# Описание программы-загрузчика для микроконтроллеров AT90CAN128 и AT90CAN64

# Общая информация о программе-загрузчике

Возможность обновления программного обеспечения ячеек по линии связи CAN обеспечивается программой-загрузчиком.

При подаче питания на МК происходит запуск программы с адреса 0x1E000 (должны быть установлены значения **TRUE** для фьюзов **BOOTRST**, **BOOTSZ1** и **BOOTSZ1**). По данному адресу располагается программа-загрузчик, которая в течение двух секунд после запуска ожидает команды по линии CAN на перепрограммирование ячейки. Если команды на перепрограммирование за это время не поступает, то программа-загрузчик проверяет целостность основной программы, расположенной по адресу 0x00000, и, если нарушений целостности не выявлено, производит переход на адрес 0x00000, то есть, запускает основную программу. Если целостность основной программы нарушена, то программа-загрузчик продолжает ожидать команду по CAN на перепрограммирование и переход на адрес 0x00000 не производит.

Память программ контроллера AT90CAN128 при использовании CAN-загрузчика имеет структуру, представленную в Табл. 1. Помимо областей основной программы и программы загрузчика, в памяти программ зарезервировано ещё два блока по 256 байт: таблица размещения блоков основной программы и словарь свойств ячейки. При первоначальном программировании ячейки внутрисистемным программатором (например, AVRISP) данная структура памяти формируется программой первоначального программирования ячеек, а при последующим обновлении ПО ячейки – программой-загрузчиком.

**Таблица размещения блоков основной программы (Табл.2)**

Загрузчиком предусмотрена возможность формирования области основной программы из нескольких бинарных файлов (но не более пяти). Каждому из таких файлов будет соответствовать свой блок данных в области основной программы, а блоки описаны таблицей размещения блоков (Табл. 2).

Но, как правило, основная программа представляет собой единственный бинарный файл прошивки, располагаемый начиная с адреса 0x00000. В этом случае таблица размещения блоков содержит единственную запись.

Таблица 1. Карта памяти программ МК AT90CAN128 при использовании CAN-загрузчика

|  |  |
| --- | --- |
| **Диапазон адресов** | **Содержимое области** |
| 0x00000…  0x1DDFF | Область основной программы |
| 0x1DE00…  0x1DEFF | Таблица размещения блоков основной программы (Табл.2) |
| 0x1DF00..  0x1DFFF | Словарь свойств ячейки (Табл.3) |
| 0x1E000  0x1FFFF | Область программы-загрузчика |

Таблица 2. Таблица размещения блоков основной программы

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Смещение (байт)** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | … |
| **Назначение** | Количество блоков основной программы | Начальный адрес блока №1 | | | | Размер блока №1 | | | | CRC-16 блока №1 | | Начальный адрес блока №2 | | | | … |

**Словарь свойств ячейки**

Программой-загрузчиком предусмотрено хранение значений следующих свойств ячейки: тип ячейки, её серийный номер, версия и подверсия, контрольная сумма программного обеспечения и т.п. (Табл. 3) в специально отведённой области, называемой словарём параметров. Значения из словаря параметров могут быть использованы как программой-загрузчиком для идентификации ячейки в процессе перепрограммирования по линии CAN, так и основной программой для получения значения каких-либо постоянных свойств, например, версий программного обеспечения для формирования ответа на запрос AUX\_RESOURCE. Формат хранения свойств представлен в Табл. 4. Всего может быть не более 50 свойств.

Таблица 3. Параметры, идентифицирующие ячейку

|  |  |
| --- | --- |
| **Код (ключ)** | **Свойство** |
| 1 | Версия программы |
| 2 | Подверсия программы |
| 6 | Контрольная сумма программы |
| … | … |
| 129 | ID ячейки |
| 130 | Номер программного модуля |
| 131 | Серийный номер блока |
| 133 | Номер канала (полукомплекта): 1, 2, 3, … |
| 134 | Модификация ячейки |

Таблица 4. Словарь свойств программы

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Смещение (байт)** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | … |
| **Назначение** | Количество параметров N | Код (ключ) параметра №1 | Значение параметра №1 | | | | Код (ключ) параметра №2 | Значение параметра №2 | | | | … |

# Описание алгоритма взаимодействия основной программы с программой-загрузчиком

Для корректного функционирования механизма обновления ПО по линии CAN в основной программе должна быть предусмотрена возможность обработки команды на инициализацию режима программирования **FU\_INIT** (дескриптор 66A8).

При приёме команды **FU\_INIT** основная программа считывает значения идентифицирующих ячейку параметров из словаря свойств ячейки (Табл. 1, 2) и сравнивает их со значениями, содержащимися в команде (Табл. 5). Если все значения одноимённых параметров, полученных из команды по CAN, совпадают со значениями, считанными из словаря свойств, то производится программный сброс ячейки, что приводит к вызову программы-загрузчика. Если в команде **FU\_INIT** значения некоторых параметров равны нулю, а остальные, не равные нулю, совпадают со значениями, считанными из словаря свойств, то основная программа формирует ответное CAN-сообщение **FU\_DEV**, имеющее дескриптор **0x66E8**, с полным набором значений параметров, идентифицирующих ячейку (Табл. 6). Программный сброс в этом случае не производится. Если же имеется хотя бы одно несовпадение, и при этом не совпадающие значения не равны нулю, то данная команда игнорируется.

Таким образом, команда **FU\_INIT**, все значения полей которой отличны от нуля, является командой на программный сброс ячейки, значения свойств которой совпадают с соответствующими значениями, переданными в команде. А команда **FU\_INIT**, некоторые поля которой содержат нулевые значения, является командой на запрос значений свойств ячеек, причём отвечают только те ячейки, значения свойств которых совпадают с соответствующими ненулевыми значениями в команде.

Таблица 5. Формат команды на инициализацию перепрограммирования ячейки **FU\_INIT**

|  |  |  |
| --- | --- | --- |
| **Байт** | **Бит** | **Содержимое** |
| 1 | 7 … 4 | = 07 |
| 3 … 0 |
| 2 | 7 … 4 | = 01 |
| 3 … 0 |
| 3 | 7 … 4 | ID ячейки |
| 3 … 0 |
| 4 | 7 … 4 |
| 3 … 0 | Модификация ячейки |
| 5 | 7 … 4 | Номер программного модуля |
| 3 … 0 |
| 6 | 7 … 4 | Номер канала |
| 3 … 0 | Серийный номер ячейки |
| 7 | 7 … 4 |
| 3 … 0 |
| 8 | 7 … 4 |
| 3 … 0 |

Таблица 6. Формат ответа **FU\_DEV**

|  |  |  |
| --- | --- | --- |
| **Байт** | **Бит** | **Содержимое** |
| 1 | 7 … 4 | = 07 |
| 3 … 0 |
| 2 | 7 … 4 | = 00 |
| 3 … 0 |
| 3 | 7 … 4 | ID ячейки |
| 3 … 0 |
| 4 | 7 … 4 |
| 3 … 0 | Модификация ячейки |
| 5 | 7 … 4 | Номер программного модуля |
| 3 … 0 |
| 6 | 7 … 4 | Номер канала |
| 3 … 0 | Серийный номер ячейки |
| 7 | 7 … 4 |
| 3 … 0 |
| 8 | 7 … 4 |
| 3 … 0 |

# Реализация алгоритма взаимодействия основной программы с программой-загрузчиком

Алгоритм взаимодействия с программой-загрузчиком реализуется в два этапа:

* Реализация функции считывания параметров из словаря свойств загрузчика
* Реализация сообщения от CAN-программатора
  1. Реализация функции считывания параметров из словаря свойств загрузчика

Функция считывания значения свойства из словаря свойств реализована внутри программы-загрузчика. При передаче управления от программы-загрузчика основной программе адрес данной функции записывается в регистровую пару **GPIOR1**, **GPIOR2**. Таким образом, сразу после запуска основная программа должна считать значение из этих регистров и присвоить его указателю на функцию, принимающую в качестве аргумента однобайтовое значение (ключ свойства) и возвращающую четырёхбайтовое значение (значение свойства). Передача параметра в функцию производится через регистр микроконтроллера R16, а результат возвращается через регистры R16 – R19 (в регистре R16 – младший байт четырёхбайтового значения)

При использовании библиотеки CANPROG.C это реализуется добавлением вызова функции **InitGetParamFuncPtr()** в самом начале основной программы:

....  
**#include** “CANPROG.H”//Подключение заголовочного файла CANPROG.H  
...  
**int** main()  
{  
 *InitGetParamFuncPtr();*// Инициализации указателя на функцию чтения свойств  
 ...  
 **while**(1)  
 {  
 ...  
 }  
}

После выполнения функции **InitParamFuncPtr()** параметры из словаря свойств становятся доступными для основной программы через функцию **GetParamValue(unsigned char** key**)**, возвращающую значение типа **unsigned long**.

* 1. Реализация обработчика сообщения от CAN-программатора

В обработчик принимаемых CAN-сообщений следует добавить функцию обработки команды **FU\_INIT**, реализующую алгоритм, описанный в п.2. В библиотеке CANPROG.C[[1]](#footnote-1) это реализуется добавлением вызова функции **ProgRQ\_processing\_buff** в обработчик CAN-сообщений. В качестве аргументов данной функции передаётся двухбайтовое значение дескриптора CAN-сообщения и указатель на массив из восьми элементов однобайтного типа, содержащий восемь байт CAN-сообщения.

**#define** FU\_INIT 0x66A8  
…  
**unsigned char** msgBuff[8];  
**switch** (descr) // Обработка сообщения в зависимости от дескриптора   
{   
 …  
 **case** FU\_INIT:  
 **for** (**char** i=0; i<8; i++) // Заполнение массива msgBuff из FIFO почтового ящика CAN  
 {  
 msgBuff[i] = CANMSG;  
 asm(“NOP”);   
 }  
 ProgRQ\_processing\_buff(descr, msgBuff);  
 **break**;  
}

Альтернативой функции **ProgRQ\_processing\_buff** является функция **ProgRQ\_processing**, которая в качестве единственного аргумента принимает значение номера почтового ящика CAN. Использование данной функции удобно в том случае, если под сообщение **FU\_INIT** зарезервирован отдельный почтовый ящик CAN.

* 1. Считывание значений параметров из словаря свойств для нужд основной программы

Дополнительной возможностью, предоставляемой загрузчиком основной программе, является получение значений произвольных параметров, хранимых в словаре свойств. Например, там хранится информация о версии основной программы и, вызвав функцию **GetParamValue**, можно запросить данную информацию для использования, например, при формировании ответа на запрос AUX\_RESOURCE.

Следует обратить внимание, что при отсутствии программы-загрузчика не будет возможности запроса значений свойств ячейки из области загрузчика, а функция **GetParamValue** всегда будет возвращать нулевое значение, так как указателю на функцию запроса свойств ячейки, расположенную в области программы-загрузчика, не будет присвоено корректное значение. Корректность значения данного указателя проверяется посредством вызова функции **IsGetParamFuncPtrAssigned()**, возвращающей значение 1, если значение указателя корректно, 0 – в противном случае.

Вышесказанное иллюстрируется примером запроса информации о версии программного обеспечения (описание ключей приведено в Табл.3):

**if** (*IsGetParamFuncPtrAssigned*())  
{ // Указатель на функцию запроса параметров присвоен.  
 // Можно запросить значения из словаря свойств  
 ProgVersion = *GetParamValue(1)*;  
 ProgSubVersion = *GetParamValue(2)***;**  
 ProgCRC = *GetParamValue(6)*;

}

**else**

{  
 // Указатель на функцию запроса параметров не присвоен.  
 // Доступа к словарю свойств нет, поэтому присваиваем значения из констант

ProgVersion = PROG\_VERSION;  
ProgSubVersion = PROG\_SUBVERSION**;**  
ProgCRC = PROG\_CRC;

}

1. Библиотека написана для компилятора IAR AVR [↑](#footnote-ref-1)