ДИСТРИБУТИВ ОС GNU/LINUX НА БАЗЕ BUILDROOT ДЛЯ 1892ВМ14Я. РУКОВОДСТВО СИСТЕМНОГО ПРОГРАММИСТА

Версия v2.0.1-14

11.08.2016

ОГЛАВЛЕНИЕ

1. О ДОКУМЕНТЕ

Документ описывает дистрибутив операционной системы GNU/Linux на базе Buildroot для микросхемы 1892ВМ14Я (MCom-02), процедуру сборки и прошивки образа SD-карты с операционной системой и настройки операционной системы (далее - “ОС”).

Документ описывает дистрибутив версии v2.0.

2. ОБЩИЕ СВЕДЕНИЯ О ДИСТРИБУТИВЕ ОС

Дистрибутив ОС GNU/Linux предназначен для распространения исходных кодов ОС GNU/Linux, инструментального ПО и скриптов сборки ОС GNU/Linux.

Дистрибутив ОС поддерживает следующие отладочные модули на базе микросхемы 1892ВМ14Я:

- Салют-ЭЛ24Д1 r1.3;
- Салют-ЭЛ24Д1 r1.4;
- Салют-ЭЛ24Д2 r1.1;

Дистрибутив ОС распространяется в виде архива исходных кодов (подробнее см. *“Состав дистрибутива ОС”*).

Результатом сборки исходных кодов дистрибутива являются:

- Образ SD-карты, содержащий ОС GNU/Linux (подробнее см. *“Состав образа SD-карты”*). Образ SD-карты является унифицированным и совместим со всеми поддерживаемыми модулями.
- Образ SPI флеш-памяти, содержащий загрузчик U-Boot для соответствующего модуля (подробнее см. “Загрузчик U-Boot для 1892ВМ14Я. Руководство пользователя”).

Дистрибутив ОС построен на базе Buildroot версии 2015.08.1[1].

В состав ОС входит набор демо-тестов для проверки работы блоков и интерфейсов СнК в составе отладочного модуля.

ОС построена на базе ядра Linux версии 4.1.27[2].

В качестве загрузчика ОС используется U-Boot v2016.03.0.1 (подробнее см. “Загрузчик U-Boot для 1892ВМ14Я. Руководство пользователя”).

[1] https://git.buildroot.net/buildroot/commit/?h=2015.08.1&id=e009e7d8615eec10d2c0c2676ef5a276f0a6a5e2

[2] https://git.kernel.org/cgit/linux/kernel/git/stable/linux-stable.git/commit/?id=95123c0b81d9478b8155fe15093b88f57ef7d0bd

3. СОСТАВ ДИСТРИБУТИВА ОС

Состав архива дистрибутива представлен в Таблица 3.1.

Таблица 3.1. Состав архива дистрибутива

Директория/файл	Описание
`build.sh`	Скрипт для сборки ОС, разметки и создания образа SD-карты
`baremetal-src/tests`	baremetal тесты для функционального контроля плат
`buildroot-script`	Рецепты и скрипты для сборки корневой файловой системы на базе Buildroot
`u-boot`	Исходные коды загрузчика U-Boot
`tools`	Дополнительные утилиты
`tools/flash-spi.py`	Утилита для прошивки SPI флеш-памяти v2.0, управляет монитором BootROM СнК по интерфейсу UART
`toolchain-arm-cs-bare-2013.11`	Пакет программ от CodeSourcery, необходимых для компиляции и генерации выполняемого кода из исходных текстов baremetal-приложений на ARM

4. СОСТАВ ОБРАЗА SD-КАРТЫ

Схема разбиения образа SD-карты представлена в Таблица 4.1.

Таблица 4.1. Схема разбиения образа SD-карты на области

Область	Начало (байт)	Размер (байт)	Примечание
MBR	0	512	
Раздел boot	1 МиБ	128 МиБ	Раздел с файловой системой FAT32
Раздел root	129 МиБ	1 ГиБ	Раздел с файловой системой EXT4 с корневой файловой системой rootfs

Состав раздела boot:

- `zImage` — скомпилированное ядро Linux
- `test-ddr.bin` — параметризируемый baremetal тест для проверки DDR-памяти;
- `test-nand.bin` — baremetal тест для проверки NAND-памяти;
- `test-bist.bin` — параметризируемый baremetal тест для проверки памятей посредством BIST;
- `test-mfbsp-gpio.bin` — параметризируемый baremetal тест для проверки MFBSP GPIO в режиме loopback;
- `u-boot.env` — переменные окружения загрузчика U-Boot.

5. СБОРКА ОБРАЗА SD-КАРТЫ И ОБРАЗА SPI ФЛЕШ-ПАМЯТИ

Сборка образа SD-карты и образа SPI флеш-памяти выполняется на ПЭВМ. ПЭВМ должна удовлетворять требованиям:

1. не менее 4 ГиБ ОЗУ, 8 ГиБ свободного места на НЖМД или твердотельном накопителе;
2. на ПЭВМ должен быть установлен кард-ридер для подключения SD-карт;
3. на ПЭВМ должен быть установлен дистрибутив GNU/Linux CentOS 7.2 архитектуры x86-64;
4. на ПЭВМ должен быть настроен доступ в интернет. Если доступ в интернет осуществляется через прокси-сервер, то должно быть установлены переменные окружения `http_proxy`, `https_proxy`, `ftp_proxy`;
5. на ПЭВМ должны быть установлены следующие приложения (пути до исполняемых файлов должны быть прописаны в переменной окружения `PATH`):
   - bash версии 4.1.2;
   - cmake версии 2.8 или выше;
   - GNU make версии 3.81;
   - parted версии 2.1;
   - texinfo версии 4.13;
   - glibc.i686.

Для сборки образов необходимо:

1. разархивировать архив для сборки образа SD-карты (<package-name> - имя архива tar.bz2, но без расширения tar.bz2):

   ```
   tar xf <package-name>.tar.bz2
   ```

2. перейти в распакованную директорию:

   ```
   cd <package-name>
   ```

3. выполнить команду по запуску сборки Linux, загрузчика и приложений:

   ```
   ./build.sh build
   ```

   Длительность сборки составляет около 45 минут и зависит от производительности CPU ПЭВМ.

4. выполнить команду для подготовки образа SD-карты:

```
./build.sh mk_image
```

При запуске появится запрос пароля для sudo (sudo требуется для монтирования образа и записи корневой файловой системы).

После завершения процедуры сборки в директории `output` будут доступны следующие файлы образов:

- `mcom02-buildroot-sdcard.img`

  Образ SD-карты, содержащий ОС GNU/Linux.

- `mcom02-*-uboot-spiflash.img`

  Образы SPI флеш-памяти, содержащие загрузчик U-Boot для соответствующих модулей.

6. ЗАПИСЬ ОБРАЗА SD-КАРТЫ

Для записи образа на SD-карту необходимо:

1. извлечь SD-карту из кард-ридера ПЭВМ и считать список устройств командой:

```
ls -la /dev/sd*
```

2. вставить SD-карту в кард-ридер ПЭВМ и повторно считать список устройств командой `ls -la /dev/sd*`. Вычесть из списка устройств после установки SD-карты список устройств до установки карты и получить устройство `/dev/sdX` и/или список устройств `/dev/sdX1,/dev/sdX2...` (где 1, 2, ... номера разделов SD-карты). В случае, если получен список устройств, то получить устройство `/dev/sdX` отбрасыванием последней цифры из устройства соответствующего первому разделу SD-карты `/dev/sdX1`.

3. записать образ на SD-карту:

```
sudo dd if=output/mcom02-buildroot-sdcard.img of=/dev/sdX bs=4M
sudo sync
```

4. извлечь SD-карту из кард-ридера ПЭВМ.

7. ПРОШИВКА ОБРАЗА SPI ФЛЕШ-ПАМЯТИ

Прошивка образа SPI флеш-памяти для соответствующего модуля (см. *“Сборка образа SD-карты и образа SPI флеш-памяти”*) выполнятся утилитой `flash-spi.py`, входящей в состав дистрибутива, согласно документу “Инструкция по прошивке SPI флеш-памяти отладочных модулей на базе 1892ВМ14Я”.

8. НАСТРОЙКА ОС

8.1 Увеличение объема ОЗУ

По умолчанию объем доступной ОЗУ равен 1 ГиБ. Для увеличения объема ОЗУ до 2 ГиБ необходимо активировать второй DDR-контроллер (если применимо для данного модуля). Для этого в файл `u-boot.env` необходимо добавить строку:

```
ddrctl_cmd=enable
```

8.2 Включение драйвера framebuffer vpoutfb

По умолчанию образ SD-карты собирается с выключенным модулем ядра vpoutfb. Для включения необходимо удалить фразу `modprobe.blacklist=vpoutfb` из файла `u-boot.env`.

8.3 Настройка сети

По умолчанию ОС настроена на получение сетевого адреса по DHCP. Настройка параметров сети задаётся в конфигурационном файле `/etc/network/interfaces` на корневой файловой системе. Полная документация по настройке сети доступна на странице https://wiki.debian.org/NetworkConfiguration.

Имя хоста по умолчанию — mcom. Для изменения имени хоста необходимо отредактировать конфигурационные файлы `/etc/hostname` и `/etc/hosts` на корневой файловой системе.

8.4 Параметры ядра Linux

Для передачи параметров ядру Linux используется переменная bootargs в файле `u-boot.env` в корне раздела boot. Загрузчик U-Boot считывает файл `u-boot.env` при загрузке, значение переменной bootargs используется в качестве параметров ядра.

Основные используемые параметры ядра:

1. `console=ttyS0,115200` — включает вывод сообщений ядра на UART0.
2. `earlyprintk` — включает вывод сообщений ядра на ранних этапах загрузки.

Список параметров ядра ОС Linux находится в `linux/Documentation/kernel-parameters.txt`.

8.5 Добавление программ в образ SD-карты

Система сборки Buildroot поддерживает добавление в сборку программ и библиотек пользователя. Подробная документация находится в директории `buildroot-script/buildroot/docs`.

8.6 Управление выводами GPIO из пространства пользователя

Управление выводами GPIO СнК осуществляется с помощью служебных файлов в `/sys/class/gpio` (см. https://www.kernel.org/doc/Documentation/gpio/sysfs.txt).

Управление выводом GPIO может быть недоступно, если он используется драйвером.

В соответствии с руководством пользователя на СнК выводы GPIO делятся на 4 группы: GPIOA, GPIOB, GPIOC, GPIOD. В Linux выводы GPIO обозначаются номерами. Соответствие базовых номеров для каждой группы приведено в Таблица 8.1. Внутри групп номера идут по порядку. Например, вывод GPIOA5 соответствует номеру 480 + 5 = 485 в Linux.

Таблица 8.1. Соответствие обозначений выводов GPIO номерам в Linux

Группа	Базовый номер в Linux
GPIOA	480
GPIOB	448
GPIOC	416
GPIOD	384

8.7 Разметка свободной области SD-карты и монтирование в `/data`

Для создания раздела из свободной области SD-карты и монтирования в директорию `/data` необходимо:

1. Запустить ОМ и выполнить логин, как описано в главе *“Проверка ОС”*.
2. Выполнить команду:

   ```
   create-data-partition && echo "Partition successfully created"
   ```

   Разметку считать завершённой успешно в случае вывода в терминал скриптом сообщения:

   ```
   Partition successfully created
   ```

9. ПРОВЕРКА ОС

Для проверки корректности работы ОС необходимо:

1. *Собрать образы SD-карты и SPI флеш-памяти*.
2. *Записать образ SD-карты*.
3. *Прошить образ SPI флеш-памяти*.
4. *Настроить ОС при необходимости*.
5. Зайти на устройство по протоколу SSH, логин: `root`, пароль: `root`.
6. Выполнить команду `uname -a`. ОС считать работающей корректно в случае вывода в терминал сообщения:

```
Linux mcom 4.1.27.1 #1 SMP Fri Jun 24 19:21:02 MSK 2016 armv7l
GNU/Linux
```

10. СООБЩЕНИЯ СИСТЕМНОМУ ПРОГРАММИСТУ

1. ОС выводит сообщения через последовательный интерфейс UART0. В случае успешной загрузки выводятся следующие сообщения:

```
DDR retention disabled

U-Boot SPL 2016.03.0.1 (Jun 23 2016 - 20:48:33)
DDR controllers init started
DDR controller #0 init done
DDR controller #1 init done
Trying to boot from SPI


U-Boot 2016.03.0.1 (Jun 23 2016 - 20:48:33 +0300)

CPU:   MCom-compatible
DRAM:  2 GiB
MMC:   sdhci: 0
*** Warning - bad CRC, using default environment

In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  0
Loading Linux...
reading u-boot.env
105 bytes read in 10 ms (9.8 KiB/s)
reading zImage
3187368 bytes read in 360 ms (8.4 MiB/s)
DDR controller #1 disabled
Kernel image @ 0x40008000 [ 0x000000 - 0x30a2a8 ]
## Flattened Device Tree blob at 50000000
   Booting using the fdt blob at 0x50000000
   Loading Device Tree to 4eff5000, end 4effffff ... OK

Starting kernel ...

Uncompressing Linux... done, booting the kernel.
[    0.000000] Booting Linux on physical CPU 0x0
[    0.000000] Linux version 4.1.27.1 (<username>@<hostname>)
(gcc version 5.2.0 (Buildroot 2015.08.1-00003-ge009e7d) )
#1 SMP Fri Jun 24 19:21:02 MSK 2016
...

Welcome!
mcom login:
```

2. Сообщения U-boot описаны в `u-boot/doc`.
3. Сообщения ядра ОС Linux описаны в `linux/Documentation`.
4. Сообщения об ошибках и действия по их устранению описаны в Таблица 10.1.

Таблица 10.1. Сообщения об ошибках при загрузке ОС

Сообщение об ошибке	Описание ошибки	Действия по устранению
`** Bad device mmc 0 **`	В устройстве отсутствует SD-карта	Установить прошитую SD-карту в отладочный модуль
`** Unrecognized filesystem type **`	Повреждена файловая система на первом разделе	Пересобрать образ загрузочной SD-карты, перепрошить SD-карту
`** File not found u-boot.env **`	На SD-карте отсутствует файл u-boot.env. Загрузка возможна, но сеть настроена не будет	Создать файл u-boot.env в корне SD-карты
`** File not found zImage **`	На карте отсутствует файл zImage	Пересобрать образ загрузочной SD-карты, перепрошить SD-карту
`Bad Linux ARM zImage magic!`	Файл zImage повреждён	Пересобрать образ загрузочной SD-карты, перепрошить SD-карту