MicroConverter™ Техническое замечание - µC004
Протокол последовательной загрузки

1.0 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 MicroConverter, 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 программа WSD.exe (работающая под управлением Windows), которая дает возможность пользователю загружать программный код через последовательный порт PC в MicroConverter. Однако следует заметить, что любая хост-машина (PC, микроконтроллер, DSP и др.) может загружать код в MicroConverter 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.

Оригинал : версия 2.0 [12/2000]
Перевод : версия 2.0 [08/2001]

Техническое замечание µ0004

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

Загрузчик версии 1: на всех кристаллах ADuC812 с кодом < 9933 (до августа 1999)

Загрузчик версии 2: на всех кристаллах ADuC812 с кодом ≥ 9933 (после августа 1999)
на всех кристаллах ADuC816
на всех кристаллах ADuC824

Загрузчик версии 1 поддерживает загрузку нерегенерированного Intel Hex формата непосредственно в область FLASH/EE памяти (только ADuC812). Этот протокол рассматривается в разделе 3.0.

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 MicroConverter.

Для пользователей, которые хотят поддерживать оба протокола в их конечных системах, раздел 4.0 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 MicroConverter. Исходный текст программы может с небольшими изменениями быть скомпилированным для любой хост-машины, требуемой для загрузки.

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.

2.0. ЗАГРУЗЧИК MICROCONVERTER ВЕРСИИ 2

ПРИМЕЧАНИЕ:
Загрузчик версии 2 применяется в следующих MicroConverter:

Загрузчик версии 2: на всех кристаллах ADuC812 с кодом ≥ 9933 (после августа 1999)
на всех кристаллах ADuC816
на всех кристаллах ADuC824

2.1. ЗАПУСК ЗАГРУЗЧИКА

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

«8xx» - является ID изделия
напр. 812 для ADuC812
824 для ADuC824

10 байт	идентификатор изделия	**«ADI<space>8xx<space><space><space>»**
4 байта	версия изделия	**«V2xx»**
2 байта	перенос строки и возврат каретки	
2 байта	конфигурация аппаратных средств	
6 байт	зарезервировано	
1 байт	дополнительный код контрольной сумма предыдущих 24 байт	

«V2xx» - является ID версии загрузчика. «xx» - номер ревизии

2.2. ФИЗИЧЕСКИЙ ИНТЕРФЕЙС

Приведенный в действие, загрузчик конфигурирует последовательный порт MicroConverter для 8-разрядного приема/передачи на скорости 9600 бод без проверки четности.

ПРИМЕЧАНИЕ:
Для **ADuC812** скорость передачи косвенно зависит от тактовой частоты, т.е. скорость 9600 бод основана на тактовой частоте 11.0592 МГц. Если тактовая частота увеличена или уменьшена, то скорость передачи также изменяется.

Скорость_В_Бодах = 9600 бод x Частота_Кристалла / 11.0592 МГц

Например, если тактовая частота составляет 1 МГц, то загрузчик сконфигурирует UART на скорость 868.055 бод. Программа WSD.exe, описанная в разделе 5, разрешает пользователю вводить конечную тактовую частоту и, соответственно, конфигурировать скорость передачи.

Для **ADuC816/ADuC824** скорость передачи настраивается на 9600 бод от частоты ядра 12.583 МГц. Если присутствует 32.768 МГц кварц, то ФАПЧ автоматически синхронизируется на эту частоту. Если он отсутствует, то система ФАПЧ не может гарантировать работу на этой частоте. Программа WSD.exe позволяет изменить скорость передачи, чтобы соответствовать генерируемой тактовой частоте. Если частота синхронизации ФАПЧ измерена (с использованием ALE) без кварца, то настоящая скорость передачи составит:

Скорость_В_Бодах = 9600 бод x Частота_ФАПЧ / 12.583 МГц

2.3. ОПРОС ЗАГРУЗЧИКА

Вначале, загрузчик должен быть опрошен для проверки его присутствия и вычисления номера версии. Последовательность опроса используется в примере программы «download.c» в разделе 4.0. В этом коде хост решает какой загрузчик присутствует и, соответственно, какой протокол необходимо использовать для загрузки.

Загрузчик версии 2 может быть опрошен в любой момент передачей следующего пакета данных:

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

Загрузчик немедленно передает следующие 25 байт идентификации:

10 байт	идентификатор изделия	**«ADI¤812¤¤¤»**
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 байт	

2.4. 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.

Таблица 1. Формат пакета передачи данных.

Идентификатор начала пакета	Число байт	Data 1 тип команды	Data *x* *x* = 2->25	Контрольная сумма
<07h> и <0Eh>	1 – 25	«C», «A», «W», «E», «S», «U»	???	Число байт + Data1 … +Data X ~(Сумма)

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 составляет 25, в зависимости от пакета данных, который будет передаваться.

Тип команды
Поле типа команды описывает функцию пакета данных. Возможно одно из шести значений. Следует заметить, что некоторые команды не требуют дополнительных данных (например, Erase - удалить). Шесть команд описываются ASCII символами – «C», «A», «W», «E», «S», «U».

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.

Таблица 2. Тип команд пакета данных.

Тип команды	Команда в Data 1	Отрицание загрузчика	Подтверждение загрузчика
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)
Установить режимы **защиты** (только ADuC816/ADuC824)	«S» (53h)	NAK (07h)	ACK (06h)
Перейти на код (удаленный **запуск**) пользователя	«U» (55h)	NAK (07h)	ACK (06h)

2.5. КОМАНДНЫЕ ФУНКЦИИ ПАКЕТА ДАННЫХ

2.5.1. Команда ERASE

Как показано выше в таблице 2, имеется две команды ERASE. Одна команда очищает и память программ FLASH/EE, и память данных («A»), а другая только память программ («C»).

Если хост-машина хочет загрузить в память FLASH/EE новые данные, то предварительно память должна быть очищена с помощью одной из этих команд. При попытке записать в неочищенную память загрузчик ответит сигналом «NAK».

Команды ERASE не требуют дополнительной информации в пакете данных. Пример пакета данных, инициализирующий очистку памяти программ и памяти данных, приведен в таблице 3(а). Пример пакета данных, инициализирующий только очистку памяти программ, приведен в таблице 3(б).

Таблица 3а. Команда очистки FLASH/EE памяти программ и данных пакета данных.

Стартовый ID команды	Число байт данных	Data 1 (команда)	Контрольная сумма
07h и 0Eh	1	«A» (41h)	**BEh**

Таблица 3б. Команда очистки только FLASH/EE памяти программ.

Стартовый ID команды	Число байт данных	Data 1 (команда)	Контрольная сумма
07h и 0Eh	1	«C» (43h)	**BCh**

2.5.2. Программирование блока FLASH/EE памяти программ

Все пакеты данных загрузки требуют начальный адрес. Он содержится в трех байтах, следующих непосредственно за байтом команды. Пакеты данных загрузки содержат также записываемые данные. Первый байт данных записывается загрузчиком по адресу, указанному в пакете данных, после чего загрузчик увеличивает значение адреса и записывает следующий байт данных, до тех пор пока не будет записано последнее значение из пакета данных. Пример пакета данных, который программирует восемь ячеек FLASH/EE памяти программ, начиная с адреса 0000h, приведен в таблице 4.

Таблица 4. Команда программирования FLASH/EE памяти программ.

Стартовый ID команды	Число байт	Data 1 CMD	Data 2 ADR U	Data 3 ADR M	Data 4 ADR L	Data 5 Prog 1	Data 6 Prog 2	Data 7 Prog 3	Data 8 Prog 4	Data 9 Prog 5	Data 10 Prog 6	Data 11 Prog 7	Data 12 Prog 8	Контрол. сумма
07h и 0Eh	0Ch (12)	«W» (57h)	00h	00h	00h	0Ch	0Eh	0Ch	0Eh	0Fh	0Eh	0Fh	63h	**BAh**

2.5.3. Программирование блока FLASH/EE памяти данных

Как показано в спецификациях MicroConverter, 640 байт FLASH/EE памяти данных должны программироваться 4-байтными страницами (640 байт сконфигурированы как 160 страниц). Пример пакета данных, который программирует пятую ячейку FLASH/EE памяти данных, приведен в таблице 5.

Таблица 5. Команда программирования FLASH/EE памяти данных.

Стартовый ID команды	Число байт	Data 1 CMD	Data 2 ADR U	Data 3 ADR M	Data 4 ADR L	Data 5 Prog 1	Data 6 Prog 2	Data 7 Prog 3	Data 8 Prog 4	Контрол. сумма
07h и 0Eh	08h	«E» (45h)	00h	00h	05h	0Ah	0Bh	0Ch	0Dh	**80h**

2.5.4. Команда установки режима защиты

Существует три режима защиты для ADuC816/ADuC824. Для уточнения особенностей каждого из режимов защиты обратитесь к спецификациям изделий (datasheets). После команды ERASE микросхема возвращается в состояние NO SECURITY («нет защиты»). Поэтому после каждой загрузки необходимо установить режим защиты заново. Для этого после командного байта в пакете данных («C») посылается соответствующий байт режима защиты (см. таблицу 6).

Таблица 6. Команда программирования FLASH/EE памяти данных.

Режим защиты	Data 2
режим LOCK	06h
режим SECURE	05h
режим SECURE и LOCK	04h
режим SERIAL и SAFE	03h
режим SERIAL SAFE и LOCK	02h
режим SERIAL SAFE и SECURE	01h
режим SERIAL SAFE, SECURE и LOCK	00h

ПРИМЕЧАНИЕ:
Режим SERIAL SAFE отключает последовательный загрузчик, поэтому запись этого бита предотвращает последующую загрузку через последовательный порт. Единственный способ отключить этот режим – инициализировать очистку памяти программ в режиме параллельной загрузки.

Режим SECURE содержит все методы защиты режима LOCK, поэтому режим SECURE и LOCK идентичен режиму SECURE.

Таблица 7. Команда установки режима защиты.

Стартовый ID команды	Число байт данных	Data 1 (команда)	Data 2	Контрольная сумма
07h и 0Eh	02h	«S» (53h)	05h	**A6h**

2.5.5. Команда перехода на код пользователя (удаленный запуск)

После того как хост передаст загрузчику все пакеты данных, он может послать заключительный пакет, указывающий загрузчику перегрузить счетчик команд значением определенного адреса и начать с него выполнением только что загруженного кода. Таблица 8 показывает пример команды перехода на код пользователя с адреса 000000h.

Таблица 8. Команда перехода на код пользователя.

Стартовый ID команды	Число байт данных	Data 1 CMD	Data 2 ADR U	Data 3 ADR M	Data 4 ADR L	Контрольная сумма
07h и 0Eh	04h	«U» (55h)	00h	00h	00h	**A7h**

ОЗУ после удаленного запуска
ADuC812 имеет функцию очистки внутреннего ОЗУ, вызываемую во время программы конфигурации по включению. Поэтому после сброса (для запуска кода) внутреннее ОЗУ во всех ячейках памяти будет содержать значение 00h. Если используется команда удаленного запуска, то очистки внутреннего ОЗУ не происходит, т.е. состояние памяти будет таким же, каким оно было до процесса последовательной загрузки, за исключением следующих ячеек, которые искажаются загрузчиком:

00h -> 02h, 05h -> 0Dh, 2Ah и 33h -> 47h

Сброс ADuC816 и ADuC824 для запуска кода пользователя сохраняет состояние внутреннего ОЗУ. Однако некоторые байты памяти будут искажены загрузчиком версии 2. Адреса этих байтов:

01h, 05h, 08h -> 0Dh и 2Fh

Использование команды удаленного запуска после последовательной загрузки вызовет в ADuC816 и ADuC824 искажение байтов памяти по следующим адресам:

00h -> 01h, 05h -> 0Dh и 2Fh -> 44h

SFR после удаленного запуска

Сброс MicroConverter (ADuC812/ADuC816/ADuC824) для запуска кода пользователя загружает все регистры специального назначения их значениями по умолчанию. Однако использование удаленного запуска после последовательной загрузки в MicroConverter оставляет скорость обмена по UART равной 9600 бод. Таблицы 9(а) и 9(б) показывают SFR, которые не загружаются значениями по умолчанию.

Таблица 9а. Значения SFR после удаленного запуска и сброса для ADuC812.

SFR	Значение (после удаленного запуска)	Значение по умолчанию (после сброса)
TH1	FDh	00h
TL1	xxh	00h
SCON	56h	00h
TCON	C0h	00h
TMOD	20h	00h
SBUF	xxh	00h

Таблица 9б. Значения SFR после удаленного запуска и сброса для ADuC816/ ADuC824.

SFR	Значение (после удаленного запуска)	Значение по умолчанию (после сброса)
RCAP2H	FFh	00h
RCAP2L	D7h	00h
TH2	FFh	00h
TL2	xxh	00h
SCON	56h	00h
T2CON	34h	00h
SBUF	xxh	00h

2.5.6. Обобщенная последовательность загрузки

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

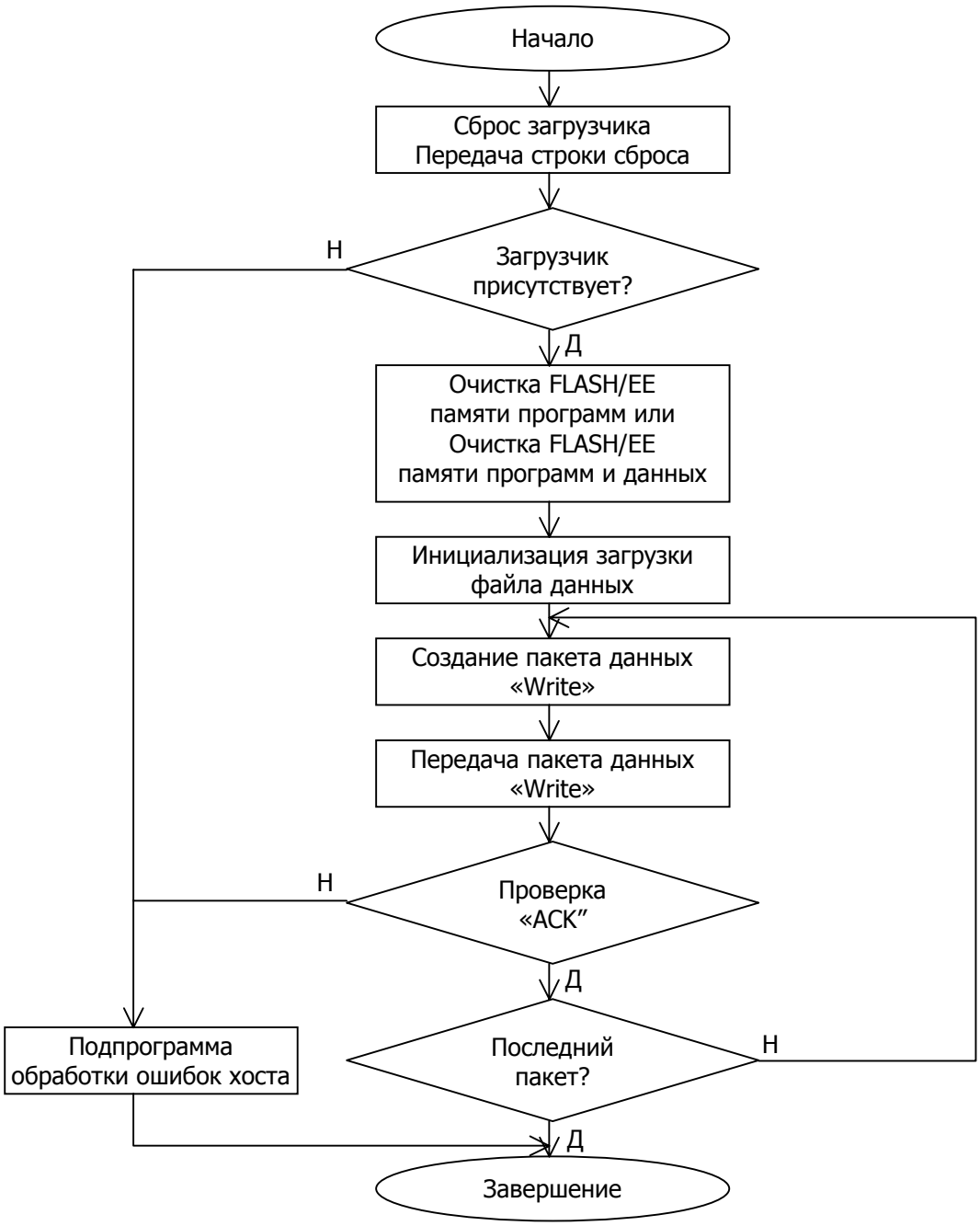

Рис.2. Блок-схема типичной последовательности загрузки.

3.0. ЗАГРУЗЧИК MICROCONVERTER ВЕРСИИ 1

ПРИМЕЧАНИЕ:
Загрузчик версии 1 присутствует на некоторых кристаллах ADuC812.

Загрузчик версии 1: на всех кристаллах ADuC812 с кодом < 9933 (до августа 1999)

3.1. ЗАПУСК ЗАГРУЗЧИКА

Как и загрузчик версии 2, загрузчик версии 1 запускается при пониженном через резистор уровне контакта *PSEN* (обычно 1К) 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»**

3.2. ФИЗИЧЕСКИЙ ИНТЕРФЕЙС

Приведенный в действие, загрузчик конфигурирует последовательный порт ADuC812 для 8-разрядного приема/передачи на скорости 9600 бод без проверки четности.

ПРИМЕЧАНИЕ:
Для **ADuC812** скорость передачи косвенно зависит от тактовой частоты, т.е. скорость 9600 бод основана на тактовой частоте 11.0592 МГц. Если тактовая частота увеличена или уменьшена, то скорость передачи также изменяется.

Скорость_В_Бодах = 9600 бод x Частота_Кристалла / 11.0592 МГц

Например, если тактовая частота составляет 1 МГц, то загрузчик сконфигурирует UART на скорость 868.055 бод. Программа WSD.exe, описанная в разделе 5, разрешает пользователю вводить конечную тактовую частоту и, соответственно, конфигурировать скорость передачи.

3.3. ОПРОС ЗАГРУЗЧИКА

Вначале, загрузчик должен быть опрошен для проверки его присутствия и вычисления номера версии. Последовательность опроса используется в примере программы «download.c» в разделе 4.0. В этом коде хост решает какой загрузчик присутствует и, соответственно, какой протокол необходимо использовать для загрузки.

Загрузчик версии 1 может быть опрошен в любой момент передачей следующего пакета данных:

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»**

3.4. ПЕРЕДАЧА КОДА ЗАГРУЗЧИКУ ВЕРСИИ 1

В отличие от загрузчика версии 2, рассмотренного ранее, загрузчик версии 1 поддерживает прямую загрузку 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. раздел 3.1). Следует заметить, что загрузчик не поддерживает передачу в FLASH/EE память данных.

Как только загрузчик был опрошен (см. раздел 3.3), 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 3.6.

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.

3.5. 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 снять понижение напряжения на выводе *PSEN* и вызвать сброс или новый энергетический цикл.

3.6. 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 форматом.

Стандартный Intel Hex формат генерируется ассемблером MicroConvertor (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:

Таблица 10. Поля стандартного Intel Hex файла.

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.9 иллюстрирует пример содержимого стандартного Intel Hex файла.

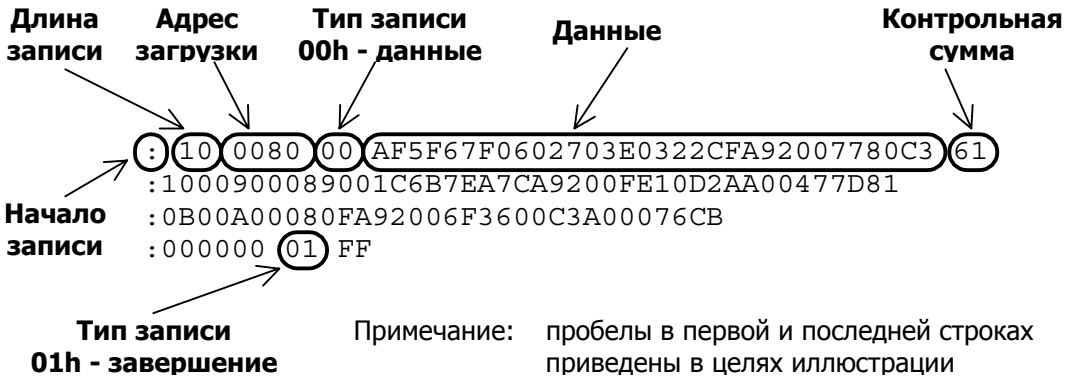

Рис.3. Формат Intel Hex

4.0. ИСХОДНЫЙ ТЕКСТ ПРОГРАММЫ НА ЯЗЫКЕ С, ПОДДЕРЖИВАЮЩЕЙ ОБА ПРОТОКОЛА

Этот раздел описывает исходный текст программы на языке С, реализующей загрузку данных. Программа доступна 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.

4.1. 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 загрузчика, программа должна передать символ «**!**» и подождать ответа. Если его не последует, то послать символы **«Z»<0x00><контрольная сумма>**. Это позволит идентифицировать присутствие и версию загрузчика, а также 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.4 иллюстрирует детализированный процесс выполнения программы.

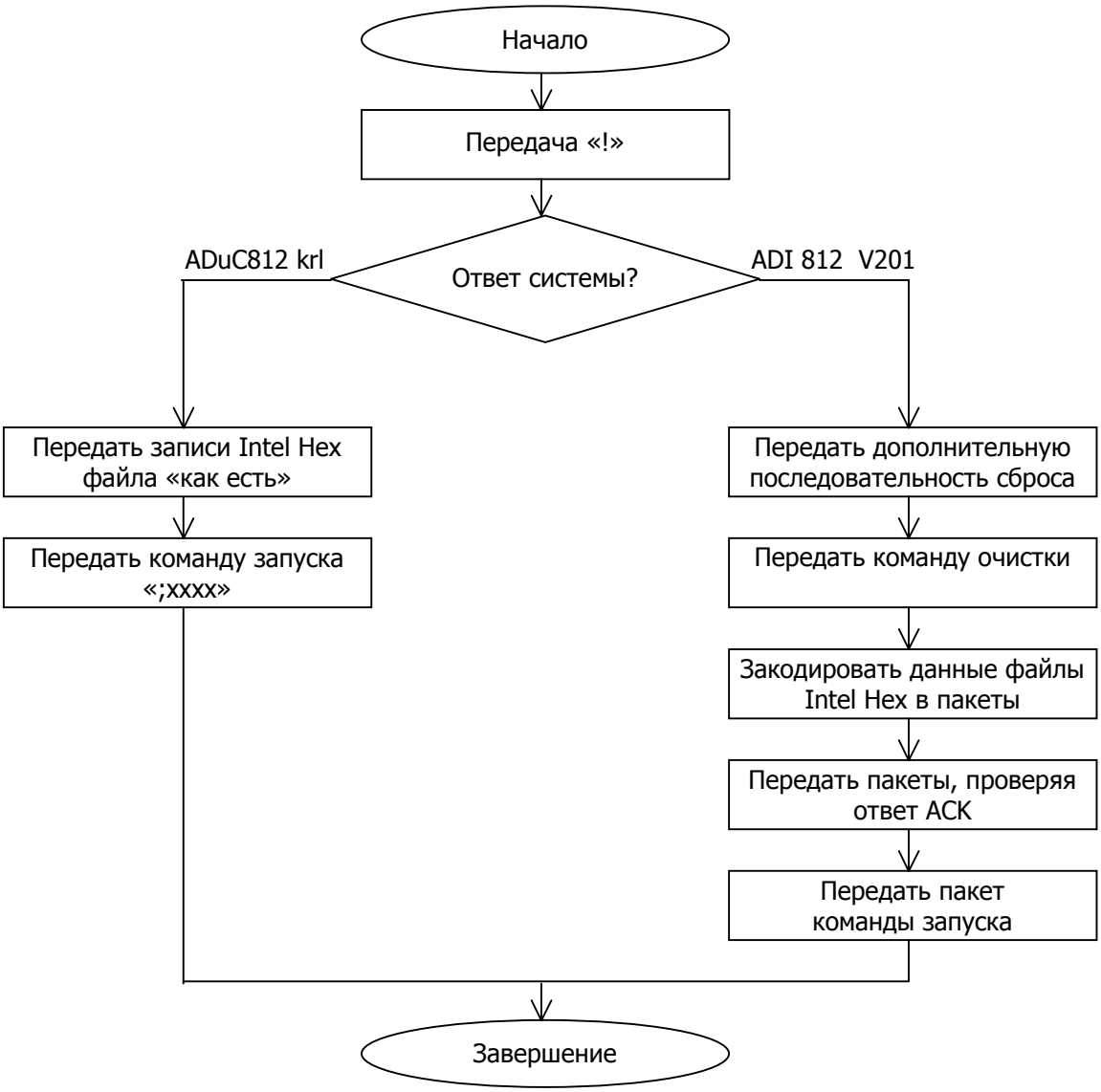

Рис.4. Последовательность выполнения программы.

5.0. ПОСЛЕДОВАТЕЛЬНЫЙ ЗАГРУЗЧИК ПОД УПРАВЛЕНИЕМ WINDOWS (WSD.EXE)

Последовательный загрузчик под управлением Windows (Windows Serial Downloader - WSD) предоставляет пользователю возможность загружать стандартные Intel Hex файлы, созданные ассемблером ASM51. WSD работает с загрузчиками версий 1 и 2. Файл загружается во встроенную FLASH/EE память через любой последовательный порт (COM1--COM4) на ПК.

Окно конфигурации дает возможность настроить параметры ЗАГРУЗКИ и ЗАПУСКА. Загрузка состоит из очистки, программирования и установки режима защиты (если необходимо). После того как будет нажата кнопка DOWNLOAD, параметры загрузки будут переданы в MicroConverter.

Доступны следующие настраиваемые параметры:

ERASE:

Пользователь может выбрать:
(а) очистить FLASH/EE память программ, или
(б) очистить и FLASH/EE память программ, и память данных.

PROGRAM:

Пользователь может выбрать:
(а) программировать FLASH/EE память программ
(б) программировать FLASH/EE память данных
(в) программировать FLASH/EE память программ и память данных

ПРИМЕЧАНИЕ:
В случае (а) пользователь может выбрать предварительную очистку FLASH/EE памяти программ, или и памяти программ, и памяти данных. В случае (б) или (в) пользователь должен предварительно очистить и память программ, и память данных.

SECURITY:

Для ADuC816/ADuC824 доступны три режима защиты. Пользователь может выбрать любую комбинацию из этих режимов.

RUN:

Пользователь может выбрать:
(а) запуск с начала
(б) запуск с определенного кода
(в) автоматический запуск после загрузки

ПРИМЕЧАНИЕ:
В случае (а) и (б) код не будет запущен до тех пор, пока пользователь не нажмет кнопку RUN. В случае (с) команда запуска автоматически посылается в MicroConverter после окончания загрузки.