **“Руководство программиста”**)

ВНИМАНИЕ! В режиме отладки или программирования через JTAG интерфейс необходимо ОТКЛЮЧИТЬ перемычку J1.



3. Описание работы модуля GM9/18-485/232

3.1 Работа с внешним супервизором.

Внешний супервизор вырабатывает сигнал RESET, когда напряжение питания процессора становится меньше 2,64В или когда период импульсов сброса сторожевого таймера становится больше 1,2 сек. Сторожевой таймер сбрасывается через порт P2.6 процессора MSP430F149.

ВНИМАНИЕ! В режиме отладки, программирования через JTAG интерфейс, или если программа не использует внешний супервизор, то необходимо ОТКЛЮЧИТЬ перемычку J1.

Функции работы с внешним супервизором	
`void IniExternalWD(void);`	*Инициализация внешнего супервизора*
`void ExternalWD(void);`	*Сброс сторожевого таймера*

3.2 Ввод дискретных сигналов D0..D7 модуля.

Контакты D0…D7 модуля подключены через оптроны к порту P1 процессора MSP430F149 в соответствии показанной с таблицей.

D0	D1	D2	D3	D4	D5	D6	D7
P1.2	P1.4	P1.0	P1.1	P1.5	P1.6	P1.3	P1.7

При **замыкании** D0…D7 на DG (цифровой “общий”) на соответствующих разрядах порта P1 устанавливается **“0”**, при размыкании – **“1”**.

Функции ввода дискретных сигналов D0…D7	
`void IniDIn (void);`	*Инициализация ввода*
`byte DIn(void);`	*Ввод сигналов D0...D7* Функция возвращает значение D0…D7 в формате: **D7,D6,D5,D4,D3,D2,D1,D0**(млад.бит). 0 - соответствует замкнутому состоянию входа; 1- соответствует разомкнутому состоянию входа.

3.3 Ввод дискретных сигналов D8..D15 модуля.

Контакты D8…D15 модуля подключены через оптроны к порту P4 процессора MSP430F149 в соответствии показанной с таблицей.

D8	D9	D10	D11	D12	D13	D14	D15
P4.7	P4.0	P4.6	P4.1	P4.3	P4.4	P4.5	P4.2

При **замыкании** D8…D15 на DG (цифровой "общий") на соответствующих разрядах порта P4 устанавливается **"0"**, при размыкании – **"1"**.

Функции ввода дискретных сигналов D8…D15	
`void IniDIn16 (void);`	*Инициализация ввода D8...D15*
`byte Din16(void);`	*Ввод сигналов D8...D15* Функция возвращает значение D8…D15 в формате: **D15,D14,D13,D12,D11,D10,D9,D8**(млад.бит). 0 - соответствует замкнутому состоянию входа; 1- соответствует разомкнутому состоянию входа.

3.4 Ввод аналоговых сигналов A0..A7 модуля.

Ввод аналоговых сигналов модуля A0…A7 осуществляется через порт P6 процессора MSP430F149 на 12-и разрядное АЦП в соответствии показанной таблицей .

A0	A1	A2	A3	A4	A5	A6	A7
P6.6	P6.7	P6.4	P6.5	P6.2	P6.3	P6.0	P6.1

Для ввода аналоговых сигналов необходимо на соответствующие контакты A0…A7 подать токовый сигнал 0…20 мА. При этом на внутреннем измерительном резисторе (120 Ом) возникает напряжение 0…2400 мВ, которое подается на вход АЦП процессора.

Для снижения погрешности измерения токового сигнала при производстве модуля GM осуществляется прецизионное измерение "токовых" резисторов для каждого канала и запись соответствующих точных значений коэффициентов преобразования в идентификационный сектор последовательной флэш-памяти (см. п 5).

Значение токового сигнала [мА] рассчитывается по формуле:

$$I = Ki \cdot Nadc \qquad (1);$$

где: $Nadc$ – измеренное значение АЦП;
Ki - коэффициент преобразования тока [мА].

Коэффициент преобразования тока определяется:

$$Ki = Kv / Ri \qquad (2);$$

где: Kv – коэффициент преобразования напряжения [мВ];
Ri - значение "токового" сопротивления [Ом].
*(точное значение **Ki** для каждого канала записано в идентификационном секторе последовательной флэш-памяти (см. п. 5)*

Коэффициент преобразования напряжения определяется:

$$Kv = Vref / 4095 \qquad (3);$$

где: ***Vref*** – значение опорного напряжения АЦП [мВ];
*(в модуле GM **Vref** выбрано равным 2,5В (2500 мВ))*

Кроме входных токовых сигналов АЦП может измерять температуру корпуса процессора ***T*** [ºС] и напряжение питания процессора ***Vcc***[мВ] по формулам:

$$T = (Kv \cdot Nadc - 986) / 3{,}55 \qquad (4);$$

$$Vcc = Kv \cdot Nadc \cdot 2 \qquad (5).$$

Функции ввода аналоговых сигналов A0…A7	
`void IniADC12 (void);`	*Инициализация аналогового ввода* Инициализация АЦП с опорным напряжением 2.5В, с частотой опроса входов – 50 Гц.
`float DatADC12(byte n_ch);`	*Ввод аналоговых сигналов A0…A7, значения температуры и напряжения процессора* n_ch = 0…7 - функция возвращает значения токового сигнала соответствующего канала [мА]; n_ch = 8 – функция возвращает значение температуры корпуса процессора [ºС]; n_ch = 9 – функция возвращает значение напряжения питания процессора [мВ].

3.5 Последовательные COM порты.

Процессор MSP430F149 имеет два последовательных порта – UART0 и UART1. В модуле GM UART1 предназначен для работы с модемом WISMO-QUIK, а UART2 для работы в внешними интерфейсами RS-485 или RS-232. При этом, кроме основных информационных сигналов **TxD0,RxD0,TxD1,RxD1** используются дополнительные управляющие сигналы:

- для модема WISMO_QUIK: **DCD**, **DTR** (см п. 3.12) ;
- для интерфейса RS-485: **сигнал “прием/передача”**(см п. 3.13);
- для интерфейса RS-232: дополнительные управляющие сигналя **не используются**.

Последовательные COM порты	
void IniCom(byte s_com, word b_rate, byte f_dat);	*Инициализация COM порта* s_com – выбор порта (0-UART0;1-UART1) b_rate – скорость обмена (300...57600 бит/сек) f_dat – формат данных (см файл **types_gm.h**)
byte ReadyRcvCom(byte s_com)	*Готовность порта для приема* Знач. функ. bit.0,1 = 0 - нет готовности приема; bit.0 = 1 - байт принят; bit.1 = 1 - ошибка приема байта * ;
byte ReadyTrsCom(byte s_com)	*Готовность порта для передачи* bit.0 = 1 - буффер передачи пуст; bit.1 = 1 - буффер передачи пуст и байт передан (выходной регистр сдвига пуст) ** ;
byte RcvCom(byte s_com)	Прием байта Знач. функции: значение буфера порта
void TrsCom(byte s_com, byte dat)	*Передача байта* dat - передаваемый байт

Пример:

```
byte i;
void main(void)
{
 IniCom(0,19200,(_8|_N|_1)); //Иниц. порта для работы с модемом (19200, 8N1)
 IniCom(1,38400,(_7|_E|_2)); //Иниц. порта для интерфейса RS-232 (38400,7E2)
 while(1)
  {
   while (ReadyRcvCom(0));    // Ожидаем готовность приема от модема
   i = RcvCom(0);             //Принимаем байт от модема
   while (ReadyTrsCom(1));    //Ожидаем готовность передачи по интерфейсу RS-232
   TrsCom(1,i);               //Передаем байт по интерфейсу RS-232
  }
}
```

Примечания:

* - ошибка приема байта формируется при перезаписи байта во входном буфере и при сбоях в приеме. Данный признак удобно использовать при приеме блока информации. При обнаружении ошибки приема байта, блок сразу считается испорченным и прием прекращается.

** - признак “выходной регистр сдвига пуст” необходимо использовать в случаях, когда необходимо иметь информацию о физическом окончании передачи байта в линию, например при работе с интерфейсом RS-485, когда осуществляется переключения прием/передача. **Игнорирование данного признака приводит к сбоям работы интерфейса RS-485.**

3.6 Таймер_А

Таймер_А процессора MSP430F149 используется для формирования точных интервалов времени обеспечивающих работу различных устройств модуля GM, например таймер_А используется в АЦП для формирования интервалов преобразования.

Таймер_А настраивается функциями БПО в режим перезагрузки с частотой прерываний 100Гц.

Таймер_А	
`Void IniTimer_A (void);`	*Инициализация таймера* Инициализация на частоту прерываний 100 Гц
`Void Sleep (word time);`	*Задержка на время time*(1/100) сек.* 1<time<65535
`word CentSec();`	*Значение счетчика сотых долей сек.* Функция обычно используется для ограничения времени работы различных процессов.

3.7 Интерфейс I2C.

В модуле GM существуют устройства такие как флэш-память и RTC таймер, передающие данные по каналу I2C. Процессор MSP430F149 не имеет аппаратного порта I2C, поэтому порт реализован программным способом функциями БПО, при этом линия SDA (данные) соединена с портом P3.3 процессора, а линия SCL соединена с портом P3.2 процессора.

При самостоятельной программной реализации порта I2C программист должен помнить, что выходные порты процессора MSP430F149 имеют симметричную структуру (имеется “верхний” управляющий транзистор). Попытка установить сигналы линий SDA,SCL порта I2C в единичное состояние путем записи “1” в выходной порт процессора приведет к сбоям работы порта I2C, из-за возникновения сквозных токов через “верхние” транзисторы порта процессора и “нижние” транзисторы портов периферийных устройств I2C.

Для устранения указанного эффекта необходимо постоянно записать в выходной регистр процессора значение “0”, а управление осуществлять регистром направления передачи порта. При этом единичное состояние на линиях интерфейса I2C будет возникать за счет внешних резисторов, подтягивающих линии интерфейса I2C к уровню напряжения источника питания.

Интерфейс I2C	
`Void IniI2C (void);`	*Инициализация порта I2C*
`Void StartI2C (void);`	*Старт протокола I2C*
`Void StopI2C (void);`	*Стоп протокола I2C*
`Byte AckI2C (void);`	*АСК протокола I2C* Функ. возвращает: 0 – есть ACK 1 – нет ACK
`Void PutByteI2C(dat);`	*Передача байта dat*
`Byte GetByteI2C();`	*Прием байта*

3.8 Последовательная флеш-память.

Модуль GM содержит последовательную флэш-память 24LC256 объемом 32Кбайт. При этом в этом объеме памяти выделено 256 старших байт для хранения идентификационной и настроечной информации (см. п. 5). Для управления записью в микросхему флэш-памяти существует дополнительный сигнал WP, соединенный с портом P2.2 процессора MSP430F149. Для разрешения записи необходимо установить этот сигнал в “0” состояние. “Slave” адрес микросхемы флэш-памяти – 0xA2 (запись),0xA3(чтение).

БПО содержит функции работы с последовательной флэш-памятью, причем в этих функциях запрещена запись в идентификационную область и реализован побайтовый произвольный доступ. Для ускорения работы памяти программист может реализовать побайтовый последовательный (групповой) доступ, при этом необходимо не допускать запись в идентификационную область.

Последовательная флэш-память	
`Void IniFMem (void);`	*Инициализация памяти*
`Byte ReadFMem(adr);`	*Чтение памяти по адресу adr* Функ. возвр. байт данных.
`Byte WriteFMem(adr,dat)`	*Запись данных в память* Функ. возвр.: 0 – запись прошла успешно 1 – ошибка записи 2 – попытка записи в идентиф. область

3.9 Часы и календарь RTC.

Модуль GM содержит микросхему часов RTC последовательного доступа PCF8583T, которая при отсутствии напряжения питания работает от литиевой батареи.

БПО содержит функции работы с часами и календарем, при этом в процессоре выделяется буфер для хранения часов и календаря RTC.

Для корректного чтения данных с RTC необходимо сначала считать **все** регистры часов и календаря в буфер, а затем считывать конкретные "замороженные" данные из буфера. Для записи данных в RTC необходимо **сначала сформировать весь буфер часов и календаря**, а затем записывать **разом** весь буфер в RTC. Все данные представлены в **HEX формате**.

Часы и календарь RTC	
`void IniRTC ();`	*Инициализация (запуск) RTC*
`byte ReadRTC(s_dat);`	*Чтение данных с RTC* s_dat = 0 - секунды (0-59) (из буфера) s_dat = 1 - минуты (0-59) (из буфера) s_dat = 2 - часы (0-23) (из буфера) s_dat = 3 - день (1-30(31)(28)(29)) (из буфера) s_dat = 4 - месяц (1-12) (из буфера) s_dat = 5 - год (0-99) (из буфера) s_dat >5 - обновление буфера Функцмя возвращает байт данных в зависимости от s_dat в HEX формате.
`void WriteRTC(s_dat, dat);`	*Запись данных в RTC* При s_dat =0…5 dat – данные в HEX формате, записываются в буфер. При s_dat>5 данные из буфера записываются в RTC.

Пример:

```
byte sec,min;
void main (void)
{
 IniRTC();

 ReadRTC(0xFF);      //Обновление буфера данных с RTC
 sec = ReadRTC(0);   //Считать значение секунд (из буфера)
 min = ReadRTC(1);   //Считать значение минут (из буфера)


 WriteRTC(0, 0);     //Запись в буфер 14:30:00  22.11.2002
 WriteRTC(1,30);
 WriteRTC(2,14);
 WriteRTC(3,22);
 WriteRTC(4,11);
 WriteRTC(5, 2);
 WriteRTC(0xFF,0);  //Данные из буфера записываются в RTC
}
```

3.10 Монитор CLK.

Процессор MSP430F149 может иметь несколько источников системной тактовой частоты. Начинает процессор работать от внутреннего генератора тактовой частоты. Поэтому необходимо осуществить переключение на тактовую частоту от внешнего кварцевого резонатора 2,4576 МГц. В процессе работы возможны сбои данного генератора, при этом процессор автоматически переходит на работу от внутреннего генератора тактовой частоты. Необходимо периодически проверять правильность выбора тактовой частоты и при сбоях осуществлять восстановление работы генератора от внешнего кварцевого резонатора.

Переключение генераторов CLK и опрос состояния процессора осуществляется функциями БПО.

Монитор CLK	
`void IniCLK();`	*Инициал. сист. частоты* Переключение на частоту 2,4576 МГц
`byte StateCLK();`	*Состояние сист. частоты* Функция возвращает: 0 – нормальная работа 1 – сбой системной частоты

3.11 Монитор PW_OFF.

При отключении питания и снижении напряжения на выходе стабилизатора до 4,5В внутренний компаратор процессора MSP430F149 вырабатывает аварийный сигнал, по которому процессор должен корректно завершить свою работу.

Опорное напряжение 2,5В подается на вход CA0 компаратора с выхода VREF+ источника опорного напряжения АЦП. На второй вход CA1 компаратора подается напряжение с делителя выходного напряжения стабилизатора 5В. Делитель подобран таким образом, что когда напряжения с выхода стабилизатора становится меньше 4,5В, на входе CA1 компаратора напряжение становится меньше 2,5В и срабатывает компаратор. При этом процессор еще некоторое время сохраняет работоспособность и должен осуществить корректное завершение работы.

Функции БПО осуществляют инициализацию компаратора и опрос его состояния.

Монитор PW_OFF	
`void IniPW_OFF();`	*Инициализация POWER_OFF*
`byte StatePW_OFF();`	*Состояние POWER_OFF* Функция возвращает: 0 – нормальная работа 1 – выключение нап. питания

3.12 Индикация модуля GM.

Модуль GM имеет шесть светодиодных индикаторов, которые **полностью управляются программным способом** от портов процессора MSP430F149 (“1” – светодиод включен, “0” – светодиод выключен).

Четыре желтых светодиода (порты P5.0…P5.3) индицируют уровень радиосигнала модема SIEMENS TC-35.

Зеленый светодиод (порт P5.5) показывает регистрацию модема SIEMENS TC-35 в сети.

Красный светодиод (порт P5.4) индицирует наличие электропитания.

Поскольку все светодиоды модуля GM полностью программно управляемые, то программист может присвоить индикаторам другие альтернативные функции, исходя из своей конкретной задачи.

Функции БПО управляют индикацией модуля. Для осуществления режима мигания используется таймер_A.

Индикация модуля GM	
`void IniIndGM ();`	*Инициализация индикации модуля GM*
`void IndGM (s_ind, ind);`	*Индикация модуля GM* s_ind = 0 - индикация уровня сигнала модема ind = 0…31 – качество сигнала s_ind = 1 – индикация регистрации модема ind = 0 – индикатор погашен ind = 1 – индикатор горит ind= 2 – индикатор мигает s_ind = 1 – индикация питания модуля GM ind = 0 – индикатор погашен ind = 1 – индикатор горит ind= 2 – индикатор мигает

3.13 Управляющие сигналы модема WISMO-QUIK.

Для управления модемом WISMO-QUIK фирмы WAVECOM кроме последовательного интерфейса (см п.3.4) используются следующие сигналы:

- сигнал **ON/OFF (включение/выключение модема)** управляется с порта P5.7,
  "1" – включение модема;
- сигнал **RST_WCOM (сброс модема)** управляется с порта P5.6,
  "1" – сброс модема;
- сигнал **DTR0(готовность модема)** соединен с портом P3.1,
  обычно используется для аппаратного разрыва связи;
- сигнал **DCD0(обнаружение несущей частоты)** соединен с портом P2.7,
  используется для аппаратного обнаружения состояния "CONNECT".

Управляющие сигналы модема WISMO-QUIK (WAVECOM)	
`void IniDirWavecom();`	*Инициализация сигналов модема*
`byte DirWavecom(s_sig,sig);`	*Управляющие сигналы модема* s_sig : =0 ON/OFF - включение модема (sig=1) = 1 RESET_WCOM - сброс модема (sig=1) = 2 DTR0 - сигнал DTR модема (sig=1 - DTR on) = 3 DCD0 - прием сигнала DCD модема (1-обнаружена несущая частота)

3.14 Управление портом RS-485.

Для управления направлением передачи по интерфейсу RS-485 используется порт процессора P3.0 ("1" – RS-485 работает на передачу, "0" – RS-485 работает не прием). При управлении микросхемой порта RS-485 необходимо учесть замечания п. 3.4.

Управление портом RS-485	
`void IniDir485 ();`	*Инициализация управления RS485*
`void Dir485 (dir);`	*Управление портом RS485* dir = 0 - RS485 работает на прием =1 - RS485 работает на передачу

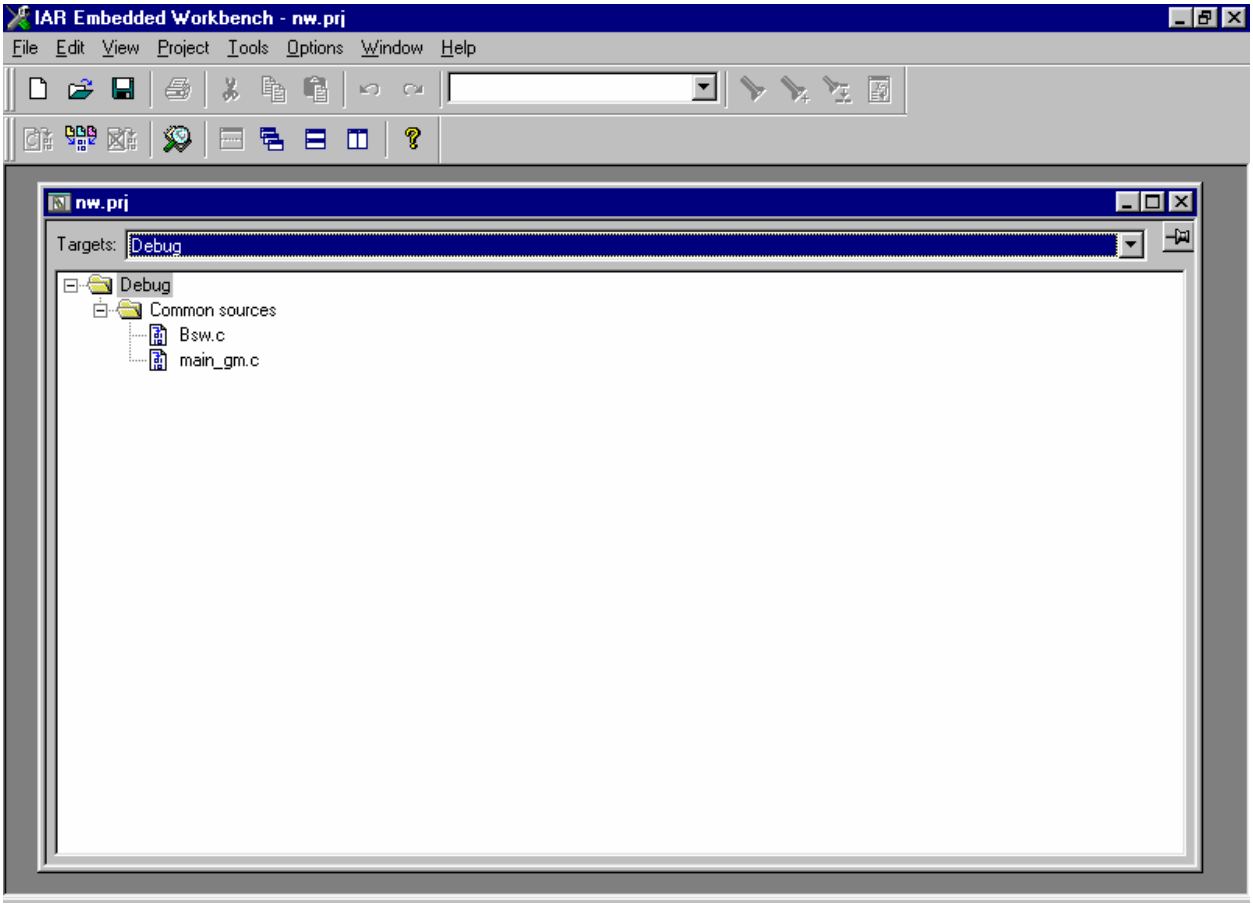
4. Отладка и запись программ в модуль GM.

Для отладки и записи программ в модуль GM установить на рабочем PC компьютере отладочную среду, для этого в папке “Tools” запустить файл “**FET_R301.exe”.**

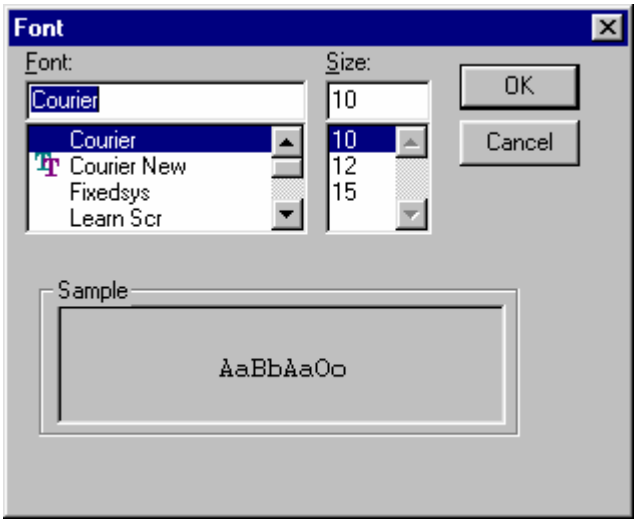
Перед началом работы рекомендуется ознакомится с описаниями среды “IAR Embedded Workbench” и отладчика “C-SPY”.

Далее, для примера рассмотрим работу с демонстрационной программой “main_gm.c”, расположенной в папке “Soft_gm”. Для этого скопируем указанный файл и файлы “bsw.c”,”bsw.h”,”types_gm.h” в любую папку рабочего компьютера.

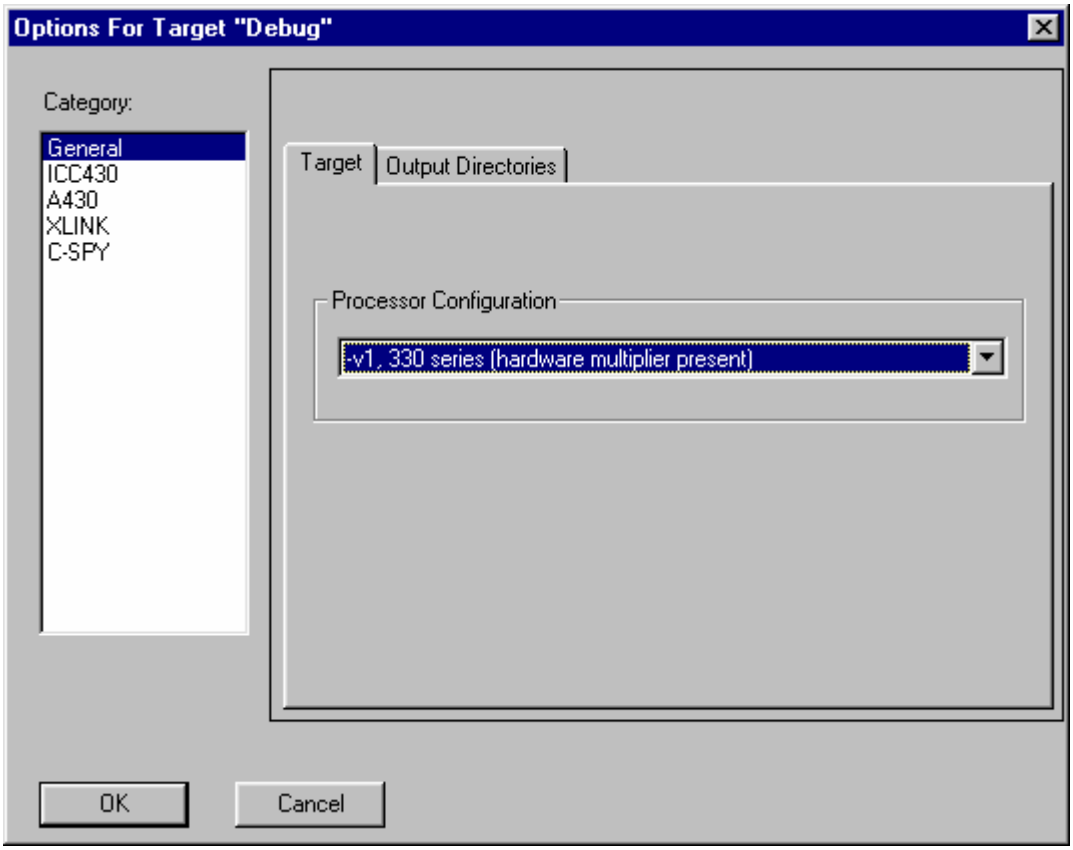
Запускаем среду и открываем новый проект “file→new→project”. В окне “New Project” указываем нашу папку, имя проекта и открываем проект. Далее необходимо указать, какие файлы входят в проект “project→files…”. В окне “Project files” добавим (Add) файлы “main_gm.c”и“bsw.c”. В результате окно проекта должно иметь вид:



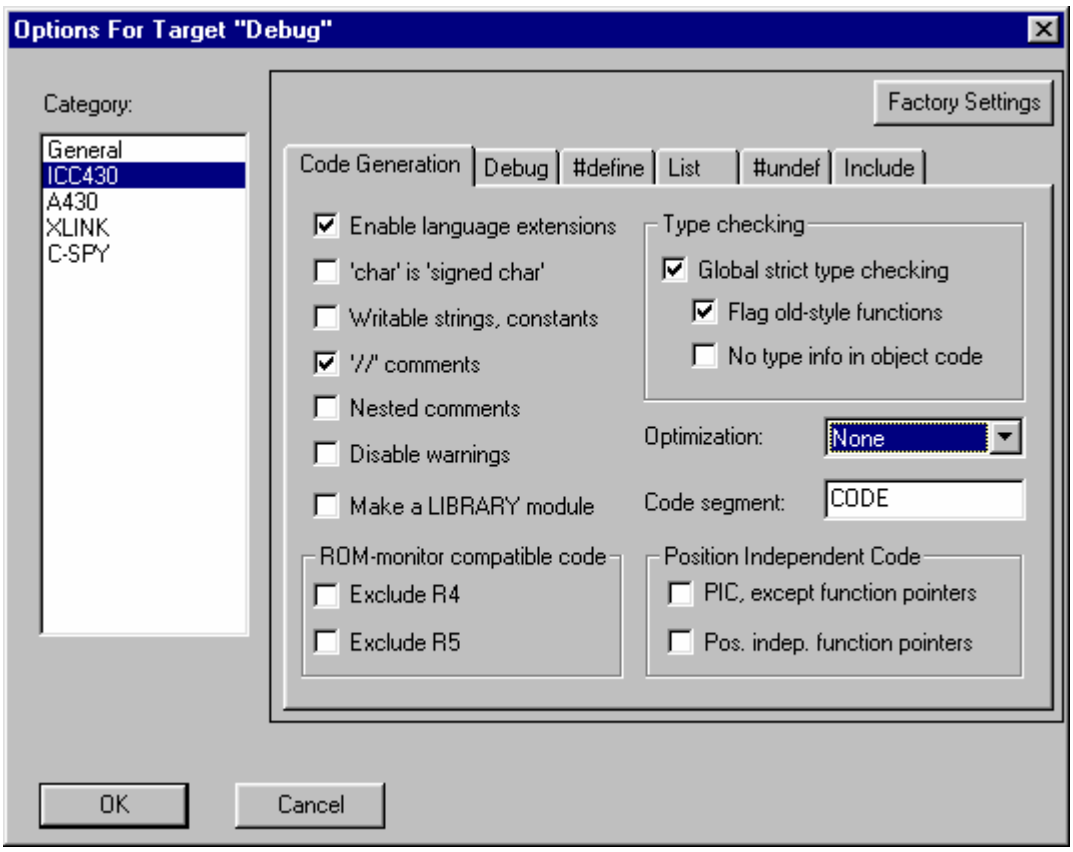
Для правильного отображения текста на русском языке рекомендуется установить шрифт Courier (10) “Options→Setings…”.



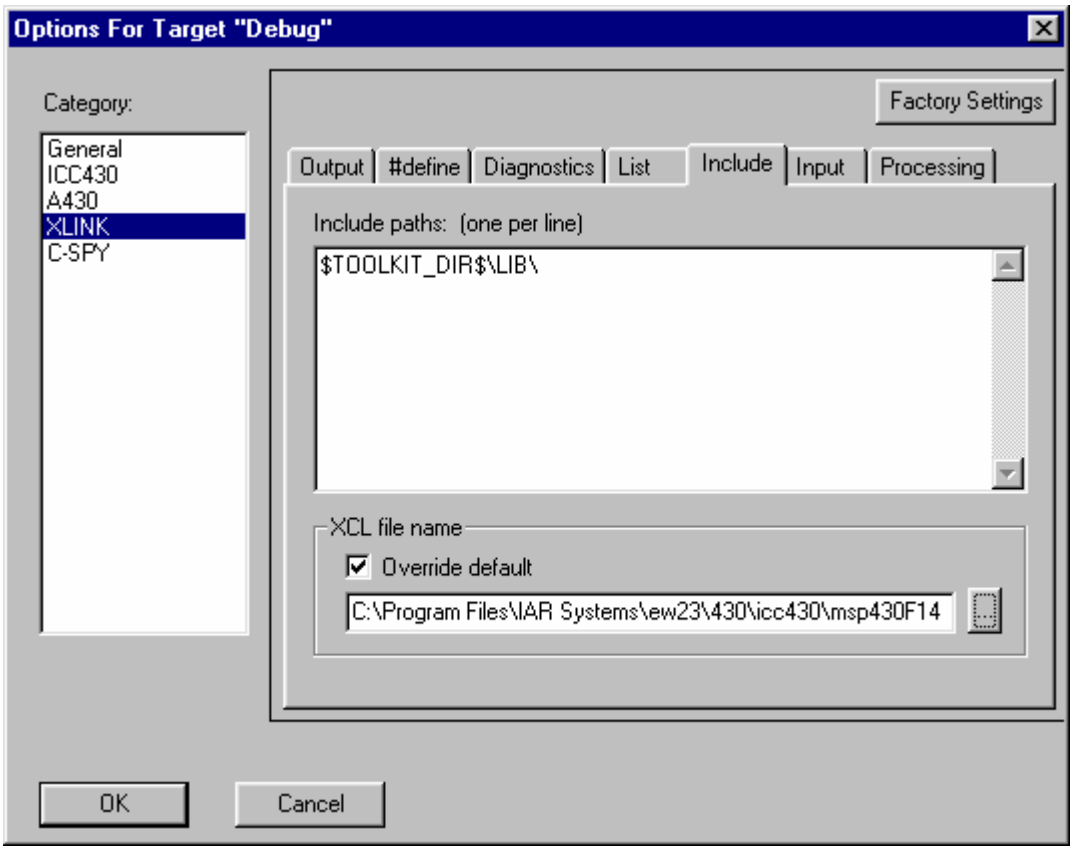
Для нормальной работы необходимо сделать **настройки проекта:** “Project→Options…”. В основных настройках выбрать ”-V1,330 series (hardware multiplier presents)”.



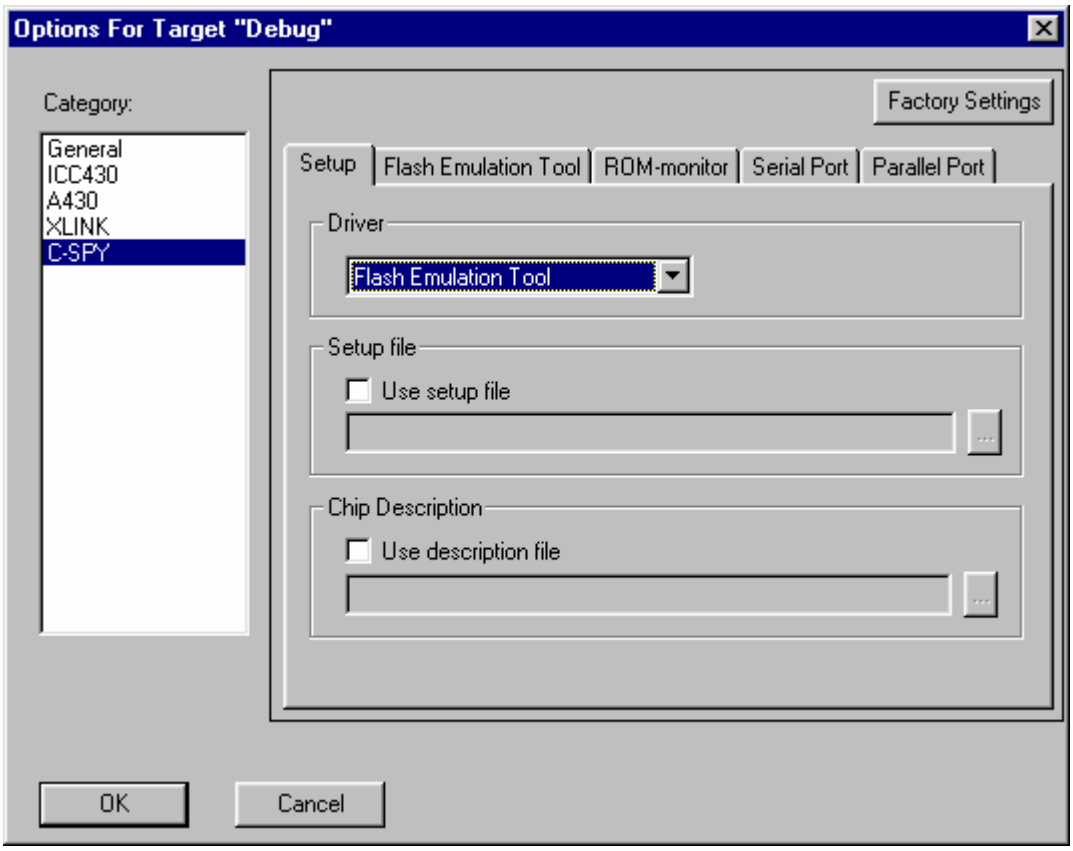
В настройках компилятора ICC430 рекомендуется **отключить оптимизацию** “Optimization: None”, это избавит в дальнейшем от ряда “загадочных” сюрпризов.



В настройках линкера XLINK, в разделе “include” подключить XCL файл “**msp430F149C.xcl**”.



В настройках отладчика C-SPY указать “Flash Emulation Tool” и выбрать нужный параллельный порт



После настройки отладочной среды следует компилировать и линковать проект “Project→Make”.

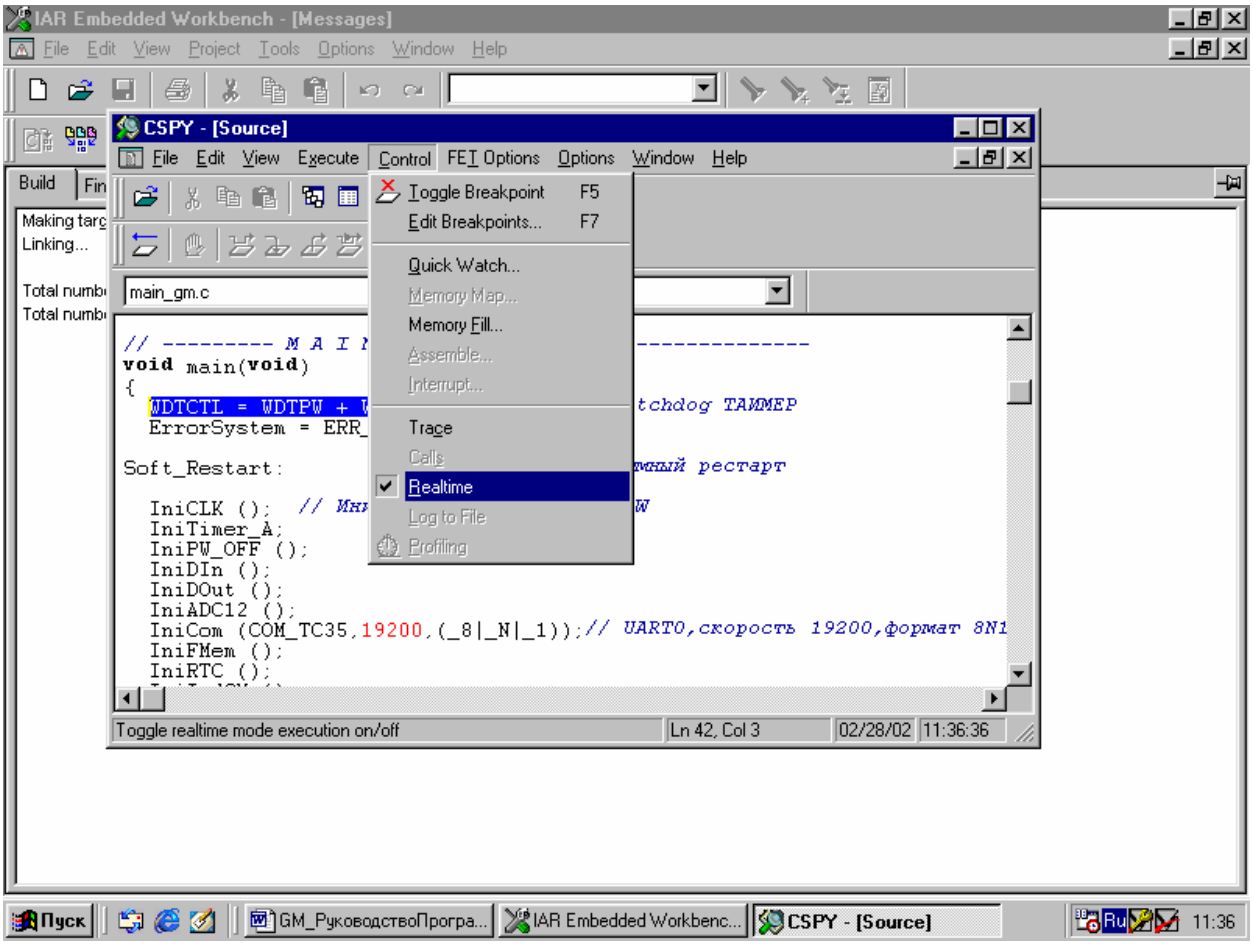
Для загрузки программы в модуль GM необходимо соединить 14-контактным ленточным кабелем модуль GM с преобразователем JTAG интерфейса и подключить преобразователь к параллельному (LPT) порту компьютера.

ВНИМАНИЕ! Во время отладки и программирования внешний супервизор должен быть ОТКЛЮЧЕН (перемычка J1 – РАЗОМКНУТА).

Включить питание модуля GM и запустить отладчик “Project→Debugger”. При запуске отладчика происходит загрузка программы в флэш-память программ процессора MSP430F149.

Для правильного отображения текста на русском языке в отладчике C-SPY рекомендуется установить шрифт Courier (10) “Options→Setings…”.

При отладке программы в условиях реальных процессов следует устанавливать режим “Realtime”, при этом процессор гарантированно работает на своей реальной рабочей частоте, однако пошаговый режим работы недоступен и возможны только три точки останова.



Для отображения информации во время отладки можно открыть окна переменных, памяти, регистров и т.д.

После завершения отладки программы питание модуля GM можно отключить и отсоединить 14 контактный ленточный кабель. После повторного включения питания, записанная программа начнет работать в модуле GM. Если в программе используется внешний супервизор, можно установить перемычку J1 (при выключенном питании).

Если необходимо записать программу без отладочной информации, то устанавливаем в окне проекта “Targets→Relese”, конфигурируем проект, как было указано выше, только без отладочной информации, осуществляем компиляцию и линкование проекта, вызываем отладчик и производим запись программы в процессор.

5. Запись идентификационной информации в модуль GM.

Для записи идентификационной информации, точных значений “токовых” резисторов АЦП и установки часов RTC служит программа “**setup_gm.c**”. Вся эта информация **записывается при производстве модуля GM**, однако в процессе эксплуатации может возникнуть необходимость в изменении (восстановлении) указанной информации, и можно воспользоваться данной программой.

В модуль GM записывается следующая идентификационная информация (приведена в паспорте модуля GM):

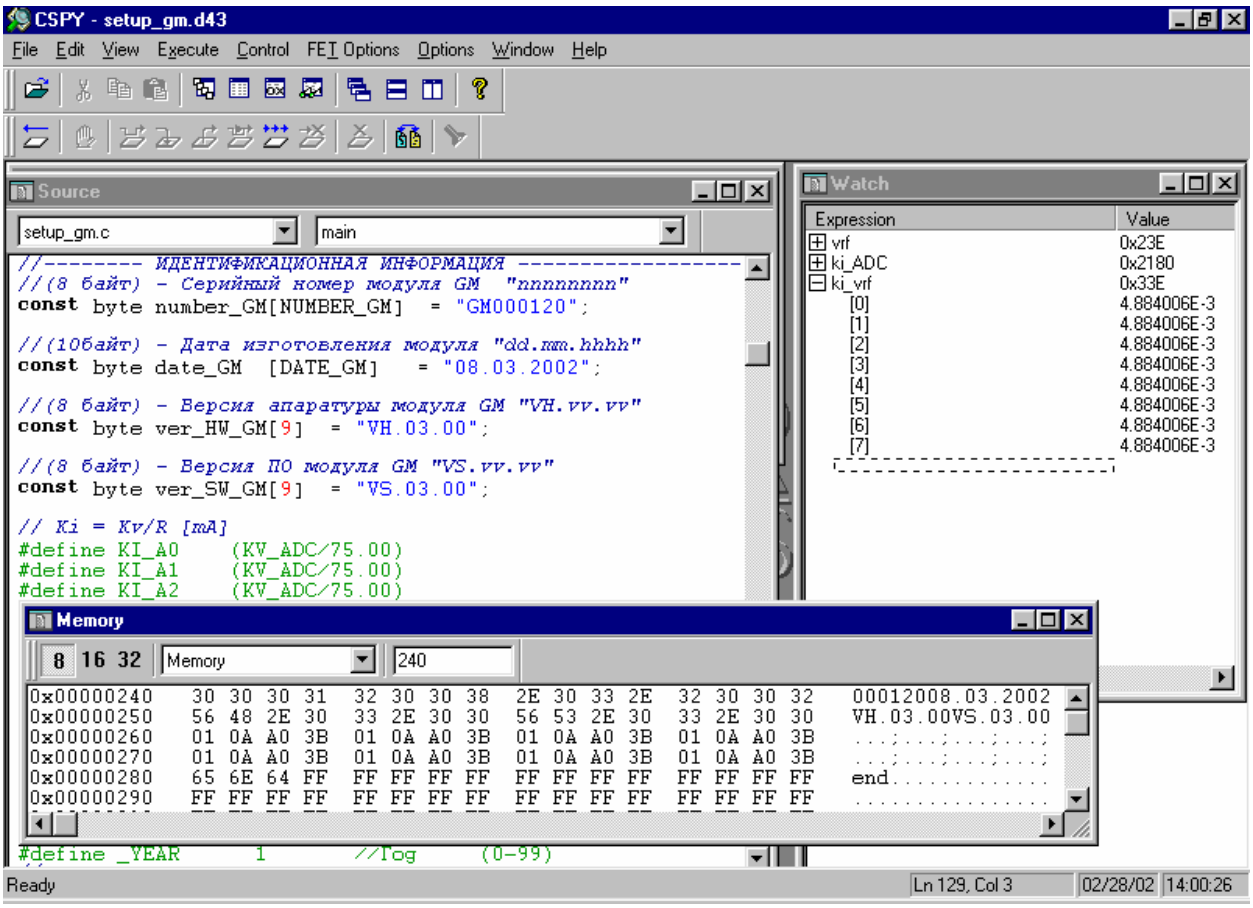
- серийный номер модуля GM (8 байт);
- дата изготовления модуля GM (день.месяц.год);
- версия аппаратной части модуля GM (VH.vv.vv);
- версия программного обеспечения модуля GM (VS.vv.vv).

Для увеличения точности АЦП, для каждого его канала записываются точные значения “токовых” измерительных резисторов, номинал которых (120 Ом) должен быть записан **с точностью до сотых долей** Ома.

Управление компиляцией программы осуществляется макросами WRITE_READ и SET_TIME.

```
#define WRITE_READ    0   // 1 - ЗАПИСЬ И ЧТЕНИЕ идентификац. информации
                          // 0 - ТОЛЬКО ЧТЕНИЕ   идентификац. информации
#define SET_TIME      0   // 1 - УСТАНОВИТЬ заданное время
                          // 0 - НЕ УСТАНАВЛИВАТЬ время
```

Просматривать идентификационную информацию можно через отладчик C-SPY**, при этом перемычка J1 должна быть РАЗОМКНУТА.**.



Следует обратить внимание, что после записи идентификационной информации или после установки часов, записанная программа остается в процессоре, и будет запускаться при каждом рестарте процессора.

Поэтому после запуска программы “setup_gm.c” в режиме записи идентификационной информации или установки часов, следует сразу после этого записать в процессор данную программу в режиме “**только чтение**”, или любую другую программу, которая будет выполняться в процессоре каждый раз при включении питания.

Приложение 1.

Перечень функций базового программного обеспечения (БПО) V 5.00.

Работа с внешним супервизором	`void IniExternalWD();`	*Инициализация супервизора*	п3.1
	`void ExternalWD();`	*Сброс сторожевого таймера*	п3.1
Ввод дискретных сигналов D0…D7	`void IniDIn();`	*Инициализация диск. ввода*	п.3.2
	`byte DIn();`	*Ввод сигналов D0…D7*	п.3.2
Вывод дискретных сигналов D8…D16	`void IniDIn16();`	*Инициализация диск. ввода*	п.3.3
	`byte Din16();`	*Ввод сигналов D8…D15*	п.3.3
Ввод аналоговых сигналов A0…A7	`void IniADC12 ();`	*Инициализация анал. ввода*	п.3.4
	`float DatADC12(n_ch);`	*Ввод сигналов A0…A7*	п.3.4
Последовательные COM порты	`void IniCom(s_com,b_rate,f_dat);`	*Инициализация COM порта*	п.3.5
	`byte ReadyRcvCom(s_com);`	*Порт готов для приема*	п.3.5
	`byte ReadyTrsCom(s_com);`	*Порт готов для передачи*	п.3.5
	`byte RcvCom (s_com);`	*Прием байта в COM порт*	п.3.5
	`void TrsCom (s_com,dat);`	*Передача байта через порт*	п.3.5
Таймер _A	`void IniTimer_A();`	*Инициализация таймера*	п.3.6
	`void Sleep (time);`	*Задержка time*(1/100) сек.*	п.3.6
	`word CentSec();`	*Значение счетчика сотых сек.*	п.3.6
Передача данных через I2C	`void IniI2C();`	*Инициализация I2C*	п.3.7
	`void StartI2C();`	*Старт протокола I2C*	п.3.7
	`void StopI2C();`	*Стоп протокола I2C*	п.3.7
	`byte AckI2C();`	*ACK протокола I2C*	п.3.7
	`void PutByteI2C(dat);`	*Передача байта dat*	п.3.7
	`byte GetByteI2C();`	*Прием байта*	п.3.7
Последовательная флеш память	`void IniFMem();`	*Инициализация памяти*	п.3.8
	`byte ReadFMem(adr);`	*Чтение памяти по адресу adr*	п.3.8
	`byte WriteFMem(adr,dat)`	*Запись данных в память*	п.3.8
Часы и календарь RTC	`void IniRTC ();`	*Инициализация (запуск) RTC*	п.3.9
	`byte ReadRTC(s_dat);`	*Чтение данных с RTC*	п.3.9
	`void WriteRTC(s_dat, dat);`	*Запись данных в RTC*	п.3.9
Монитор CLK	`void IniCLK();`	*Инициал. сист. частоты*	п.3.10
	`byte StateCLK();`	*Состояние сист. частоты*	п.3.10
Монитор PW_OFF	`void IniPW_OFF();`	*Инициализация POWER_OFF*	п.3.11
	`byte StatePW_OFF();`	*Состояние POWER_OFF*	п.3.11
Индикация модуля GM8/18-485M	`void IniIndGM ();`	*Инициал. индикации модуля*	п.3.12
	`void IndGM (s_ind, ind);`	*Индикация модуля*	п.3.12
Управляющие сигналы WAVECOM	`void IniDirWavecom();`	*Инициализация сигналов модема*	п.3.13
	`byte DirWavecom(s_sig,sig);`	*Управляющие сигналы модема*	п.3.13
Управление портом RS485	`void IniDir485 ();`	*Инициал. управл. RS485*	п.3.14
	`void Dir485 (dir);`	*Управление портом RS485*	п.3.14

Приложение 2.

Обмен информацией по каналу данных GSM.

Для ***демонстрации*** работы модуля GM по каналу данных GSM в комплект поставки входит демонстрационная программа **main_gm.c.** Отладка и запуск этой программы описаны в п. 4. (перемычку J1 необходимо снять).

Для организации обмена необходим GSM терминал **SIEMENS TC35**, который подключается на COM порт "диспетчерского" компьютера.

В качестве "диспетчерского" компьютера может выступать второй удаленный компьютер, или тот же рабочий компьютер на котором ведется программирование и отладка модуля GM.

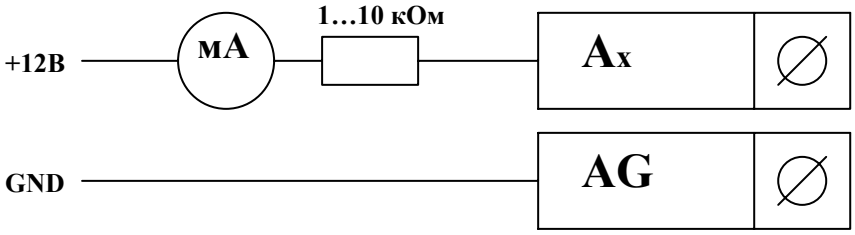
На "диспетчерский" компьютер необходимо установить телеметрическую программу **GMTest.exe.** Руководство по установке и работе с данной программой приводится в файле **gmtest.htm,** который находится в папке **GMTest-setup .**

Далее необходимо установить в модуль GM и терминал SIM карты с открытым каналом данным и **снятым PIN кодом**.

После включения питания должен загореться красный светодиод "питание", и примерно через 30 сек происходит инициализация модема и загораются индикаторы "регистрация в сети GSM" и "уровень сигнала". Если произошла регистрация модема в сети GSM, то зеленый светодиод горит постоянно, в противном случае он мигает. После этого можно с диспетчерского компьютера осуществить связь с модулем GM, получить текущее состояние дискретных и аналоговых входов, а так же установить дискретные релейные выходы.

Для имитации сигналов на дискретных входах можно замыкать соответствующий D-вход с "цифровой землей" DG.

Для имитации сигнала на аналоговых входах можно на соответствующие A-входы подать токовый сигнал 0…20 мА как показано ниже.



ВНИМАНИЕ! Превышение тока выше указанных величин может вывести выходу из строя процессора модуля GM.

Возможные неисправности:

- нет регистрации в сети – проверить снятие PIN кода с SIM карты, используя мобильный телефон или терминал на "диспетчерском" компьютере, проверить качество радиосигнала;
- нет связи по каналу данных – проверить, открыт ли канал данных на SIM карте.

Приложение 3.

Использование модуля GM в “прозрачном” (терминальном) режиме.

Иногда возникает необходимость использования модуля GM в режиме терминала, при этом данные с интерфейса модема WISMO-QUIK напрямую передаются на интерфейс модуля GM, однако в отличие от обычного терминала SIEMENS TC-35 возможно:

- использование интерфейса RS-485;
- преобразование формата данных;

Однако существуют некоторые ограничения:

- затруднительно использовать индикацию регистрации в сети GSM и качества сигнала, т.к. эта информация получается программным путем, посылкой модему соответствующих AT команд, в общем случае это нарушает “прозрачность” режима работы;
- невозможно использовать дополнительные управляющие сигналы интерфейса RS-232.

Для ***демонстрации*** “прозрачного” (терминального) режима поставляется программа **terminal.c**, которую необходимо записать в модуль и запустить, как было указано выше.

Программа работает с интерфейсами RS-232 и RS-485. Для вывода информации на PC компьютер можно использовать любую стандартную терминальную программу. Схему соединения см. **“Техническое описание и руководство по эксплуатации”.**