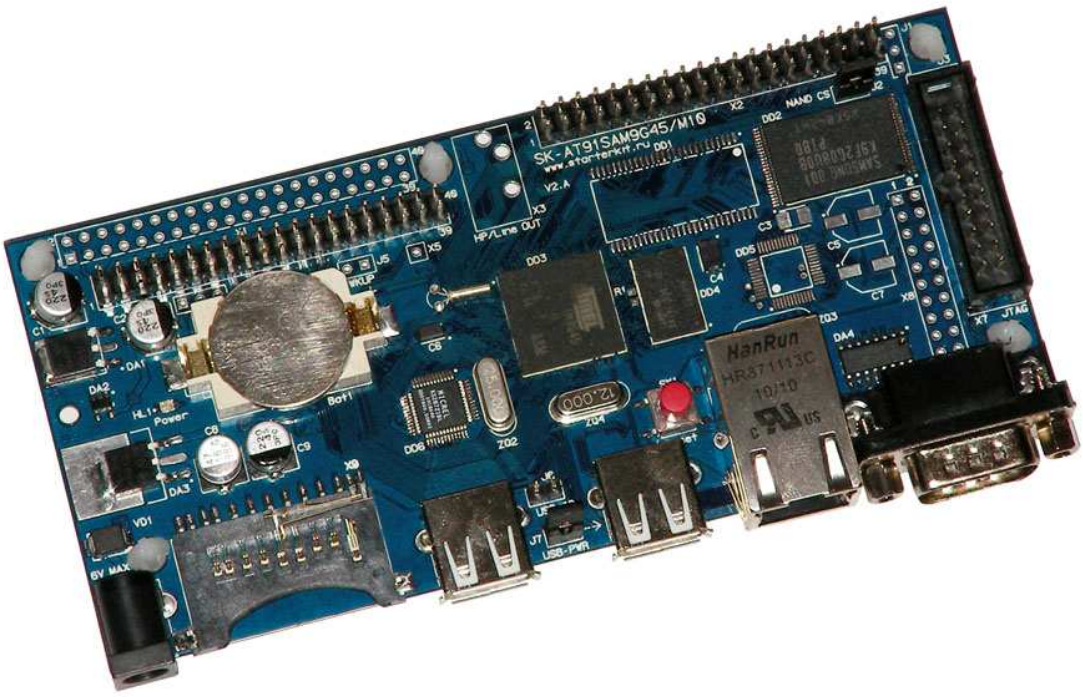
Отладочная плата SK-MAT91SAM9G45

Инструкция пользователя



г.Ижевск 2010г.

SK-MAT91SAM9G45:

- Atmel AT91SAM9G45 (ARM9 400МГц)
- DDR2 64Мбайт (32Mx16)
- NAND Flash 256Мбайт
- 100/10M Ethernet
- SD/MMC держатель
- USB Host/Device
- USB Host
- RS232
- RTC (часы реального времени)
- JTAG разъем
- Разъемы расширения
- Система питания

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

Комплект поставки: отладочная плата SK-AT91SAM9G45, RS232 кабель, ссылка для скачивания на необходимые материалы

1. Общие характеристики

- Напряжение питания: 5-7В, при использовании USB-host 6В максимум, рекомендуемое напряжение 5В
  При совместном использовании SK-T070-Plug, 6В максимум
- Потребляемый ток до 1А
- Габариты 138x64x20мм

2. Назначение джамперов

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

- J2 позволяет исключить NAND Flash из системы, актуально при программировании через SAM-BA или загрузке с SD карты
- J4 позволяет выбирать какой из сигналов (VS или FIELD) будет использован модулем ISI в качестве синхросигнала кадровой развертки
- J5 позволяет подключить внешнюю кнопку управления питанием
- J6 USB-ID – позволяет конфигурировать один из USB портов как device/host, актуально для OTG режима работы порта
- J7 управляет подачей питающего напряжения к X13, ВНИМАНИЕ!!! В режиме работы порта как Device (например, при программировании NAND flash через SAM-BA) должен быть разомкнут
- J8 позволяет подключить питание к USB разъему X12 минуя транзисторные каскады управления
- J9 позволяет подключить питание к USB разъему X13 минуя транзисторные каскады управления

  По умолчанию замкнуты перемычки: J2

3. 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 2007q1-10)) #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 unrecognised 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  : 0xfff00000 - 0xfffe0000   ( 896 kB)
    DMA     : 0xffa00000 - 0xffe00000   (   4 MB)
    vmalloc : 0xc4800000 - 0xfee00000   ( 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-CPUs 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 0xfeffee00 (irq = 1) is a ATMEL_SERIAL
atmel_usart.2: ttyS2 at MMIO 0xfff90000 (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":
0x000000000000-0x000001000000 : "Boot partition"
0x000001000000-0x000010000000 : "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 0xfffa4000 (irq 14)
UBI: background thread "ubi_bgt0d" started, PID 831
atmel_spi atmel_spi.1: Atmel SPI Controller at 0xfffa8000 (irq 15)
MACB_mii_bus: probed
eth0: Atmel MACB at 0xfffbc000 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 0xfff78000 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 0xfff80000 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-MAT91SAM9G45

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 звгрузке, необходимо откалибровать тачскрин панель с системной библиотекой TSLIB, для этого запустите ts_calibrate и следуйте инструкциям, после чего можете запустить ts_test для демонстрации.

3.1. Подключение модулей расширения

SK-MI0430FT-Plug – разъем X2

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 перемычки J5-J8, перемычки 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 X2

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 перемычки J2,J3,J6,J7, перемычки 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

влючить модуль «ADS7846/TSC2046/AD7873 and AD(S)7843 based touchscreens», вылючить 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 X2

а) в свойствах ядра Device Drivers -> Input device support -> Touchscreens обязательно вылючить 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 X4

Важно – на данный момент драйвер ISI не инсталлируется 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 графичесий файл image.ppm.

SK-SIMCOM-Plug – разъем X4

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

4. Состав ОС Linux

Ядро 2.6.36, включая драйвера:

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

5. 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 ситемы и смтемы 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 mat91_sd.bin и передает ему управление
- mat91_sd.bin – образ Linux системы со встроенной (initramfs) корневой файловой системой

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

- system_prepare_mat9g45 – скрипт подготовки системы к исходному состоянию. Корневая система safe режима, сканирует внешние накопители на предмет наличия system_prepare_mat9g45 файла и запускает его. В текущем варианте, скрипт очищает первый раздел NAND flash, копирует необходимые образы, форматирует и распакорвывает 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_mat9g45.tgz – архив корневой ФС

При наличии всех вышеперечисленных файлов на SD карте, система загрузится и 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_mat9g45 файла, система просто загрузится.

Корневая файловая система (ФС), в поставляемом варианте платы, хранится в NAND flash и монтируется во время загрузки, поэтому, следует внимательней относится 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.

6. 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-MAT91SAM9G45_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, задуман для взаимодействие 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-2010.06 - 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 клавишы
- выполняем “run system_update”

7. 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 bootstap загрузчик (предоставляемый фирмой Atmel и адаптированный под конкретную плату), котоый 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.

8. 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.

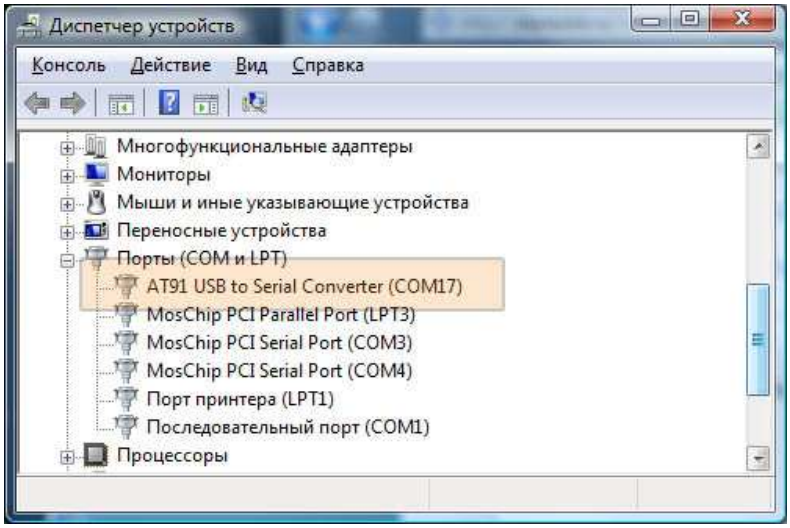
В общем виде, возможно подключение к плате 3-мя интерфейсами USB-device, через COM порт, JTAG (в текущей версии его поддержка отсутствует, присутствовала в более ранних версиях SAM-BA). Самым «надежным» себя зарекомендовал USB интерфейс, обращаю внимание, после установки драйвера (при первом подключении 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 J2 и выключить-включить питание платы, после этого, J2 следует замкнуть.

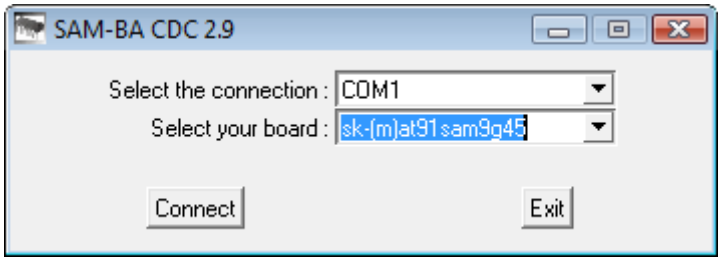
8.1 Распаковываем sam-ba_2.9_cdc_xp_vista.rar, инсталляции не требует, скрипты плат уже в архиве.

8.2 Подключаем USB-A кабель к разъему X13, J7 обязательно разомкнут!!! 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":



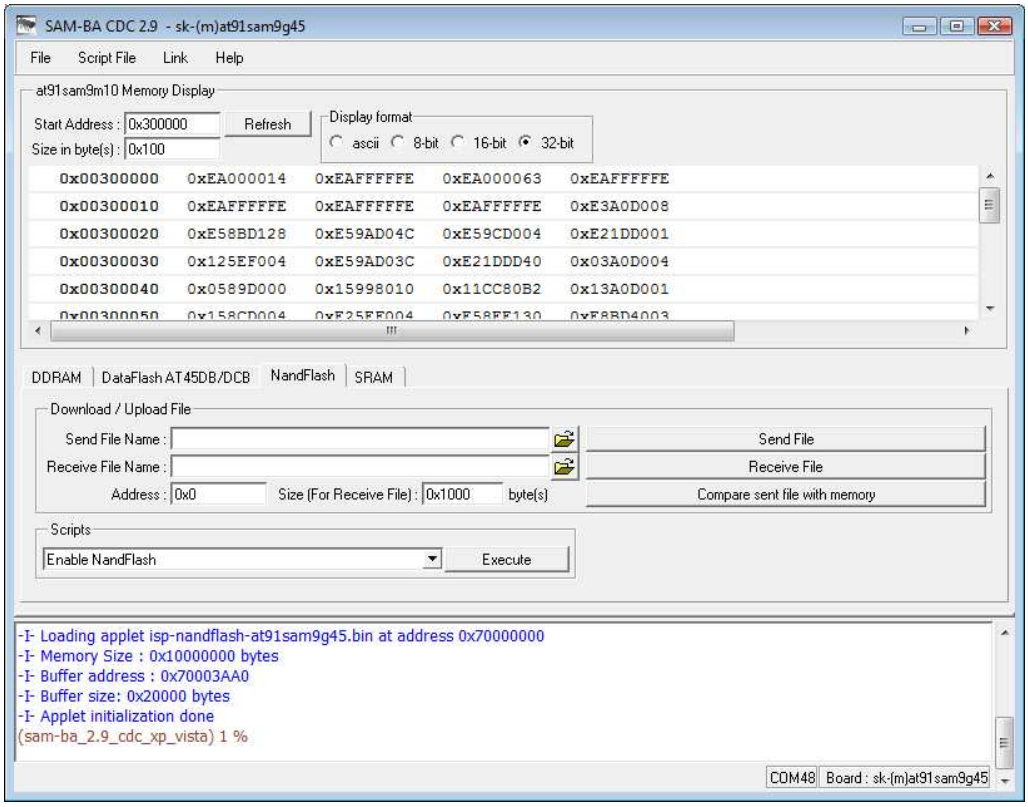
8.3 Запускаем SAM-BA, выбираем COM порт AT91 USB to Serial Converter, выбираем плату sk-(m)at91sam9g45 (или at91sam9g45-ek):



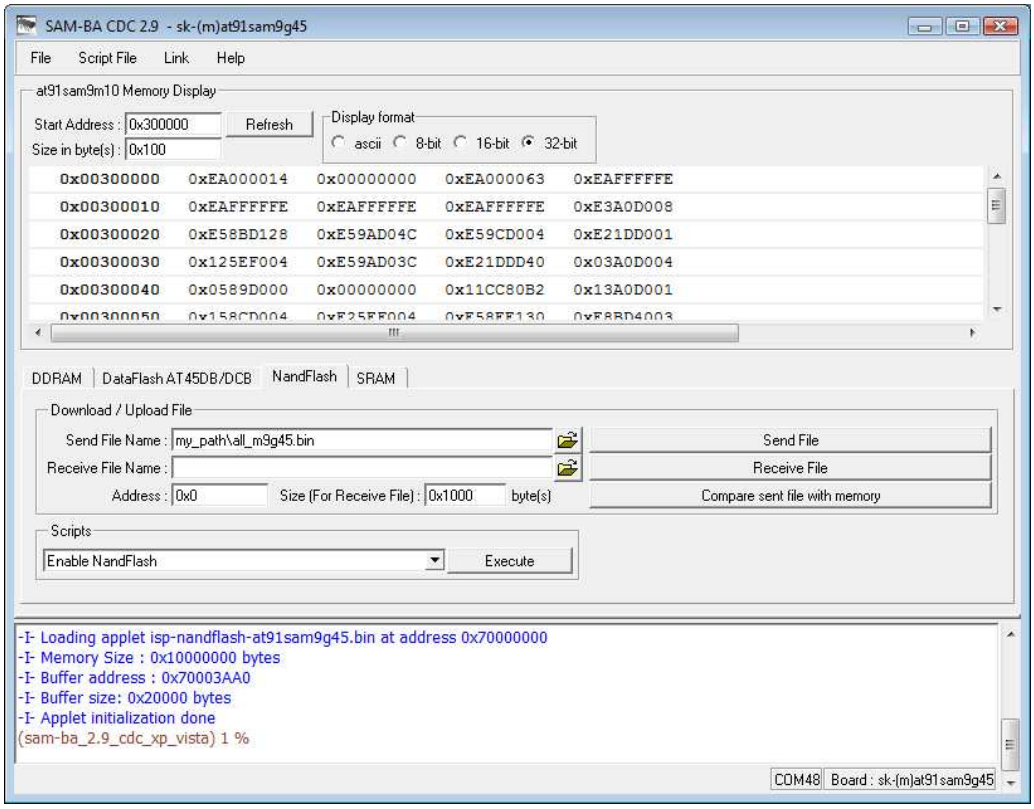
В виду описываемой выше ошибки внутреннего загрузчика AT91SAM9G45, соединение может устанавливаться далеко не с первого раза, необходимо повторять (возможно потребуется это сделать не один десяток раз) процедуру:

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

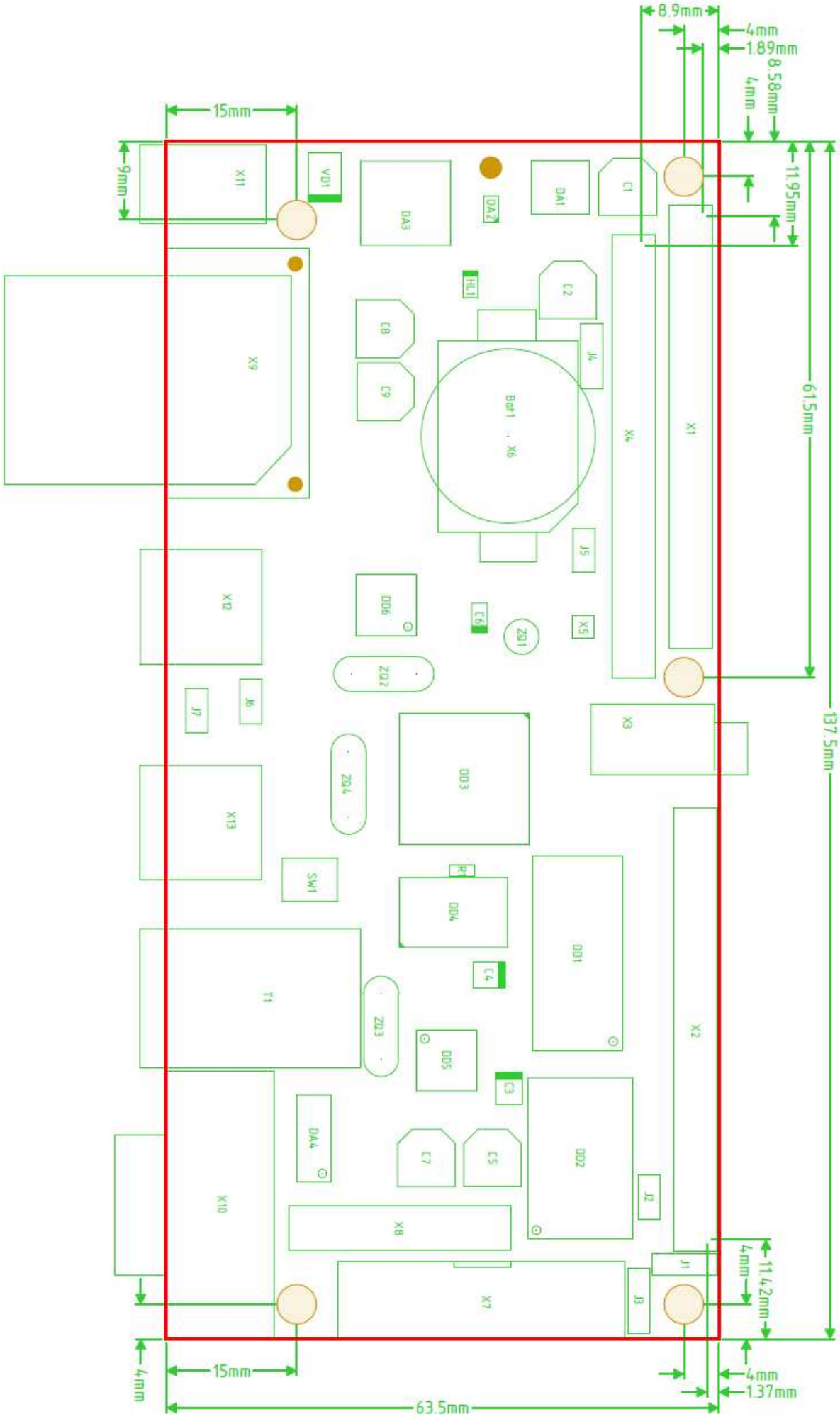
8.4 Выполняем скрипт Enable NandFlash:



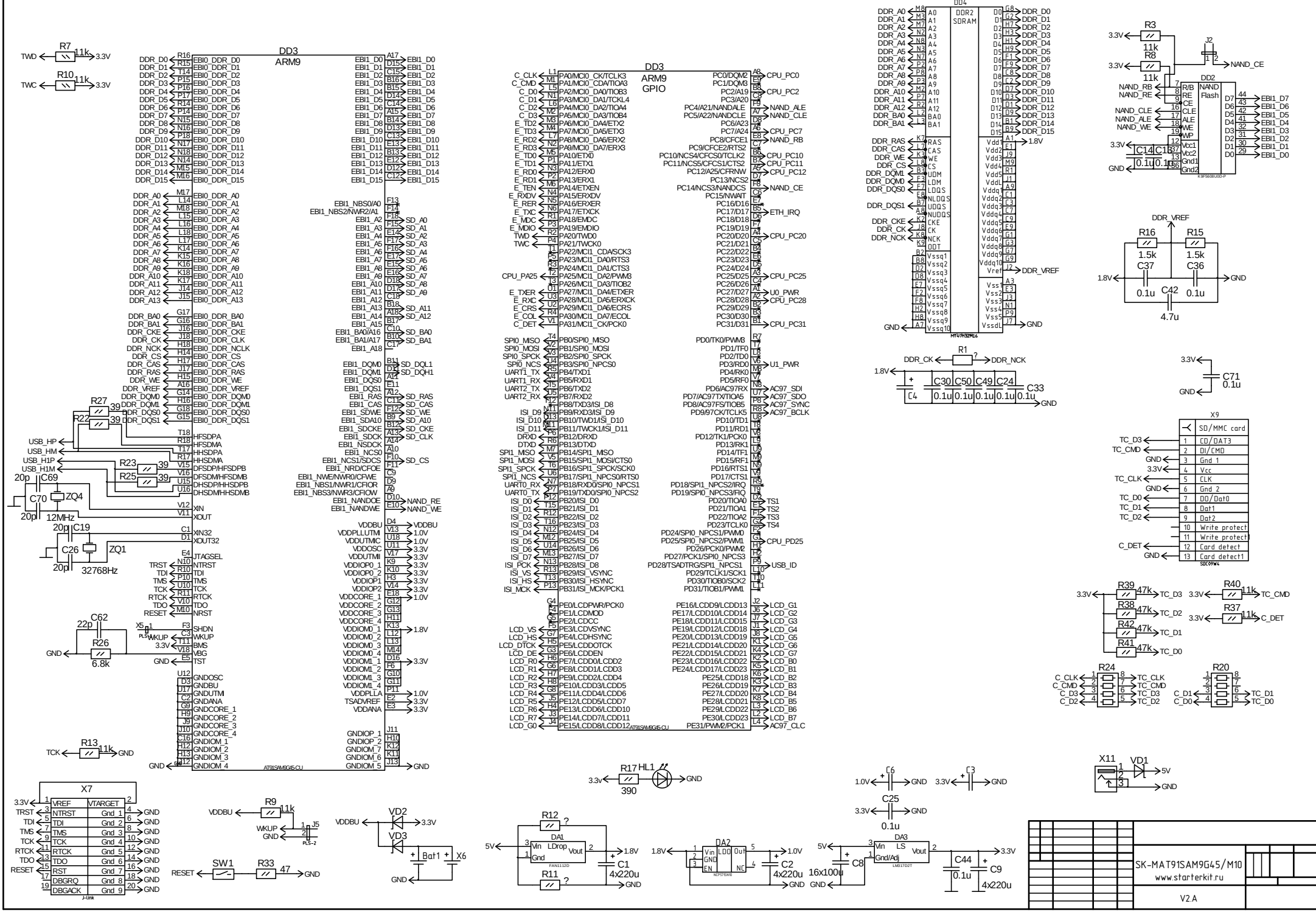
8.5 Открываем файл all_m9g45.bin и записываем (Send File) его по адресу 0:

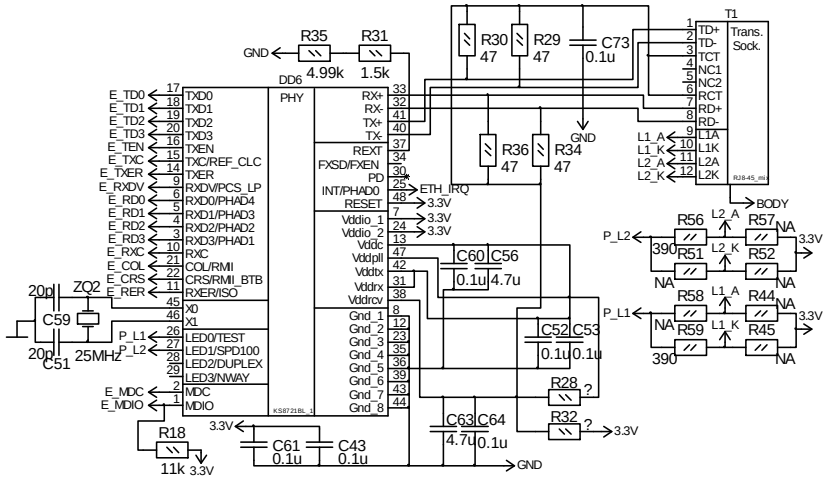


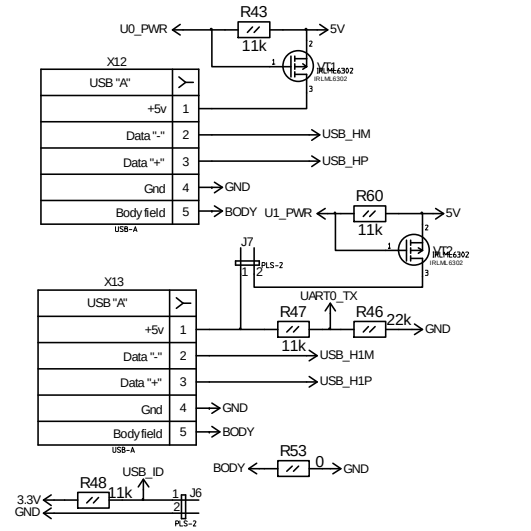
8.6 По окончании записи (примерно через 2-3 минуты) перезагружаемся, если основная корневая ФС была удалена, останавливаем процесс загрузки u-boot, выпоняем “run safe_boot”.

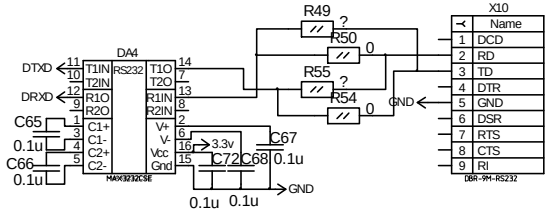


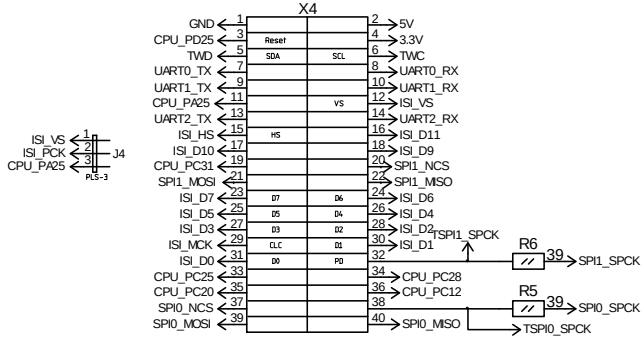
Габаритный чертеж

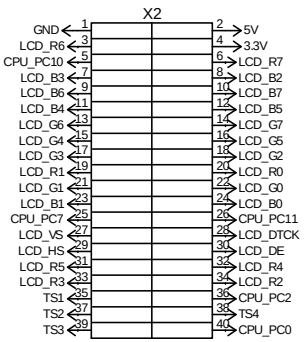

DD3
ARM9
DD3
ARM9
GPIO
DD4
DDR2
SDRAM
R7 11k
R10 11k
TWD
TWC
3.3V
R3 11k
R8 11k
J2
NAND_CE
DD2
NAND
Flash
NAND_RB
NAND_RE
NAND_CLE
NAND_ALE
NAND_WE
R/B
RE
CE
CLE
ALE
WE
WP
Vcc1
Vcc2
Gnd1
Gnd2
C14 0.1u
C15 0.1u
C16 0.1u
DDR_VREF
R16 1.5k
R15 1.5k
C37 0.1u
C36 0.1u
C42 4.7u
1.8V
GND
R27 39
R22 39
R23 39
R25 39
USB_HP
USB_HM
USB_H1P
USB_H1M
HFSDPA
HFSDMA
HHSDPA
HHSDMA
DFSDP/HFSDPB
DFSDMHFSDMB
DHSDP/HHSDPB
DHSDMHHSDMB
20p C69
C70 20p
ZQ4
12MHz
XIN
XOUT
20p C19
C26 20p
ZQ1
32768Hz
XIN32
XOUT32
JTAGSEL
NTRST
TDI
TMS
TCK
RTCK
TDO
RESET
NRST
C62 22p
R26 6.8k
WKUP
BMS
VBG
TST
SHDN
GNDOSC
GNDBU
GNDUTMI
GNDANA
GNDCORE_1
GNDCORE_2
GNDCORE_3
GNDCORE_4
GNDIOM_1
GNDIOM_2
GNDIOM_3
GNDIOM_4
R13 11k
TCK
AT91SAM9G45-CU
VDDBU
VDDPLLUTMI
VDDUTMIC
VDDOSC
VDDUTMI
VDDIOP0_1
VDDIOP0_2
VDDIOP1
VDDIOP2
VDDCORE_1
VDDCORE_2
VDDCORE_3
VDDCORE_4
VDDIOM_1
VDDIOM_2
VDDIOM_3
VDDIOM_4
VDDIOM_1
VDDIOM_2
VDDIOM_3
VDDIOM_4
VDDPLLA
TSADVREF
VDDANA
GNDIOP_1
GNDIOP_2
GNDIOM_7
GNDIOM_6
GNDIOM_5
1.0V
3.3V
1.8V
GND
DDR_D0
DDR_D1
DDR_D2
DDR_D3
DDR_D4
DDR_D5
DDR_D6
DDR_D7
DDR_D8
DDR_D9
DDR_D10
DDR_D11
DDR_D12
DDR_D13
DDR_D14
DDR_D15
EBI0_DDR_D0
EBI0_DDR_D1
EBI0_DDR_D2
EBI0_DDR_D3
EBI0_DDR_D4
EBI0_DDR_D5
EBI0_DDR_D6
EBI0_DDR_D7
EBI0_DDR_D8
EBI0_DDR_D9
EBI0_DDR_D10
EBI0_DDR_D11
EBI0_DDR_D12
EBI0_DDR_D13
EBI0_DDR_D14
EBI0_DDR_D15
DDR_A0
DDR_A1
DDR_A2
DDR_A3
DDR_A4
DDR_A5
DDR_A6
DDR_A7
DDR_A8
DDR_A9
DDR_A10
DDR_A11
DDR_A12
DDR_A13
EBI0_DDR_A0
EBI0_DDR_A1
EBI0_DDR_A2
EBI0_DDR_A3
EBI0_DDR_A4
EBI0_DDR_A5
EBI0_DDR_A6
EBI0_DDR_A7
EBI0_DDR_A8
EBI0_DDR_A9
EBI0_DDR_A10
EBI0_DDR_A11
EBI0_DDR_A12
EBI0_DDR_A13
DDR_BA0
DDR_BA1
DDR_CKE
DDR_CK
DDR_NCK
DDR_CS
DDR_CAS
DDR_RAS
DDR_WE
DDR_VREF
DDR_DQM0
DDR_DQM1
DDR_DQS0
DDR_DQS1
EBI0_DDR_BA0
EBI0_DDR_BA1
EBI0_DDR_CKE
EBI0_DDR_CLK
EBI0_DDR_NCLK
EBI0_DDR_CS
EBI0_DDR_CAS
EBI0_DDR_RAS
EBI0_DDR_WE
EBI0_DDR_VREF
EBI0_DDR_DQM0
EBI0_DDR_DQM1
EBI0_DDR_DQS0
EBI0_DDR_DQS1
EBI1_D0
EBI1_D1
EBI1_D2
EBI1_D3
EBI1_D4
EBI1_D5
EBI1_D6
EBI1_D7
EBI1_D8
EBI1_D9
EBI1_D10
EBI1_D11
EBI1_D12
EBI1_D13
EBI1_D14
EBI1_D15
EBI1_NBS0/A0
EBI1_NBS2/NWR2/A1
EBI1_A2
EBI1_A3
EBI1_A4
EBI1_A5
EBI1_A6
EBI1_A7
EBI1_A8
EBI1_A9
EBI1_A10
EBI1_A11
EBI1_A12
EBI1_A13
EBI1_A14
EBI1_A15
EBI1_BA0/A16
EBI1_BA1/A17
EBI1_A18
EBI1_DQM0
EBI1_DQM1
EBI1_DQS0
EBI1_DQS1
EBI1_RAS
EBI1_CAS
EBI1_SDWE
EBI1_SDA10
EBI1_SDCKE
EBI1_SDCK
EBI1_NSDCK
EBI1_NCS0
EBI1_NCS1/SDCS
EBI1_NRD/CFOE
EBI1_NWE/NWR0/CFWE
EBI1_NBS1/NWR1/CFIOR
EBI1_NBS3/NWR3/CFIOW
EBI1_NANDOE
EBI1_NANDWE
SD_A0
SD_A1
SD_A2
SD_A3
SD_A4
SD_A5
SD_A6
SD_A7
SD_A8
SD_A9
SD_A11
SD_A12
SD_BA0
SD_BA1
SD_DQL1
SD_DQH1
SD_RAS
SD_CAS
SD_WE
SD_A10
SD_CKE
SD_CLK
SD_CS
NAND_RE
NAND_WE
C_CLK
C_CMD
C_D0
C_D1
C_D2
C_D3
E_TD2
E_TD3
E_RD2
E_RD3
E_TD0
E_TD1
E_RD0
E_RD1
E_TEN
E_RXDV
E_RER
E_TXC
E_MDC
E_MDIO
TWD
TWC
CPU_PA25
E_TXER
E_RXC
E_CRS
E_COL
C_DET
PA0/MCI0_CK/TCLK3
PA1/MCI0_CDA/TIOA3
PA2/MCI0_DA0/TIOB3
PA3/MCI0_DA1/TCLK4
PA4/MCI0_DA2/TIOA4
PA5/MCI0_DA3/TIOB4
PA6/MCI0_DA4/ETX2
PA7/MCI0_DA5/ETX3
PA8/MCI0_DA6/ERX2
PA9/MCI0_DA7/ERX3
PA10/ETX0
PA11/ETX1
PA12/ERX0
PA13/ERX1
PA14/ETXEN
PA15/ERXDV
PA16/ERXER
PA17/ETXCK
PA18/EMDC
PA19/EMDIO
PA20/TWD0
PA21/TWCK0
PA22/MCI1_CDA/SCK3
PA23/MCI1_DA0/RTS3
PA24/MCI1_DA1/CTS3
PA25/MCI1_DA2/PWM3
PA26/MCI1_DA3/TIOB2
PA27/MCI1_DA4/ETXER
PA28/MCI1_DA5/ERXCK
PA29/MCI1_DA6/ECRS
PA30/MCI1_DA7/ECOL
PA31/MCI1_CK/PCK0
PB0/SPI0_MISO
PB1/SPI0_MOSI
PB2/SPI0_SPCK
PB3/SPI0_NPCS0
PB4/TXD1
PB5/RXD1
PB6/TXD2
PB7/RXD2
PB8/TXD3/ISI_D8
PB9/RXD3/ISI_D9
PB10/TWD1/ISI_D10
PB11/TWCK1/ISI_D11
PB12/DRXD
PB13/DTXD
PB14/SPI1_MISO
PB15/SPI1_MOSI/CTS0
PB16/SPI1_SPCK/SCK0
PB17/SPI1_NPCS0/RTS0
PB18/RXD0/SPI0_NPCS1
PB19/TXD0/SPI0_NPCS2
PB20/ISI_D0
PB21/ISI_D1
PB22/ISI_D2
PB23/ISI_D3
PB24/ISI_D4
PB25/ISI_D5
PB26/ISI_D6
PB27/ISI_D7
PB28/ISI_D8
PB29/ISI_VSYNC
PB30/ISI_HSYNC
PB31/ISI_MCK/PCK1
PC0/DQM2
PC1/DQM3
PC2/A19
PC3/A20
PC4/A21/NANDALE
PC5/A22/NANDCLE
PC6/A23
PC7/A24
PC8/CFCE1
PC9/CFCE2/RTS2
PC10/NCS4/CFCS0/TCLK2
PC11/NCS5/CFCS1/CTS2
PC12/A25/CFRNW
PC13/NCS2
PC14/NCS3/NANDCS
PC15/NWAIT
PC16/D16
PC17/D17
PC18/D18
PC19/D19
PC20/D20
PC21/D21
PC22/D22
PC23/D23
PC24/D24
PC25/D25
PC26/D26
PC27/D27
PC28/D28
PC29/D29
PC30/D30
PC31/D31
PD0/TK0/PWM8
PD1/TF0
PD2/TD0
PD3/RD0
PD4/RK0
PD5/RF0
PD6/AC97RX
PD7/AC97TX/TIOA5
PD8/AC97FS/TIOB5
PD9/97CK/TCLK5
PD10/TD1
PD11/RD1
PD12/TK1/PCK0
PD13/RK1
PD14/TF1
PD15/RF1
PD16/RTS1
PD17/CTS1
PD18/SPI1_NPCS2/IRQ
PD19/SPI0_NPCS3/IRQ
PD20/TIOA0
PD21/TIOA1
PD22/TIOA2
PD23/TCLK0
PD24/SPI0_NPCS1/PWM0
PD25/SPI0_NPCS2/PWM1
PD26/PCK0/PWM2
PD27/PCK1/SPI0_NPCS3
PD28/TSADTRG/SPI1_NPCS1
PD29/TCLK1/SCK1
PD30/TIOB0/SCK2
PD31/TIOB1/PWM1
PE0/LCDPWR/PCK0
PE1/LCDMOD
PE2/LCDCC
PE3/LCDVSYNC
PE4/LCDHSYNC
PE5/LCDDOTCK
PE6/LCDDEN
PE7/LCDD0/LCDD2
PE8/LCDD1/LCDD3
PE9/LCDD2/LCDD4
PE10/LCDD3/LCDD5
PE11/LCDD4/LCDD6
PE12/LCDD5/LCDD7
PE13/LCDD6/LCDD10
PE14/LCDD7/LCDD11
PE15/LCDD8/LCDD12
PE16/LCDD9/LCDD13
PE17/LCDD10/LCDD14
PE18/LCDD11/LCDD15
PE19/LCDD12/LCDD18
PE20/LCDD13/LCDD19
PE21/LCDD14/LCDD20
PE22/LCDD15/LCDD21
PE23/LCDD16/LCDD22
PE24/LCDD17/LCDD23
PE25/LCDD18
PE26/LCDD19
PE27/LCDD20
PE28/LCDD21
PE29/LCDD22
PE30/LCDD23
PE31/PWM2/PCK1
CPU_PC0
CPU_PC2
NAND_ALE
NAND_CLE
CPU_PC7
NAND_RB
CPU_PC10
CPU_PC11
CPU_PC12
NAND_CE
ETH_IRQ
CPU_PC20
CPU_PC25
U0_PWR
CPU_PC28
CPU_PC31
U1_PWR
AC97_SDI
AC97_SDO
AC97_SYNC
AC97_BCLK
TS1
TS2
TS3
TS4
CPU_PD25
USB_ID
LCD_G1
LCD_G2
LCD_G3
LCD_G4
LCD_G5
LCD_G6
LCD_G7
LCD_B0
LCD_B1
LCD_B2
LCD_B3
LCD_B4
LCD_B5
LCD_B6
LCD_B7
AC97_CLC
DDR_D0
DDR_D1
DDR_D2
DDR_D3
DDR_D4
DDR_D5
DDR_D6
DDR_D7
DDR_D8
DDR_D9
DDR_D10
DDR_D11
DDR_D12
DDR_D13
DDR_D14
DDR_D15
DDR_A0
DDR_A1
DDR_A2
DDR_A3
DDR_A4
DDR_A5
DDR_A6
DDR_A7
DDR_A8
DDR_A9
DDR_A10
DDR_A11
DDR_A12
DDR_BA0
DDR_BA1
DDR_RAS
DDR_CAS
DDR_WE
DDR_CS
DDR_DQM1
DDR_DQM0
DDR_DQS0
DDR_DQS1
DDR_CKE
DDR_CK
DDR_NCK
MT47H32M16
R1 ?
C4
C30 0.1u
C50 0.1u
C49 0.1u
C24 0.1u
C33 0.1u
C71 0.1u
X9
SD/MMC card
CD/DAT3
DI/CMD
Gnd 1
Vcc
CLK
Gnd 2
DO/Dat0
Dat1
Dat2
Write protect
Write protect
Card detect
Card detect1
SDC09W4
TC_D3
TC_CMD
TC_CLK
TC_D0
TC_D1
TC_D2
C_DET
R39 47k
R40 11k
R38 47k
R37 11k
R42 47k
R41 47k
R24
R20
TC_CLK
TC_CMD
TC_D3
TC_D2
TC_D1
TC_D0
C_CLK
C_CMD
C_D3
C_D2
C_D1
C_D0
X11
VD1
5V
R17 390
HL1
R12 ?
DA1
FAN1112D
R11 ?
C1 4x220u
1.8V
DA2
NCP576/NB
C2 4x220u
1.0V
C6
C3
C25 0.1u
DA3
LM1770ZT
C8 16x100u
C44 0.1u
C9 4x220u
3.3V
X7
VREF
VTARGET
NTRST
TDI
TMS
TCK
RTCK
TDO
RST
DBGRQ
DBGACK
Gnd 1
Gnd 2
Gnd 3
Gnd 4
Gnd 5
Gnd 6
Gnd 7
Gnd 8
Gnd 9
J-Link
R9 11k
WKUP
J5
PLS-2
SW1
R33 47
RESET
VD2
VD3
Bat1
X6
SK-MAT91SAM9G45/M10
www.starterkit.ru
V2.A

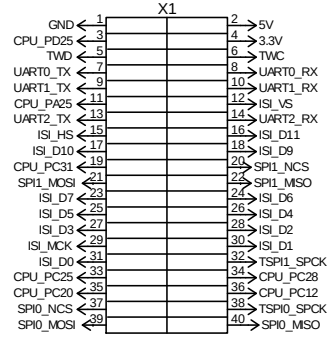

DD6
PHY
KS8721BL
R35 4.99k
R31 1.5k
GND
R30 47
R29 47
C73 0.1u
R36 47
R34 47
T1
Trans. Sock.
RJ8-45_mid
ETH_IRQ
BODY
3.3V
ZQ2
25MHz
20p
C59
C51
C60 0.1u
C56 4.7u
C52 0.1u
C53 0.1u
C61 0.1u
C43 0.1u
C63 4.7u
C64 0.1u
R28 ?
R32 ?
R18 11k
R56 390
R57 NA
R51 NA
R52 NA
R58 NA
R44 NA
R59 390
R45 NA
P_L1
P_L2
L1_A
L1_K
L2_A
L2_K


R43 11k
U0_PWR
5V
VT1 IRLML6302
X12
USB "A"
+5v
Data "-"
Data "+"
Gnd
Body field
USB_HM
USB_HP
GND
BODY
R60 11k
U1_PWR
VT2 IRLML6302
J7
PLS-2
X13
R47 11k
R46 22k
UART0_TX
USB_H1M
USB_H1P
R53 0
USB_ID
R48 11k
J6


DA4
MAX3232CSE
DTXD
DRXD
C65 0.1u
C66 0.1u
C67 0.1u
C72 0.1u
C68 0.1u
R49 ?
R50 0
R55 ?
R54 0
X10
DBR-9M-RS232
DCD
RD
TD
DTR
GND
DSR
RTS
CTS
RI


X4
ISI_VS
ISI_PCK
CPU_PA25
J4
PLS-3
TSPI1_SPCK
R6 39
R5 39
SPI1_SPCK
SPI0_SPCK
TSPI0_SPCK


X2


X1