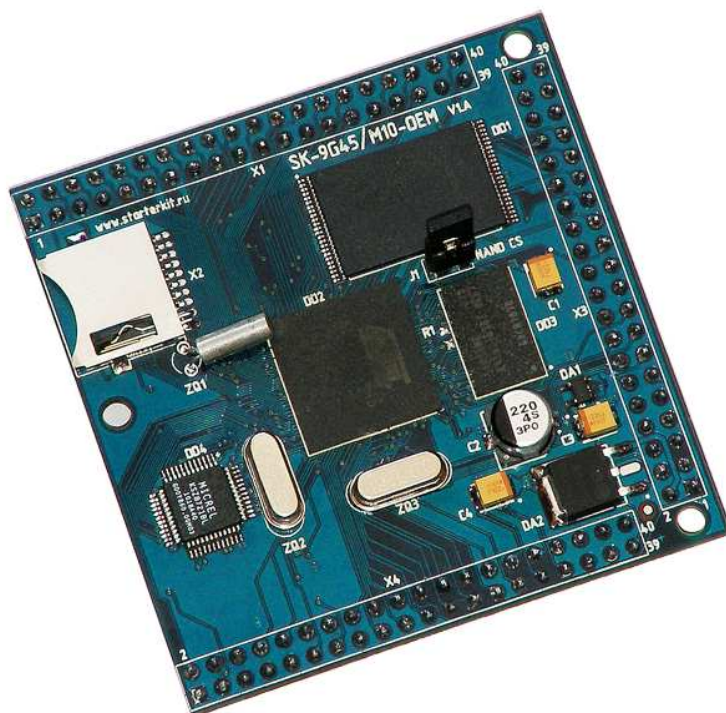


Процессорный модуль SK-9G45-OEM

Инструкция пользователя при совместном
использовании с платой SK-9G45-MB



SK-9G45-OEM:

Atmel AT91SAM9G45 (ARM9 400МГц)
DDR2 64Мбайт (32Мx16)
NAND Flash 256Мбайт
100/10М Ethernet
uSD держатель
Разъемы расширения

SK-9G45-OEM, возможность прямого подключения:

SK-9G45-MB – материнская плата для демонстрации возможностей процессорного модуля

SK-9G45-MB, возможность прямого подключения:

SK-MI0430FT-Plug или аналог – плата расширения LCD TFT 4,3” панелей
SK-ATM0700D4-Plug или аналог – плата расширения LCD TFT 7” панелей
SK-HDMI-Plug – плата расширения HDMI выхода
SK-SIMCOM-Plug – плата расширения GSM/GPS/3G модулей
SK-VideoADC-Plug – плата расширения видеозахвата

Общие характеристики

SK-9G45-OEM:

- Напряжение питания: 3,3В
- Потребляемый ток до 0,5А
- Габариты 60х60мм

SK-9G45-MB:

- Напряжение питания: 5-7В, при использовании USB-host 6В максимум, рекомендуемое напряжение 5В
При совместном использовании SK-T070-Plug, 6В максимум
- Потребляемый ток (зависит от подключения внешних модулей) до 2А
- Габариты 129х104мм

1. Назначение джамперов

1-ый вывод перемычек и переключающих перемычек помечен квадратной контактной площадкой!!!

SK-9G45-OEM:

- J1 позволяет исключить NAND Flash из системы, актуально при программировании через SAM-BA или загрузке с SD карты

SK-9G45-MB:

- J1 позволяет выбирать какой из сигналов (VS или FIELD) будет использован модулем ISI в качестве синхросигнала кадровой развертки
- J2 управляет подачей питающего напряжения с разъема X12 (USB device) для питания всей платы

2. Начало работы

Подключите RS232 кабель, идущий в комплекте, к COM порту PC (или USB-COM преобразователю), настройте терминальную программу на используемый COM порт с параметрами 115200 без управления потоком.

Подключите сетевой (Ethernet) кабель, настройте IP адрес сетевой карты PC в диапазоне 192.168.0.XXX.

При необходимости, подключите SK-MI0430FT-Plug или аналог.

Подключите питание (питающее напряжение – центральная жила разъема), в терминальной программе появятся следующие сообщения:

```
-- AT91bootstrap Project 3.0 --
-- SK-MAT91SAM9G45/M10
-- Compiled: Oct 30 2010 12:56:05 --
-I- Setting: MCK = 133MHz
-I- I cache enabled.
-I- Init DDRAM
-I- Init NAND Flash
-I- Nandflash ID is 0x9510DAEC
-I- Nandflash driver initialized
-I- Size of the whole device in bytes : 0x10000000
-I- Size in bytes of one single block of a device : 0x20000
-I- Number of blocks in the entire device : 0x800
-I- Size of the data area of a page in bytes : 0x800
-I- Number of pages in the entire device : 0x40
-I- Bus width : 8
-I- Copy " (262144 bytes) from NAND 0x00020000 to 0x73f00000
-I- Jump to 0x73f00000
```

```
U-Boot 2010.06 (Oct 08 2010 - 12:12:08)
```

```

DRAM: 64 MiB
## Unknown FLASH on Bank 1 - Size = 0x00000000 = 0 MB
Flash: 0 Bytes
NAND: 256 MiB
*** Warning - bad CRC or NAND, using default environment

In: serial
Out: serial
Err: serial
Net: macb0
macb0: PHY present at 1
macb0: link up, 100Mbps full-duplex (lpa: 0xc5e1)
Hit any key to stop autoboot: 0

NAND read: device 0 offset 0x80000, size 0x790000
7929856 bytes read: OK
## Booting kernel from Legacy Image at 70200000 ...
Image Name: Linux Kernel Image
Image Type: ARM Linux Kernel Image (gzip compressed)
Data Size: 2006613 Bytes = 1.9 MiB
Load Address: 70008000
Entry Point: 70008000
Verifying Checksum ... OK
Uncompressing Kernel Image ... OK

Starting kernel ...

Linux version 2.6.36-rc6 (user@debian) (gcc version 4.2.0 20070413 (prerelease)
(CodeSourcery Sourcery G++ Lite 2007ql-10)) #243 Sat Oct 9 02:49:21 EDT 2010
CPU: ARM926EJ-S [41069265] revision 5 (ARMv5TEJ), cr=00053177
CPU: VIVT data cache, VIVT instruction cache
Machine: SK-MAT91SAM9G45
Ignoring unrecognized tag 0x54410009
Memory policy: ECC disabled, Data cache writeback
Clocks: CPU 400 MHz, master 133 MHz, main 12.000 MHz
Built 1 zonelists in Zone order, mobility grouping on. Total pages: 16256
Kernel command line: console=ttyS0,115200 ubi.mtd=1 root=ubi0:nandfs rw rootfstype=ubifs
PID hash table entries: 256 (order: -2, 1024 bytes)
Dentry cache hash table entries: 8192 (order: 3, 32768 bytes)
Inode-cache hash table entries: 4096 (order: 2, 16384 bytes)
Memory: 64MB = 64MB total
Memory: 60928k/60928k available, 4608k reserved, 0K highmem
Virtual kernel memory layout:
vector : 0xffff0000 - 0xffff1000 ( 4 kB)
fixmap : 0xffff0000 - 0xffffe000 ( 896 kB)
DMA : 0xffa00000 - 0xffe00000 ( 4 MB)
vmalloc : 0xc4800000 - 0xf0000000 ( 934 MB)
lowmem : 0xc0000000 - 0xc4000000 ( 64 MB)
modules : 0xbf000000 - 0xc0000000 ( 16 MB)
 .init : 0xc0008000 - 0xc0028000 ( 128 kB)
 .text : 0xc0028000 - 0xc039c000 (3536 kB)
 .data : 0xc03b6000 - 0xc03d6140 ( 129 kB)
Hierarchical RCU implementation.
Verbose stalled-CPU detection is disabled.
NR_IRQS:192
AT91: 160 gpio irqs in 5 banks
Console: colour dummy device 80x30
console [ttyS0] enabled
Calibrating delay loop... 199.47 BogoMIPS (lpj=997376)
pid_max: default: 32768 minimum: 301
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
NET: Registered protocol family 16
bio: create slab <bio-0> at 0
SCSI subsystem initialized
usbcore: registered new interface driver usbfs
usbcore: registered new interface driver hub
usbcore: registered new device driver usb
i2c-gpio i2c-gpio.0: using pins 52 (SDA) and 53 (SCL)
cfg80211: Calling CRDA to update world regulatory domain
Switching to clocksource pit
NET: Registered protocol family 2
IP route cache hash table entries: 1024 (order: 0, 4096 bytes)
TCP established hash table entries: 2048 (order: 2, 16384 bytes)
TCP bind hash table entries: 2048 (order: 1, 8192 bytes)
TCP: Hash tables configured (established 2048 bind 2048)
TCP reno registered
UDP hash table entries: 256 (order: 0, 4096 bytes)
UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
NET: Registered protocol family 1
msgmni has been set to 119
io scheduler noop registered (default)

```

```

atmel_lcdfb atmel_lcdfb.0: backlight control is not available
atmel_lcdfb atmel_lcdfb.0: 255KiB frame buffer at 73940000 (mapped at ffa00000)
Console: switching to colour frame buffer device 60x34
atmel_lcdfb atmel_lcdfb.0: fb0: Atmel LCDC at 0x00500000 (mapped at c4814000), irq 23
atmel_usart.0: ttyS0 at MMIO 0xfeff0000 (irq = 1) is a ATMEL_SERIAL
atmel_usart.2: ttyS2 at MMIO 0xffff90000 (irq = 8) is a ATMEL_SERIAL
brd: module loaded
loop: module loaded
NAND device: Manufacturer ID: 0xec, Chip ID: 0xda (Samsung NAND 256MiB 3,3V 8-bit)
Scanning device for bad blocks
Bad eraseblock 227 at 0x000001c60000
Bad eraseblock 561 at 0x000004620000
Bad eraseblock 932 at 0x000007480000
Bad eraseblock 935 at 0x0000074e0000
Bad eraseblock 1313 at 0x00000a420000
Bad eraseblock 1438 at 0x00000b3c0000
Bad eraseblock 1559 at 0x00000c2e0000
Bad eraseblock 1876 at 0x00000ea80000
Bad eraseblock 1913 at 0x00000ef20000
Creating 2 MTD partitions on "atmel_nand":
0x0000000000000-0x0000010000000 : "Boot partition"
0x0000010000000-0x0000100000000 : "FS partition"
UBI: attaching mtd1 to ubi0
UBI: physical eraseblock size: 131072 bytes (128 KiB)
UBI: logical eraseblock size: 129024 bytes
UBI: smallest flash I/O unit: 2048
UBI: sub-page size: 512
UBI: VID header offset: 512 (aligned 512)
UBI: data offset: 2048
UBI: max. sequence number: 356
UBI: attached mtd1 to ubi0
UBI: MTD device name: "FS partition"
UBI: MTD device size: 240 MiB
UBI: number of good PEBs: 1911
UBI: number of bad PEBs: 9
UBI: max. allowed volumes: 128
UBI: wear-leveling threshold: 4096
UBI: number of internal volumes: 1
UBI: number of user volumes: 1
UBI: available PEBs: 262
UBI: total number of reserved PEBs: 1649
UBI: number of PEBs reserved for bad PEB handling: 19
UBI: max/mean erase counter: 1/0
UBI: image sequence number: 1311984814
atmel_spi atmel_spi.0: Atmel SPI Controller at 0xffffa4000 (irq 14)
UBI: background thread "ubi_bgt0d" started, PID 831
atmel_spi atmel_spi.1: Atmel SPI Controller at 0xffffa8000 (irq 15)
MACB_mii_bus: probed
eth0: Atmel MACB at 0xffffbc000 irq 25 (00:1f:f2:00:00:00)
eth0: attached PHY driver [Micrel KS8001 or KS8721] (mii_bus:phy_addr=ffffffff:01, irq=-1)
usbcore: registered new interface driver rt2500usb
usbcore: registered new interface driver rt73usb
usbcore: registered new interface driver rt2800usb
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
atmel-ehci atmel-ehci: Atmel EHCI UHP HS
atmel-ehci atmel-ehci: new USB bus registered, assigned bus number 1
atmel-ehci atmel-ehci: irq 22, io mem 0x00800000
atmel-ehci atmel-ehci: USB 2.0 started, EHCI 1.00
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 2 ports detected
ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
at91_ohci at91_ohci: AT91 OHCI
at91_ohci at91_ohci: new USB bus registered, assigned bus number 2
at91_ohci at91_ohci: irq 22, io mem 0x00700000
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 2 ports detected
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.
atmel_usba_udc atmel_usba_udc: MMIO registers at 0xffff78000 mapped at c4872000
atmel_usba_udc atmel_usba_udc: FIFO at 0x00600000 mapped at c4900000
mice: PS/2 mouse device common for all mice
setting trigger mode 2 for irq 149 failed (gpio_irq_type+0x0/0x20)
ads7846 spi3.0: trying pin change workaround on irq 149
ads7846 spi3.0: touchscreen, irq 149
input: ADS7843 Touchscreen as /devices/platform/spi_gpio.3/spi3.0/input/input0
rtc-at91sam9 at91_rtt.0: rtc core: registered at91_rtt as rtc0
rtc-at91sam9 at91_rtt.0: rtc0: SET TIME!
i2c /dev entries driver
at_hdmac at_hdmac: Atmel AHB DMA Controller ( cpy slave ), 8 channels
usbcore: registered new interface driver hiddev
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver

```

```
TCP cubic registered
NET: Registered protocol family 17
lib80211: common routines for IEEE802.11 drivers
rtc-at91sam9 at91_rtt.0: hctosys: unable to read the hardware clock
atmel_mci atmel_mci.0: Using dma0chan0 for DMA transfers
atmel_mci atmel_mci.0: Atmel MCI controller at 0xffff80000 irq 11, 1 slots
UBIFS: recovery needed
UBIFS: recovery completed
UBIFS: mounted UBI device 0, volume 0, name "nandfs"
UBIFS: file system size: 208244736 bytes (203364 KiB, 198 MiB, 1614 LEBs)
UBIFS: journal size: 10450944 bytes (10206 KiB, 9 MiB, 81 LEBs)
UBIFS: media format: w4/r0 (latest is w4/r0)
UBIFS: default compressor: lzo
UBIFS: reserved for root: 4952683 bytes (4836 KiB)
VFS: Mounted root (ubifs filesystem) on device 0:12.
Freeing init memory: 128K
Initializing random number generator... done.
eth0: link up (100/Full)
```

Welcome to SK-9G45-OEM module!

buildroot login:

Что означает, что система успешно загрузилась и готова к работе, во конце загрузки.

Для входа в консоль введите имя пользователя root, пароль не требуется (других пользователей в системе нет), после чего имеете полный консольный доступ к системе. Так же можно подключиться с помощью Telnet, FTP, HTTP, сетевой адрес платы 192.168.0.136. При подключении-отключении USB, SD/MMC карт памяти, они будут автоматически монтироваться-размонтироваться в системе.

Если был подключен SK-MI0430FT-Plug, на экране появится графическое изображение и сообщение о старте системы, при первой загрузке, необходимо откалибровать сенсорный экран системной библиотекой TSLIB, для этого запустите ts_calibrate и следуйте инструкциям, после чего можете запустить ts_test для демонстрации.

2.1. Подключение модулей расширения

SK-MI0430FT-Plug – разъем X1

В штатной поставке ядро сконфигурировано на использование данного модуля расширения, в качестве контроллера TP включен ADS7843 (или аналог). Для работы TP непосредственно со встроенным в AT91SAM9G45 контроллером TP, необходимо:

а) на модуле расширения разорвать переключки J5-J8, переключки J1-J4 перевести в положение 2-3

б) в свойствах ядра (скрипт make_menuconfig) зайти в меню Device Drivers -> Input device support -> Touchscreens выключить модуль «ADS7846/TSC2046/AD7873 and AD(S)7843 based touchscreens», включить модуль «Atmel Touchscreen Interface»

в) пересобрать и обновить ядро (или просто загрузить ядро по TFTP)

SK-ATM0700D4-Plug – разъем X1

а) в свойствах ядра (скрипт make_menuconfig) зайти в меню Device Drivers -> Graphics support -> Support for frame buffer devices -> Starterkit.ru TFT plug selection выбрать «SK-ATM0700D4-Plug»

б) если предполагается использовать встроенный в ARM контроллер TP, в свойствах ядра Device Drivers -> Input device support -> Touchscreens выключить модуль «ADS7846/TSC2046/AD7873 and AD(S)7843 based touchscreens», включить модуль «Atmel Touchscreen Interface». На самом модуле расширения разомкнуть переключки J2,J3,J6,J7, переключки J4,J5,J8,J9 перевести в положение 2-3.

Если предполагается использовать внешний контроллер TP (на модуле расширения), в свойствах ядра Device Drivers -> Input device support -> Touchscreens включить модуль «ADS7846/TSC2046/AD7873 and AD(S)7843 based touchscreens», выключить

модуль «Atmel Touchscreen Interface». На самом модуле расширения замкнуть перемычки J2,J3,J6,J7, перемычки J4,J5,J8,J9 перевести в положение 1-2

в) пересобрать и обновить ядро (или просто загрузить ядро по TFTP)

SK-HDMI-Plug – разъем X1

а) в свойствах ядра Device Drivers -> Input device support -> Touchscreens обязательно выключить модуль « Atmel Touchscreen Interface»

б) в свойствах ядра Device Drivers -> Graphics support -> Support for frame buffer devices -> Starterkit.ru TFT plug selection выбрать необходимые параметры экрана

в) на самом модуле расширения замкнуть J1, обращаю внимание, светодиод «DETECT» работает в «инверсном» режиме, т.е. когда монитор подключен – он гаснет

г) пересобрать и обновить ядро (или просто загрузить ядро по TFTP)

SK-VideoADC-Plug – разъем X6

Важно – на данный момент драйвер ISI не устанавливается в системе при разрешениях FB более 640x480, видимо есть неточности с выделением памяти, позже будем устранять причину.

В штатной поставке драйвер ISI включен в ядро, для теста достаточно запустить скрипт isi_test, в результате работы которого появится графический файл image.ppm.

SK-SIMCOM-Plug – разъем X6

В папке корневой ФС /bin/SK-SIMCOM-Plug находятся скрипты для операций с различными модемами данного модуля расширения.

3. Состав ОС Linux

Ядро 2.6.36, включая драйвера:

- Ethernet
- NAND flash
- USB-host
- USB-gadget
- I2C
- ISI
- SPI
- UART
- RTC
- WatchDog
- Frame Buffer
- TP ADS7843
- ...

4. Способы загрузки и содержимое корневой файловой системы

AT91SAM9G45 подразумевает различные возможные источники загрузки, на плате их предусмотрено два - NAND flash и SD/MMC карта

5.1. NAND flash

NAND flash разбита на две части:

- 1) 16M – для хранения загрузчиков, ядра системы и системы загрузки «safe mode»
- 2) 240M – раздел UBI файловой системы, используется в качестве корневой файловой системы

5.2. SD/MMC

Загрузчик (нестираемый, располагаемый в самом процессоре), в случае отсутствия приложения на NAND flash (или разомкнут джампер NAND CS), пытается загрузить приложение с MCI0 порта процессора (SD держатель). Карта должна иметь либо один раздел, либо обязательно первый раздел, отформатированный под FAT файловую систему и в корневой папке должны присутствовать:

- BOOT.BIN – загрузчик, верхний регистр имени важен (при копировании в Linux системах). Копирует из корневой папки карты 9g45oem.bin и передает ему управление
- 9g45oem.bin – образ Linux системы со встроенной (initramfs) корневой файловой системой

Дополнительно, в корневую папку uSD карты можно скопировать с поставляемого диска файлы:

- system_prepare_9g45oem – скрипт подготовки системы к исходному состоянию. Корневая система safe режима, сканирует внешние накопители на предмет наличия system_prepare_9g45oem файла и запускает его. В текущем варианте, скрипт очищает первый раздел NAND flash, копирует необходимые образы, форматирует и распаковывает корневую ФС во второй раздел NAND flash.
- bootstrap – загрузчик для запуска с NAND flash
- uboot – загрузчик u-boot
- zlinux – ядро linux
- zlinux_safe – ядро linux включая корневую ФС
- rootfs_9g45oem.tgz – архив корневой ФС

При наличии всех вышеперечисленных файлов на SD карте, система загрузится, подготовит, отформатирует и скопирует необходимые файлы на NAND flash, обращаю ВНИМАНИЕ – джампер NAND CS должен быть разомкнут во время включения-сброса платы и замкнут в первые секунды старта ядра (для того чтобы ядро правильно инсталлировало драйвер MTD устройства)!

При отсутствии system_prepare_9g45oem файла, система просто загрузится.

Корневая файловая система (ФС), в поставляемом варианте платы, хранится в NAND flash и монтируется во время загрузки, поэтому, следует внимательней относиться к изменениям в скриптах загрузки системы.

Корневая ФС содержит набор базовых приложений (большинство из которых являются реализацией мультифункционального приложения BusyBox), содержит:

- HTTPD – сервер HTTP
- FTPD – сервер FTP
- Telnetd – сервер Telnet
- TFTP – утилита приема-передачи файлов по TFTP протоколу
- Z-modem утилиты (для обмена файлами через COM порт)
- Microcom – терминальная программа
- TS-lib – набор утилит для операций с сенсорной панелью
- Memtester – тест памяти
- Mplayer – медиа-проигрыватель
- MC – файловый менеджер
- ...

На случай аварии корневой файловой системы, предусмотрен режим «Safe boot», для его активации необходимо прервать загрузку в U-boot (нажав на любую клавишу) и выполнить команду «run boot_safe». Загрузится образ системы, в котором корневая ФС расположена в памяти и можно будет приступить к ремонту основной корневой ФС, например, запустить скрипт «install_rootfs», в результате работы которого будет заново отформатирован второй раздел NAND flash, скопирован с TFTP сервера и распакован архив корневой ФС.

5. Виртуальная машина VMware

Для сборки ядра и корневой ФС используется виртуальная машина VMware с установленной ОС Debian, в состав которой входят все исходные тексты, компилятор и утилиты для сборки (toolchain), скрипты. Так же в виртуальной машине установлены и настроены сервисы для удобства взаимодействия с «материнской» ОС и отладочной платой: SSH, FTP, TFTP.

Разархивируйте файл “ SK-9G45-OEM_linux_build_machine.exe“, установите VMware-player или VMware, откройте и проект виртуальной машины.

Для работы необходимо настроить сетевые интерфейсы (появляющиеся после установки VMware), присвоив им описываемые ниже IP адреса:

Eth0 (Bridget) с адресом 192.168.0.2, предусмотрен для взаимодействия с платой, для загрузки образов по TFTP ... Т.е. для нормальной работы, потребуется присвоить IP адрес PC сетевой карты (к которой подключается отладочная плата) 192.168.0.1

Eth1 (Host-only) с адресом 192.168.2.2, задуман для взаимодействия с PC (т.к. Bridget интерфейс отключается при физически выключенном кабеле), в частности, для возможности копирования файлов из виртуальной системы по FTP. В свойствах сетевых устройств, этому виртуальному адаптеру нужно присвоить IP 192.168.2.1

После правильной настройки (и с подключенной платой) должны успешно проходить PING с PC по адресам 192.168.2.2, 192.168.0.2, 192.168.0.136.

После того, как сетевые интерфейсы настроены, можно запускать виртуальную машину, после загрузки ее не обязательно выключать, достаточно будет нажать кнопку паузы и во время следующего сеанса работы не придется ждать загрузки виртуальной ОС, но при этом, в некоторых случаях, нужно следить за системными временем, особенно при копировании новых файлов (имеющих более позднюю дату создания относительно системы) для сборки.

По умолчанию, в системе присутствует два пользователя:

- root, пароль 123456
- user, пароль 123456 (настоятельно рекомендую работать под этим пользователем, или создать нового, но не вести всю работу под root)

После входа, переключаемся на консоль (Ctrl+Alt+F(1-6)) (потребуется в опциях VMware освободить сочетание клавиш Ctrl+Alt - по умолчанию это выход из окна виртуальной машины), запускаем MidnightComander (mc).

Основная рабочая папка /home/user/src, ее содержимое:

- buildroot-2010.08 - пакет сборки корневой файловой системы
- buildroot-2010.08_safe - пакет сборки корневой файловой системы для «Safe mode»
- linux-2.6.36 - ядро, скрипты сборки внутри

- u-boot-2010.09 - вторичный загрузчик (загружается SAM-BA в NAND flash по адресу 0x20000)

В корневом каталоге ядра присутствует два скрипта:

make_kernel – собирает ядро и копирует файл в папку TFTP сервера

make_menuconfig – запускает конфигурационное меню ядра

Так же в корневом каталоге ядра присутствуют образец конфигурационного файла для «Safe mode» загрузки – «config_safe».

В корневом каталоге buildroot-2010.08 присутствует скрипт:

build_system – собирает файловую систему и запускает скрипт сборки ядра

Например, необходимо обновить ядро Linux, для этого:

- запускаем скрипт linux-2.6.XX/make_kernel
- включаем/перезагружаем плату с подключенным Ethernet и RS232 кабелями
- прерываем в u-boot процесс загрузки нажатием любой клавиши
- выполняем “run system_update”

6. Общий принцип работы системы

После подачи питания (перезагрузки), процессор запускает первичный загрузчик (находится во внутренней не перепрограммируемой ROM) и по определенному алгоритму определяет наличие исполняемого кода во внешних носителях. Если приложение не найдено, процессор остается в режиме, который подразумевает взаимодействие с ним утилиты SAM-BA, которая позволяет программировать внешние носители, подключенные к процессору.

Поскольку внешняя DDR2 (или любая другая память не инициализирована), первое запускаемое приложение должно быть загрузчиком т.к. его максимальный размер не может превышать размера внутренней памяти процессора. Это приложение (загрузчик) в первую очередь должен проинициализировать внешнюю память (например, правильно настроить параметры DDR2), скопировать исполняемое приложение из внешней Flash памяти во внешнюю DDR2 память и передать ему управление.

В нашем контексте, первым приложением является так называемый bootstrap загрузчик (предоставляемый фирмой Atmel и адаптированный под конкретную плату), который инициализирует DDR2, копирует из NAND приложение и запускает его.

Вторым приложением так же является загрузчик (u-boot) но с уже более обширными возможностями, например, он умеет копировать файлы по TFTP, поддерживает целый набор команд и режимов. В переменных окружения u-boot есть команда запуска, в которой указано, по какому адресу NAND flash следует прочитать образ ядра, куда этот образ памяти записать и по какому адресу запустить. Следующие сообщения консоли иллюстрируют этот процесс:

```
U-Boot 2010.06 (Oct 08 2010 - 12:12:08)
DRAM: 64 MiB
## Unknown FLASH on Bank 1 - Size = 0x00000000 = 0 MB
Flash: 0 Bytes
NAND: 256 MiB
*** Warning - bad CRC or NAND, using default environment

In: serial
Out: serial
Err: serial
Net: macb0
macb0: PHY present at 1
```

```
macb0: link up, 100Mbps full-duplex (lpa: 0xc5e1)
Hit any key to stop autoboot:  0
```

```
NAND read: device 0 offset 0x80000, size 0x790000
7929856 bytes read: OK
```

Переменные окружения u-boot подразумевают возможность хранить команды и некоторые аргументы (например, IP адрес сервера) в специально отведенном блоке flash памяти, сообщение вида “ Warning - bad CRC, using default environment” говорит о том, что u-boot не находит валидные переменные окружения и использует переменные окружения по умолчанию.

Перед запуском ядра Linux, оно первым делом проверяет контрольную сумму собственного архива и распаковывает себя и корневую ФС, иллюстрация:

```
## Booting kernel from Legacy Image at 70200000 ...
Image Name:      Linux Kernel Image
Image Type:      ARM Linux Kernel Image (gzip compressed)
Data Size:       2006613 Bytes = 1.9 MiB
Load Address:    70008000
Entry Point:     70008000
Verifying Checksum ... OK
Uncompressing Kernel Image ... OK
```

```
Starting kernel ...
```

Далее идет инициализация всей системы, драйверов, файловых систем, после чего управление передается скрипту начального запуска.

7. Программирование внешних носителей с помощью SAM-BA

В большинстве случаев, обновить-восстановить систему можно из загрузчика u-boot, но если он или bootstrap поврежден, тогда восстановить систему можно только с помощью утилиты (предоставляемой фирмой Atmel) SAM-BA. К сожалению, с ростом версии этой утилиты, стабильность ее работы не улучшается, скорее наоборот ...

В добавок, на текущий момент AT91SAM9G45 содержит ошибку USB соединения во внутреннем загрузчике (см. errata), к счастью, ошибка не фатальна и при многократном повторе попытки соединения, связь установить можно.

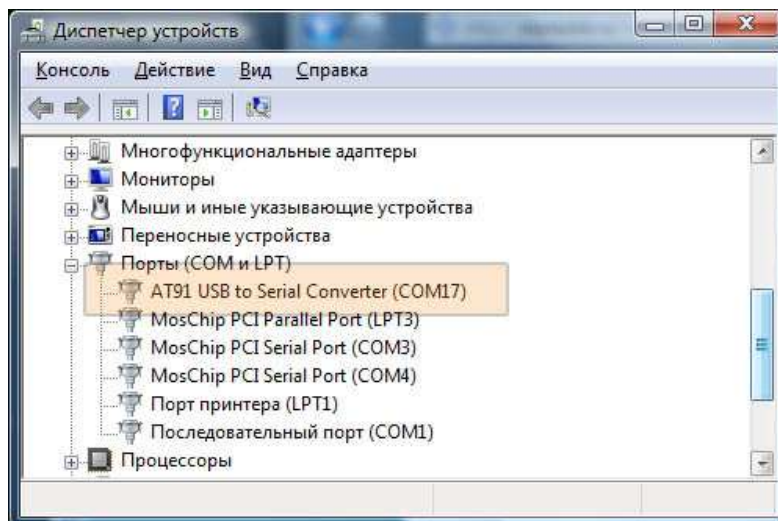
В общем виде, возможно подключение к плате 3-мя интерфейсами USB-device, через COM порт, JTAG (в текущей версии его поддержка отсутствует, присутствовала в более ранних версиях SAM-BA). Самым «надежным» себя зарекомендовал USB интерфейс, обращаю внимание, после установки драйвера (при первом подключении платы) не следует менять порт USB хаба вашего PC для взаимодействия с платой.

Для того, чтобы процессор был способен взаимодействовать с SAM-BA, необходимо, чтобы его первоначальный загрузчик не смог найти исполняемые коды во внешних носителях (в случае с JTAG интерфейсом этого условия не требуется), для этого достаточно разомкнуть перемычки J1 и выключить-включить питание платы, после этого, J1 следует замкнуть.

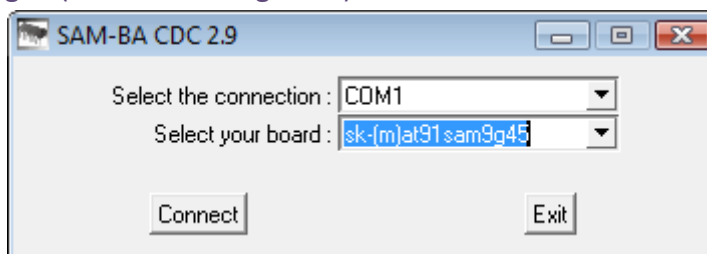
7.1 Распаковываем sam-ba_2.9_cdc_xp_vista.rar, инсталляции не требует, скрипты плат уже в архиве.

7.2 Подключаем USB-A кабель к разъему X12. При включении питания, система должна найти новое USB устройство (если ранее эта процедура не выполнялась), присваиваем ему драйвер /drv/atm6124_cdc.inf .

В результате манипуляций, при подключении платы, в диспетчере устройств должен появляться "AT91 USB to Serial Converter COMXX":



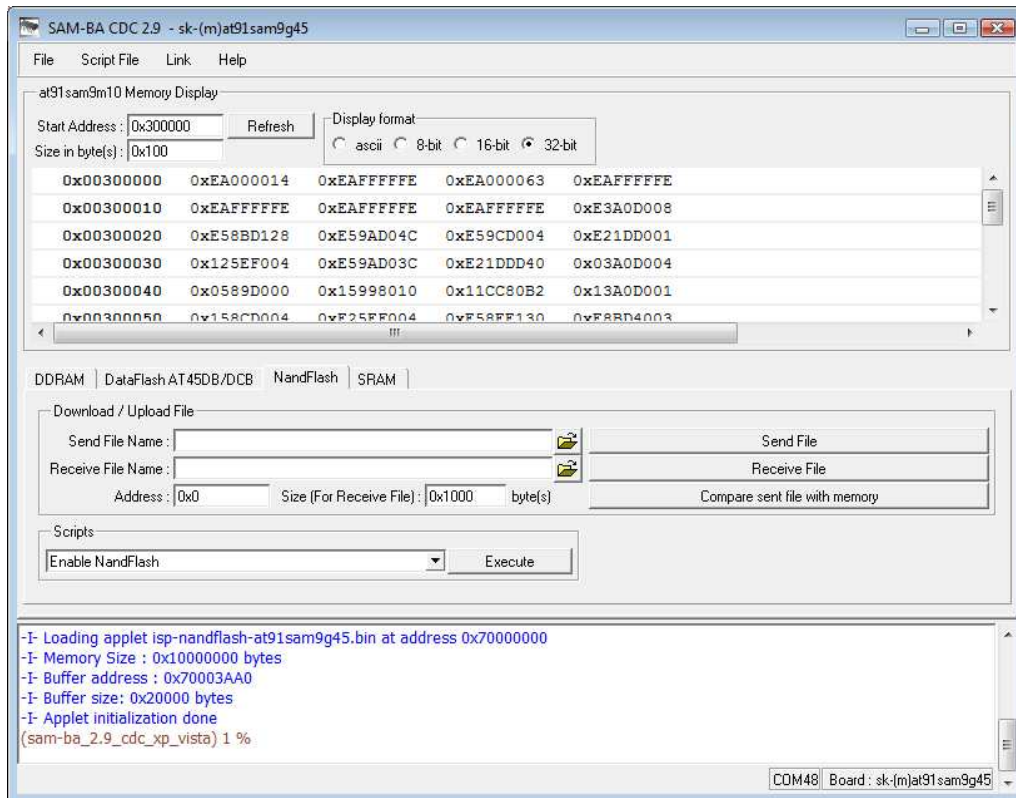
7.3 Запускаем SAM-BA, выбираем COM порт AT91 USB to Serial Converter, выбираем плату sk-(m)at91sam9g45 (или at91sam9g45-ek):



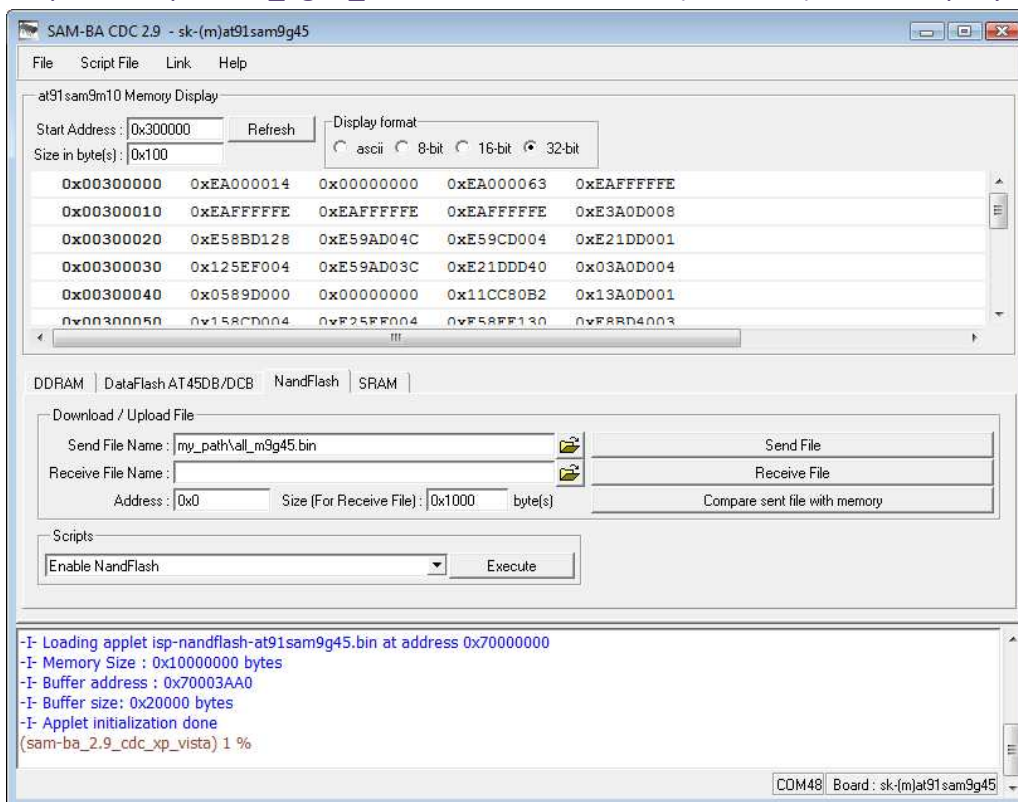
В виду описываемой выше ошибки внутреннего загрузчика AT91SAM9G45, соединение может устанавливаться далеко не с первого раза, необходимо повторять (возможно потребуется это сделать не один десяток раз) процедуру:

- 1) удалить через диспетчер задач SAM-BA
- 2) нажать кнопку сброса на плате
- 3) запустить SAM-BA

7.4 Выполняем скрипт Enable NandFlash:



7.5 Открываем файл all_9g45_оem.bin и записываем (Send File) его по адресу 0:



7.6 По окончании записи (примерно через 2-3 минуты) перезагружаемся, если основная корневая ФС была удалена, останавливаем процесс загрузки u-boot, выполняем "run safe_boot".

8. Назначение контактов модуля SK-9G45-OEM

8.1 Разъем X1

N	Наименование в структурной схеме	Назначение
1	GND	0В
2	CPU_PD12	PORT D, 12 бит
3	LCD_G2	PORT E, 17 бит
4	LCD_R7	PORT E, 14 бит
5	LCD_B2	PORT E, 25 бит
6	LCD_B4	PORT E, 27 бит
7	LCD_B6	PORT E, 29 бит
8	LCD_B3	PORT E, 26 бит
9	LCD_B7	PORT E, 30 бит
10	LCD_B5	PORT E, 28 бит
11	LCD_G7	PORT E, 22 бит
12	LCD_G6	PORT E, 21 бит
13	LCD_G5	PORT E, 20 бит
14	LCD_B0	PORT E, 23 бит
15	LCD_G4	PORT E, 19 бит
16	LCD_G0	PORT E, 15 бит
17	LCD_G1	PORT E, 16 бит
18	LCD_R0	PORT E, 7 бит
19	LCD_G3	PORT E, 18 бит
20	LCD_B1	PORT E, 24 бит
21	LCD_R6	PORT E, 13 бит
22	LCD_R5	PORT E, 12 бит
23	LCD_R3	PORT E, 10 бит
24	LCD_R2	PORT E, 9 бит
25	LCD_DE	PORT E, 6 бит
26	LCD_DTCK	PORT E, 5 бит
27	LCD_R1	PORT E, 8 бит
28	LCD_VS	PORT E, 3 бит
29	LCD_R4	PORT E, 11 бит
30	LCD_HS	PORT E, 4 бит
31	CPU_PC10	PORT C, 10 бит
32	CPU_PD25	PORT D, 25 бит
33	CPU_PC11	PORT C, 11 бит
34	CPU_PC7	PORT C, 7 бит
35	CPU_PC0	PORT C, 0 бит
36	CPU_PC2	PORT C, 2 бит
37	CPU_PD22	PORT D, 22 бит
38	CPU_PD23	PORT D, 23 бит
39	CPU_PD20	PORT D, 20 бит

40	CPU_PD21	PORT D, 21 бит
----	----------	----------------

8.2 Разъем X3

N	Наименование в структурной схеме	Назначение
1	GND	0В
2	3.3V	+3.3В
3	UART2_TX	PORT B, 6 бит
4	DRXD	PORT B, 12 бит
5	UART0_RX	PORT B, 18 бит
6	DTXD	PORT B, 13 бит
7	UART0_TX	PORT B, 19 бит
8	AC97_SDI	PORT D, 6 бит
9	AC97_BCLK	PORT D, 9 бит
10	AC97_SYNC	PORT D, 8 бит
11	AC97_SDO	PORT D, 7 бит
12	ISI_D3	PORT B, 23 бит
13	ISI_D5	PORT B, 25 бит
14	ISI_D4	PORT B, 24 бит
15	ISI_D1	PORT B, 21 бит
16	ISI_VS	PORT B, 29 бит
17	ISI_D10	PORT B, 8 бит
18	RESET	RESET
19	BATTERY	Питание RTC части процессора (1.8-3,6В)
20	WKUP	WKUP
21	SHDN	SHDN
22	EBI1_NWE	Шина EBI1, NWE сигнал
23	EBI1_NRD	Шина EBI1, NRD сигнал
24	EBI1_NCS0	Шина EBI1, NCS0 сигнал
25	EBI1_A14	Шина EBI1, A14 сигнал
26	EBI1_A15	Шина EBI1, A15 сигнал
27	EBI1_D1	Шина EBI1, D1 сигнал
28	EBI1_D0	Шина EBI1, D0 сигнал
29	EBI1_D3	Шина EBI1, D3 сигнал
30	EBI1_D2	Шина EBI1, D2 сигнал
31	EBI1_D5	Шина EBI1, D5 сигнал
32	EBI1_D4	Шина EBI1, D4 сигнал
33	EBI1_D7	Шина EBI1, D7 сигнал
34	EBI1_D6	Шина EBI1, D6 сигнал
35	CPU_PC17	PORT C, 17 бит
36	CPU_PC12	PORT C, 12 бит
37	CPU_PC25	PORT C, 25 бит
38	CPU_PC20	PORT C, 20 бит

39	CPU_PC27	PORT C, 27 бит
40	CPU_PC28	PORT C, 28 бит

8.3 Разъем X4

N	Наименование в структурной схеме	Назначение
1	GND	0В
2	TWD	PORT A, 20
3	TWC	PORT A, 21
4	P_L1	Ethernet PHY LED0 сигнал (светодиодная индикация), см.
5	P_L2	Ethernet PHY LED1 сигнал (светодиодная индикация), см.
6	ERXM	Ethernet PHY RX- сигнал, схему подключения см. SK-9G45-MB
7	ERXP	Ethernet PHY RX+ сигнал, схему подключения см. SK-9G45-MB
8	SPIO_MISO	PORT B, 0
9	ETP	Ethernet PHY «средняя точка», см. схему подключения SK-
10	SPIO_MOSI	PORT B, 1
11	SPIO_NCS	PORT B, 3
12	SPI1_SPCK	PORT B, 16
13	SPIO_SPCK	PORT B, 2
14	ETXM	Ethernet PHY TX- сигнал, см. схему подключения SK-9G45-MB
15	ETXP	Ethernet PHY TX+ сигнал, см. схему подключения SK-9G45-MB
16	SPI1_MISO	PORT B, 14
17	UART1_RX	PORT B, 5
18	UART2_RX	PORT B, 7
19	SPI1_MOSI	PORT B, 15
20	SPI1_NCS	PORT B, 17
21	CPU_PD3	PORT D, 3
22	ISI_D9	PORT B, 9
23	ISI_HS	PORT B, 30
24	CPU_PB28	PORT B, 28
25	ISI_D6	PORT B, 26
26	ISI_D0	PORT B, 20
27	ISI_D7	PORT B, 27
28	ISI_MCK	PORT B, 31
29	ISI_D2	PORT B, 22
30	USB_H1P	USB 0 порт "+"
31	USB_H1M	USB 1 порт "-"
32	USB_HM	USB 0 порт "-"
33	USB_HP	USB 1 порт "+"
34	C1_CMD	PORT A, 22
35	C1_D1	PORT A, 24
36	C1_D0	PORT A, 23
37	C1_D3	PORT A, 26

38	C1_D2	PORT A, 25
39	C1_CLK	PORT A, 31
40	UART1_TX	PORT B, 4

Габаритный чертеж модуля в DXF формате можно скачать на странице описания модуля на сайте www.starterkit.ru, на странице описания материнской платы SK-9G45-MB можно скачать проектные файлы печатной платы для посадочного места модуля.

CPU
ARM926EJ-S
400MHz

Memory
DDR2
64MBytes

Flash memory
NAND
256MBytes

Ethernet PHY
100/10M

uSD
(Bootable MCI0)

Power system

External
IO Interfaces:
2 x USB
Ethernet RX/TX
SD/MMC (MCI1)
LCD
TSADC
ISI
AC97
4 x UART
2 x SPI
I2C
EBI1 8 bit databus + 2 bit address +
CS + OE + WE

X5	
<	Name
1	USB1+
2	USB1-
3	USB0+
4	USB0-
5	USB1+
6	USB1-

X1			
GND	1	PD12	2
LCD_G2	3	PE17	4
LCD_B2	5	PE25	6
LCD_B6	7	PE29	8
LCD_B7	9	PE30	10
LCD_G7	11	PE22	12
LCD_G5	13	PE20	14
LCD_G4	15	PE19	16
LCD_G1	17	PE16	18
LCD_G3	19	PE18	20
LCD_R6	21	PE13	22
LCD_R3	23	PE10	24
LCD_DE	25	PE6	26
LCD_R4	27	PE8	28
LCD_R1	29	PE11	30
CPU_PC10	31	PC10	32
CPU_PC11	33	PC11	34
CPU_PC0	35	PC0	36
TS3	37	PD22	38
TS1	39	PD20	40

X3			
GND	1		2
UART2_TX	3	PB6	4
UART0_RX	5	PB18	6
UART0_TX	7	PB19	8
AC97_BCLK	9	PD9	10
AC97_SDO	11	PD7	12
ISL_D5	13	PB25	14
ISL_D1	15	PB21	16
ISL_D10	17	PB8	18
BATTERY	19		20
SHDN	21	SHDN	22
EBI1_NRD	23	EBI1_NRD	24
EBI1_A14	25	EBI1_A14	26
EBI1_D1	27	EBI1_D1	28
EBI1_D3	29	EBI1_D3	30
EBI1_D5	31	EBI1_D5	32
EBI1_D7	33	EBI1_D7	34
CPU_PC17	35	PC17	36
CPU_PC25	37	PC25	38
CPU_PC27	39	PC27	40

X4			
GND	1	PA20	2
TWC	3	PA21	4
P_L2	5	LED1	6
ERXP	7	RX+	8
ETP	9	CT	10
SPI0_NCS	11	PB3	12
SPI0_SPCK	13	PB3	14
ETXP	15	PB2	16
UART1_RX	17	PB5	18
SPI1_MOSI	19	PB15	20
CPU_PD3	21	PD3	22
ISL_HS	23	PB30	24
ISL_D6	25	PB26	26
ISL_D7	27	PB27	28
ISL_D2	29	PB22	30
USB_HM	31	USB1+	32
USB_HP	33	USB0+	34
C1_D0	35	PA24	36
C1_D3	37	PA26	38
C1_CLK	39	PA31	40