

1. Вступление

Одной из множества возможностей изделий семейства MicroConverter является способность устройства загружать код в расположенную на кристалле FLASH/EE память программ. Эта встроенная возможность загрузки кода использует последовательный UART порт, и поэтому часто называется «последовательной загрузкой». Способность последовательной загрузки позволяет разработчикам перепрограммировать элемент, в то время когда оно припаяно к целевой системе, без использования внешних устройств программирования. Последовательная загрузка открывает также возможность обновления версии систем, все что нужно это последовательный порт для доступа к MicroConverter. Это означает, что производители могут обновлять аппаратно-программное обеспечение без необходимости изъятия компонент системы.

Каждое устройство MicroConverter может быть отконфигурировано для режима последовательной загрузки с помощью специального контакта при включении или при использовании внешнего сигнала сброса. Для ADuC812, контакт *PSEN* переведен на низкий уровень сигнала при помощи резистора. Если это состояние обнаруживается элементом при включении или при использовании сигнала сброса, то элемент переходит в режим последовательной загрузки. В этом режиме инициализируется резидентная программа загрузчика, встроенный загрузчик конфигурирует устройство UART и при помощи специального протокола последовательной загрузки устанавливает соединение с некоторой управляющей машиной для управления загрузкой данных в FLASH/EE память. Следует заметить, что режим последовательной загрузки работает от стандартного уровня питания (+2.7В до +5.5В). Поэтому, нет необходимости в специальном повышенном уровне напряжения, так как это происходит внутри кристалла.

Как часть программного обеспечения к средству разработки QuickStart поставляется ДОС-программа (download.exe), которая дает возможность пользователю загружать программный код через последовательный порт PC в ADuC812 MicroConverter. Однако следует заметить, что любая хост-машина (PC, микроконтроллер, DSP и др.) может загружать код в ADuC812 с помощью описываемого в этом техническом замечании протокола последовательной загрузки.

Цель этого технического замечания - обрисовать детали протокола последовательной загрузки MicroConverter, позволяя пользователям как понять протокол, так и успешно реализовывать в некоторой конечной системе.

2. Загрузчик ADuC812

Протокол последовательной загрузки на любом компоненте семейства MicroConverter реализован как встроенная подпрограмма. Встроенный загрузчик ADuC812 прошел две ревизии:

Загрузчик версии 1: на всех кристаллах с проставленным кодом < 9933 (до августа 1999)

Загрузчик версии 2: на всех кристаллах с проставленным кодом ≥ 9933 (после августа 1999)

Загрузчик версии 1 поддерживает загрузку нерегенерированного Intel Hex формата непосредственно в область FLASH/EE памяти ADuC812. Этот функциональный начальный протокол рассматривается в пункте 2.2.

Загрузчик версии 2 поддерживает более полный протокол, разрешающий последовательную загрузку в FLASH/EE память программ или в FLASH/EE память данных. Имеются и другие расширенные возможности, позволяющие более гибкое и более защищенное использование встроенной загрузки ADuC812.

Для пользователей, которые хотят поддерживать оба протокола в их конечных системах, пункт 3 описывает исходный текст программы на языке C, которая иллюстрирует как это можно сделать. Исходный текст «download.c», сопровождающий это техническое замечание, включен в своей скомпилированной форме (download.exe) как часть пакета MicroConverter QuickStart. Запущенная из командной строки ДОС, программа может установить связь с любой версией загрузчика и передавать код в FLASH/EE память программ ADuC812. Исходный текст программы может с небольшими изменениями быть скомпилированным для любой хост-машины, требуемой для загрузки ADuC812.

Для ясности в описании вводится термин «хост», обозначающий любую хост-машину, пытающуюся загрузить данные в MicroConverter. Термин «загрузчик» обозначает специальное встроенное резидентное аппаратно-программное обеспечение MicroConverter.

2.1. Загрузчик ADuC812 версии 2

2.1.1. Запуск загрузчика

Как было сказано выше, загрузчик ADuC812 запускается при пониженном через резистор уровне контакта *PSEN* (обычно 1K) и включении (переключении контакта *RESET*) элемента. При включении или переключении контакта *RESET*, загрузчик немедленно передает следующие 25 байт идентификации:

10 байт	идентификатор изделия	«ADI<space>812<space><space><space>»
4 байта	версия изделия	«V201»
2 байта	перенос строки и возврат каретки	
2 байта	конфигурация аппаратных средств	
6 байт	зарезервировано	
1 байт	контрольная сумма предыдущих 24 байт	

2.1.2. Физический интерфейс

Приведенный в действие, загрузчик конфигурирует последовательный порт ADuC812 для 8-разрядного приема/передачи на скорости 9600 бод без проверки четности. Скорость в бодах напрямую зависит от тактовой частоты ADuC812, т.е. 9600 бод устанавливаются по тактовой частоте 11.0592 МГц. Если тактовая частота увеличена или уменьшена, скорость в бодах соответственно изменяется, например, если тактовая частота равна 1 МГц, тогда загрузчик сконфигурирует UART на скорость (1 МГц/11.0592МГц) 868.055 бод. Программа download.exe, описанная в главе 3, разрешает пользователю вводить конечную тактовую частоту и, соответственно, конфигурировать скорость передачи.

2.1.3. Опрос загрузчика

Вначале, загрузчик должен быть опрошен для проверки его присутствия и вычисления номера версии. Последовательность опроса используется как пример в пункте 3 – описании программы. В этом коде хост решает какой загрузчик присутствует и, соответственно, какой протокол необходимо использовать для загрузки.

Загрузчик опрашивается (в любой момент) передачей следующего пакета данных в MicroConverter:

Пакет опроса загрузчика (в HEX-форме): <21h><5Ah><00h><A6h>

Загрузчик немедленно передает следующие 25 байт идентификации:

10 байт	идентификатор изделия	«ADI<space>812<space><space><space>»
4 байта	версия изделия	«V201»
2 байта	перенос строки возврат каретки	
2 байта	конфигурация аппаратных средств	
6 байт	зарезервировано	
1 байт	контрольная сумма предыдущих 24 бай	

2.1.4. Определение формата пакета передачи данных

Как только хост уведомляется в присутствии загрузчика, можно начать передачу данных. Основной формат пакета передачи данных представлен на рис.1.

Идентификатор начала пакета	Число байт	Data 1 тип команды	Data 2	Data 3	Data X X=25 (макс.)	Контрольная сумма
<07h> и <0Eh>	1 – 25	C, A, W, E, U	-	-	-	Число байт + Data1 ... +Data X ~(Сумма)

Рис.1. Формат пакета передачи данных

Идентификатор начала пакета

Первое поле пакета - идентификатор начала пакета, которое содержит два стартовых символа (<07h> и <0Eh>). Эти байты являются константами и используются загрузчиком для определения правильного пакета данных.

Число байт данных

Следующее поле используется для хранения общего числа байт данных. Максимальное число байт – 25, в зависимости от пакета данных, который будет передаваться.

Тип команды

Поле типа команды описывает функцию пакета данных. Возможно одно из пяти значений. Следует заметить, что некоторые команды не требуют дополнительных данных (например, Erase - удалить). Пять команд описываются ASCII символами – 'C','A','W','E','U' . Загрузчик ответит или подтверждением ACK (06h) или отрицанием NAK (07h) для переданного ранее пакета. Полный список команд приведен на рис.2.

Протокол последовательной загрузки

<i>Тип команды</i>	<i>Команда в Data 1</i>	<i>Отрицание загрузчика</i>	<i>Подтверждение загрузчика</i>
Очистить FLASH/EE память программ	'C' (43h)	NAK (07h)	ACK (06h)
Очистить FLASH/EE память программ и данных	'A' (41h)	NAK (07h)	ACK (06h)
Записать в FLASH/EE память программ	'W' (57h)	NAK (07h)	ACK (06h)
Записать в FLASH/EE память данных	'E' (45h)	NAK (07h)	ACK (06h)
Перейти на код пользователя	'U' (55h)	NAK (07h)	ACK (06h)

Рис.2. Тип команд пакета данных

Поля данных 2-25

Здесь содержатся байты данных, которые нужно загрузить. Эти данные должны быть в 16-байтном формате Intel HEX и перекодированы хостом до передачи загрузчику, как часть формата данных.

Контрольная сумма

Здесь записывается контрольная сумма. Она рассчитывается как сумма всех байтов, начиная с Data 1 до Data X, в формате дополнения до двух.

2.1.5. Использование формата пакета данных для загрузки

Программа может быть загружена из хоста в MicroConverter с использованием форматов пакета данных, описанных выше. На рис.3 отображена типичная последовательность действий.

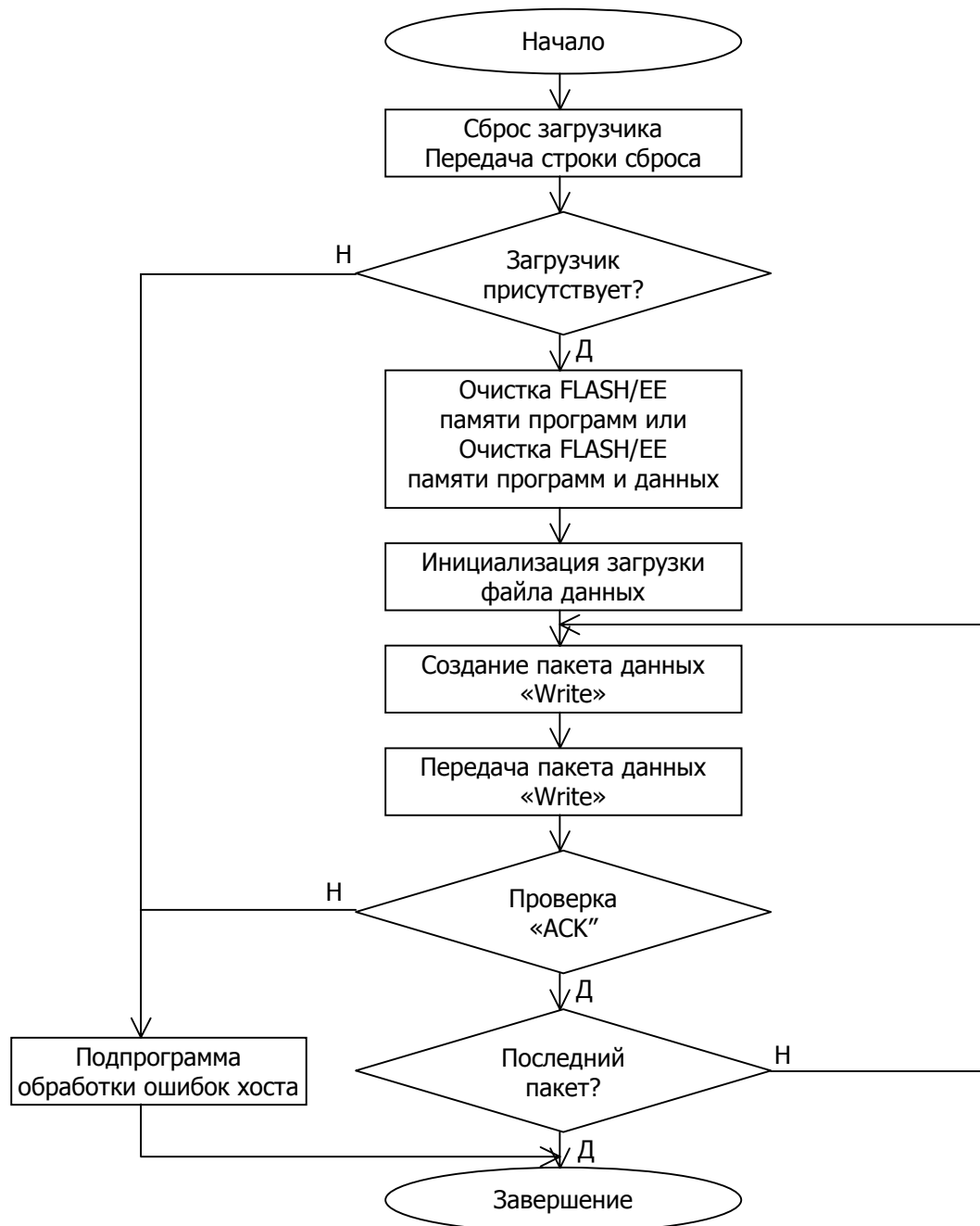


Рис.3. Блок-схема типичной последовательности загрузки

Следует отметить следующие моменты:

- Если хост загружает данные в FLASH/EE память программ или в FLASH/EE память данных, они должны быть предварительно очищены командой «С» (очистка памяти программ) или «А» (очистка памяти программ и памяти данных). Загрузчик ответит «NAK», если хост будет пытаться загрузить данные в неочищенную память.

б. Команды очистки, представленные на рис.2, не требуют дополнительной информации в формате пакета данных. Пример пакета данных, инициализирующего очистку FLASH/EE памяти программ и памяти данных, показан на рис.4.

<i>Идентификатор начала пакета</i>	<i>Число байт</i>	<i>Data 1</i>	<i>Контрольная сумма</i>
07h и 0Eh	01h	«A», 41h	BEh

Рис.4. Команда очистки FLASH/EE памяти программ и памяти данных

в. Все загружаемые пакеты данных требуют начальный адрес. Адрес содержится в трех байтах, следующих непосредственно за байтом команды. Пакеты содержат также сами данные, которые необходимо загрузить. Первый байт данных записывается загрузчиком по адресу, указанному в пакете. Далее, загрузчик увеличивает значение адреса и записывает по нему следующий байт и т.д., до тех пор пока все байты данных не будут записаны. Пример пакета данных, записывающего 8 значений в FLASH/EE память программ, начиная с адреса 0000h, приведен на рис.5.

<i>Идентификатор начала пакета</i>	<i>Число байт</i>	<i>Data 1</i>	<i>Data 2</i>	<i>Data 3</i>	<i>Data 4</i>	<i>Data 5</i>	<i>Data 6</i>	<i>Data 7</i>	<i>Data 8</i>	<i>Data 9</i>	<i>Data 10</i>	<i>Data 11</i>	<i>Data 12</i>	<i>Контрольная сумма</i>
07h и 0Eh	0Ch	«W», 57h	00h	00h	00h	00h	0Ch	0Eh	0Ch	0Fh	0Eh	4Fh	63h	BAh

Начальный адрес

Рис.5. Команда записи в FLASH/EE память программ

г. Как можно заметить в спецификации ADuC812, 640 байт FLASH/EE памяти данных должны программироваться 4-байтными страницами (640 байт сконфигурированы как 160 страниц). Пример пакета данных, записывающего в пятую страницу FLASH/EE памяти данных, приведен на рис.6.

Идентификатор начала пакета	Число байт	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7	Data 8	Контрольная сумма
07h и 0Eh	08h	«E»,45h	00h	00h	05h	0Ah	0Bh	0Ch	0Dh	80h

Начальный адрес

Рис.6. Команда записи в FLASH/EE память данных

д. Как было отмечено ранее, программа загрузчика MicroConverter передает символ «NAK» (07h) как отрицательный ответ или символ «ACK» (06h) как положительный ответ. Отрицательный ответ может быть получен в следующих случаях:

- неправильный пакет данных при проверке контрольной суммы
- загрузчик не смог проверить корректность записи данных
- загрузчик пытался записать данные в область памяти, которая не была очищена


е. Контрольная сумма в формате дополнения до двух, рассчитывается как сложение шестнадцатеричных значений в полях числа байтов и байтов данных (Data 1-25) с дополнением до двух. Она может быть рассчитана как:

$$\text{Контрольная сумма} = 100h - (\text{DataByte}_{\text{число байт}} + \sum_{N=1,25} \text{DataByte}_N)$$

2.1.6. Передача загрузчику команды запуска кода

После того, как хост передал все пакеты данных загрузчику, он может передать последний пакет, командующий загрузчику установить счетчик команд MicroConverter на указанный адрес, что вызовет исполнение загруженного кода. Рис.7 показывает пример пакета данных с командой запуска программы с адреса 0000h.

<i>Идентификатор начала пакета</i>	<i>Число байт</i>	<i>Data 1</i>	<i>Data 2</i>	<i>Data 3</i>	<i>Data 4</i>	<i>Контрольная сумма</i>
07h и 0Eh	04h	«U»,55h	00h	00h	00h	A7h



Начальный адрес

Рис.7. Команда запуска кода пользователя

2.2. Загрузчик ADuC812 версии 1

2.2.1. Запуск загрузчика

Как было сказано выше, загрузчик ADuC812 запускается при пониженном через резистор уровне контакта *PSEN* (обычно 1K) и включении (переключении контакта *RESET*) элемента. При включении или переключении контакта *RESET*, загрузчик немедленно передает следующие 11 байт идентификации:

8 байт	идентификатор изделия	«ADuC812<space>»
3 байта	версия изделия	«krl»

2.2.2. Физический интерфейс

Приведенный в действие, загрузчик конфигурирует последовательный порт ADuC812 для 8-разрядного приема/передачи на скорости 9600 бод без проверки четности. Скорость в бодах напрямую зависит от тактовой частоты ADuC812, т.е. 9600 бод устанавливаются по тактовой частоте 11.0592 МГц. Если тактовая частота увеличена или уменьшена, скорость в бодах соответственно изменяется, например, если тактовая частота равна 1 МГц, тогда загрузчик сконфигурирует UART на скорость (1 МГц/11.0592МГц) 868.055 бод. Программа download.exe, описанная в главе 3, разрешает пользователю вводить конечную тактовую частоту и, соответственно, конфигурировать скорость передачи.

2.2.3. Опрос загрузчика

Вначале, загрузчик должен быть опрошен для проверки его присутствия и вычисления номера версии. Последовательность опроса используется как пример в пункте 3 – описании программы. В этом коде хост решает какой загрузчик присутствует и, соответственно, какой протокол необходимо использовать для загрузки.

Загрузчик опрашивается (в любой момент) передачей следующего символа в MicroConverter:

Символ опроса загрузчика (в HEX-форме): <21h>

Загрузчик немедленно передает следующие 11 байт идентификации:

8 байт	идентификатор изделия	«ADuC812<space>»
3 байта	версия изделия	«krl»

2.2.4. Передача кода загрузчику версии 1

В отличие от загрузчика версии 2, рассмотренного ранее, загрузчик версии 1 поддерживает прямую загрузку только в FLASH/EE память программ. Хост должен передать неизменяемый стандартный файл формата Intel Hex. Загрузчик воспринимает эту передачу как часть последовательной загрузки.

Загрузчик версии 1 автоматически очищает FLASH/EE память программ и память данных, как только загрузчик активизируется, перед передачей 11 байт идентификационной строки (см. пункт 2.2.1). Следует заметить, что загрузчик не поддерживает передачу в FLASH/EE память данных.

Как только загрузчик был опрошен (см. пункт 2.2.3), он ждет передачи стандартного файла формата Intel Hex, который будет записываться в FLASH/EE память программ. Загрузчик принимает файл на основе записей, требуя от хоста придерживаться протокола передачи. Формат стандартного Intel Hex файла описан в пункте 2.2.6.

- загрузчик определяет начало записи по символу «:». Важно заметить, что загрузчик будет использовать различное число байт, которые будут определять запись, такие как адрес, число байт в записи, контрольная сумма и окончание записи. Поэтому, хост должен передавать Intel Hex файл «как есть».
- в конце каждой записи, загрузчик будет передавать или символ ACK (положительный ответ), или NACK (отрицательный ответ). ACK является кодом 06h, NACK является кодом 15h, т.е. они придерживаются промышленного стандарта UART. Пользовательский хост должен перехватывать эти символы и, соответственно, продолжать передачу при получении положительного ответа и сообщать об ошибке при получении отрицательного ответа. При ошибке загрузчик ждет начала новой записи, поэтому пользователь может или остановить загрузку, или попробовать передать запись повторно.

2.2.5. Запуск кода

При достижении конца файла, пользователь может заставить загрузчик начать выполнение кода, передав ему командную строку с указанием начального адреса. Строка состоит из первого символа «;» и последующих четырех символов адреса. Важно заметить, что в настоящих программах, если хост не хочет передавать адрес, тогда он должен быть равным FF00h (этот начальный адрес указывает на резидентную программу включения, которая вызывает калибровку АЦП, внутреннего опорного напряжения сигнала и затем переходит на адрес 0000h к пользовательскому коду).

Если вы не заканчиваете передачу последовательностью стартового адреса, единственный путь для запуска кода – это убрать понижение напряжения *PSEN* и вызвать сброс или новый энергетический цикл.

2.2.6. Формат стандартного Intel Hex файла

Эта часть описывает формат стандартного Intel Hex файла, используемого протоколом загрузки версии 1. Шестнадцатеричный формат Intel или формат Intel Hex является стандартом для запоминающих машин в отображаемом и печатаемом формате.

Стандартный Intel Hex формат генерируется ассемблером MicroConverter (8051-совместимый код). Этот ассемблер (Metalink 2-pass) доступен как часть средств разработки для QuickStart или как свободно распространяемая копия, доступная на веб-сайте www.analog.com/microconverter.

Intel Hex файл это последовательность строк или «шестнадцатеричных записей», содержащих следующие поля:

Поле	Число байт	Описание
Начало записи	1	«:», характеризует начало записи
Длина записи	2	число байт в записи
Адрес загрузки	4	начальный адрес для размещения байтов данных
Тип записи	2	00 = данные, 01 = завершение
Байты данных	0 – 16	данные
Контрольная сумма	2	сумма байтов данных в записи + контрольная сумма = 0

Рис.8. Формат записи Intel Hex

Эти поля показаны в раскрытом виде на рис.9, который иллюстрирует пример содержимого стандартного Intel Hex файла.

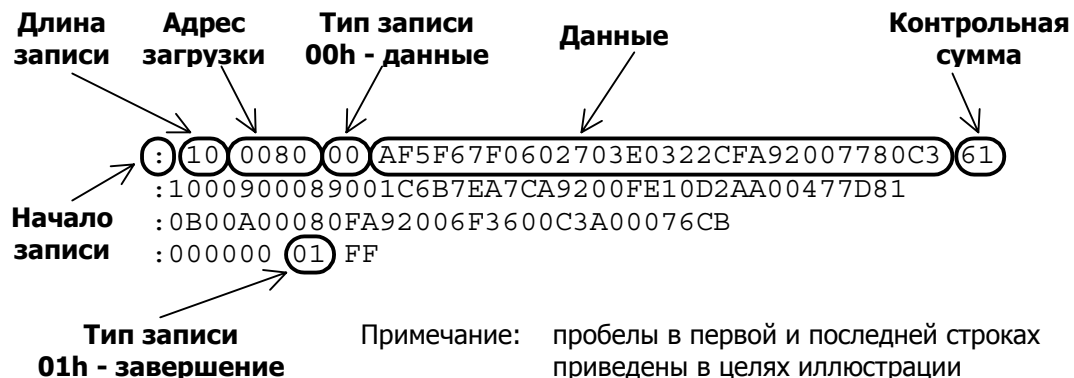


Рис.9. Формат Intel Hex

3. Исходный текст программы на языке C, поддерживающей оба протокола

Эта часть описывает общие положения исходного текста программы на языке C, реализующей загрузку данных, и доступной на веб-сайте MicroConverter («download.c»). Хотя программа загрузки может быть откомпилирована для ПК под управлением ДОС, она с небольшими изменениями может быть откомпилирована для любой конечной системы, которая поддерживает встроенный загрузчик.

Программа совместима с любыми версиями загрузчика и предназначена для иллюстрации примера связи хост-машины со встроенным аппаратно-программным обеспечением загрузки. В программу встроены процедуры, которые считывают содержимое стандартного Intel Hex файла (readBlockFromFile, readRecordFromFile), и процедуры, которые формируют правильно отформатированные пакеты данных для передачи загрузчику (sendLoaderPacket).

Программа содержит также некоторые дополнительные, удобные пользователю возможности, которые могут быть включены из командной строки во время выполнения.

3.1. Параметры командной строки

DOWNLOAD filename.hex /c:n /f:n.n /d /r ,где

«filename.hex»	имя Intel Hex файла для загрузки. Обязательный параметр
/C:n	выбрать COM-порт. По умолчанию COM1 (1)
/F:n.n	выбрать частоту кристалла (в МГц). По умолчанию 11.0592
/D	не стирать FLASH/EE память данных (начиная с версии 2.0)
/R	запустить программу с адреса FF00h
/R:xxxx	запустить программу с указанного адреса

Некоторые примеры:

DOWNLOAD test1.hex /c:1 /f:16 /d /r

Загрузить программу test1.hex через порт COM1 на конечную систему, где MicroConverter работает на тактовой частоте 16 МГц. FLASH/EE память программ не будет стерта перед загрузкой (FLASH/EE память программ будет стерта по умолчанию). Программа автоматически запускается с адреса FF00h как только процесс загрузки завершается.

DOWNLOAD test2.hex /c:2

Загрузить программу test2.hex через порт COM2 на конечную систему, где MicroConverter работает на тактовой частоте 11.0592 МГц. FLASH/EE память программ будет стерта перед загрузкой (FLASH/EE память программ будет стерта по умолчанию). Программа не запускается после процесса загрузки. Для инициализации запуска необходимо убрать низкое напряжение на контакте *PSEN* и начать новый энергетический цикл.

DOWNLOAD test3.hex /c:1 /r:0F00

Загрузить программу test3.hex через порт COM1 на конечную систему, где MicroConverter работает на тактовой частоте 11.0592 МГц. FLASH/EE память программ будет стерта перед загрузкой (FLASH/EE память программ будет стерта по умолчанию). Программа автоматически запускается с адреса 0F00h как только процесс загрузки завершается. (Пользователь может использовать запуск с указанного адреса во время разработки для избежания повтора выполнения некоторых системных подпрограмм, которые задерживают время отладки).

3.2. Процесс выполнения программы

Процесс выполнения программы download.c может быть рассмотрен как

Последовательность сброса

Примечание: загрузчик версии 1 автоматически стирает FLASH/EE память программ и память данных перед передачей идентификационной строки. Загрузчик версии 2 будет ждать приема специального командного байта очистки FLASH/EE памяти программ отдельно или FLASH/EE памяти программ и памяти данных.

Загрузчик версии 1 использует символ «!» для сброса, а система отвечает последовательностью: «ADuC812 krl»

Загрузчик версии 2 использует четырех байтовую последовательность сброса: «!Z» <0x00> <контрольная сумма>. Система отвечает 25 байтами идентификационной строки, начинающейся с «ADI» и заканчивающейся байтом контрольной суммы.

Для поддержки всех версий, программа должна передать загрузчику символ «!» и подождать ответа. Если его не последует, то послать символы «Z» <0x00> <контрольная сумма>. Это позволит идентифицировать присутствие и версию загрузчика, какой протокол необходимо использовать для последовательной загрузки.

Последовательность загрузки

Старая версия загрузчика может декодировать записи Intel Hex самостоятельно. Пакеты считываются из файла и посылаются системе «как есть». Система отвечает на каждый пакет ответами ACK и NACK для проверки состояния загрузки. Новая версия загрузчика требует, чтобы данные были закодированы в форме «записи в память», которая посылается загрузчику как пакеты команд.

Последовательность запуска

В старой версии загрузчика команда запуска выглядела строкой «;xxxx», где xxxx – начальный адрес. Система отвечает символом ACK, если запуск был успешен. Новый загрузчик использует передачу пакета команды запуска.

Блок-схема на рис.10 иллюстрирует детализированный процесс выполнения программы.

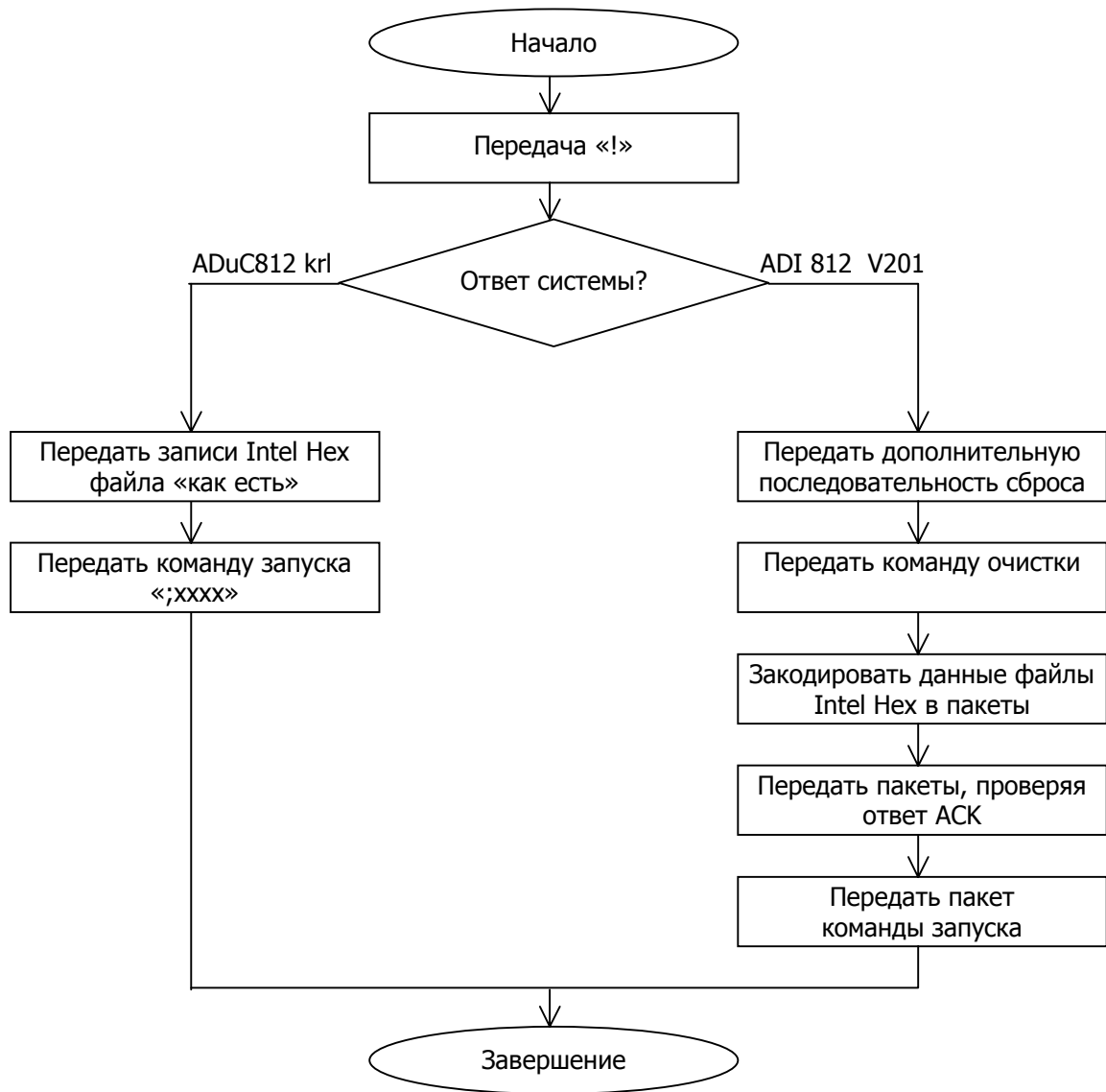


Рис.10. Последовательность выполнения программы