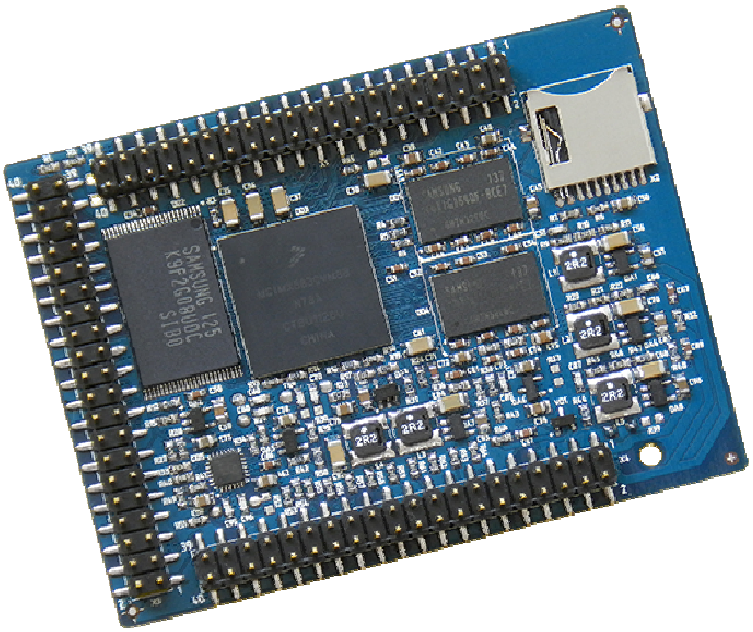
Процессорный модуль SK-iMX50-OEM

Инструкция пользователя при совместном использовании с платой SK-iMX50-MB



SK-iMX50-OEM:

Freescale iMX503 (ARM Cortex-A8 800МГц)
DDR2 256Мбайт
NAND Flash 256Мбайт
100/10M Ethernet
uSD держатель
Разъемы расширения

Модуль проектировался с учетом максимально возможной совместимости с модулем SK-iMX53-OEM, исключение составляют сигналы, функции которых отсутствуют у процессоров iMX50 семейства (SATA, LVDS, CAN, CSI).

SK-iMX50-OEM, возможность прямого подключения:

SK-iMX50-MB – материнская плата для демонстрации возможностей процессорного модуля

SK-iMX50-MB, возможность прямого подключения:

SK-MI0430FT-Plug или аналог – модуль расширения LCD TFT 480x272 4,3”
SK-ATM0700D4-Plug или аналог – модуль расширения LCD TFT 800x480 7”
SK-TFT1024x768TP-Plug или аналог – модуль расширения LCD TFT 1024x768 8”
SK-TFT1024x768-Plug или аналог – модуль расширения LCD TFT 1024x768 8”
EV-ATM5HD-Plug или аналог – модуль расширения LCD TFT 800x480 5”
SK-HDMI-Plug – модуль расширения HDMI выхода
SK-SIMCOM-Plug – модуль расширения GSM/GPS/3G модулей

Общие характеристики

SK-iMX50-OEM:

- Напряжение питания: 5В
- Потребляемый ток до 0,5А
- Габариты 77х57мм

SK-iMX50-MB:

- Напряжение питания: 5-6В (питающее напряжение – центральный штырь разъема), при использовании USB-host 6В максимум
- Потребляемый ток (зависит от подключения внешних модулей) до 2А
- Габариты 124х109мм

1. Назначение джамперов

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

SK-iMX50-OEM:

- J1 – не используется
- J2 – замыкание этого джампера при включении питания конфигурирует процессор для загрузки с uSD карты (подробности ниже)

SK-iMX50-MB:

- J3-J4 позволяет выбирать подключение разъема X24 к микрофонному или линейному входу звукового кодека
- J5-J6 позволяет выбирать подключение разъема X25 к выходу на наушники или линейному выходу звукового кодека
- J8 позволяет использовать питание шины USB

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-ATM0700D4-Plug к разъему X9.

Подключите питание, в терминальной программе появятся подобные сообщения:

```
U-Boot 2009.08 (May 09 2013 - 08:51:01)
CPU:   Freescale i.MX50 family 1.1V at 800 MHz
mx50 pll1: 800MHz
mx50 pll2: 400MHz
mx50 pll3: 216MHz
ipg clock     : 66666666Hz
ipg per clock : 20000000Hz
uart clock    : 24000000Hz
ahb clock     : 133333333Hz
axi_a clock   : 400000000Hz
axi_b clock   : 200000000Hz
weim_clock    : 100000000Hz
ddr clock     : 266666666Hz
esdhc1 clock  : 80000000Hz
esdhc2 clock  : 80000000Hz
esdhc3 clock  : 80000000Hz
esdhc4 clock  : 80000000Hz
GPMI clock    : 24000000Hz
BCH clock     : 24000000Hz
Board: SK-iMX50-OEM module
```

```
    Boot Reason: [WDOG]
    Boot Device: NAND
    I2C:   ready
    DRAM:  256 MB
    NAND:  Manufacturer      : Samsung (0xec)
    Device Code       : 0xda
    Cell Technology   : SLC
    Chip Size         : 256 MiB
    Pages per Block   : 64
    Page Geometry     : 2048+64
    ECC Strength      : 4 bits
    ECC Size          : 512 B
    Data Setup Time   : 20 ns
    Data Hold Time    : 10 ns
    Address Setup Time: 20 ns
    GPMI Sample Delay : 6 ns
    tREA              : Unknown
    tRLOH             : Unknown
    tRHOH             : Unknown
    Description       : K9F2G08U0A
    256 MiB

MMC:   FSL_ESDHC: 0, FSL_ESDHC: 1, FSL_ESDHC: 2
    *** Warning - bad CRC or NAND, using default environment
In:    serial
Out:   serial
Err:   serial
PMIC Mode: SPI
Net:   got MAC address from IIM: 00:00:00:00:00:00
FEC0 [PRIME]
Hit any key to stop autoboot:  3  2  1  0

NAND read: device 0 offset 0x1900000, size 0x600000
 6291456 bytes read: OK
## Booting kernel from Legacy Image at 70800000 ...
   Image Name:   Linux-2.6.35.3
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    2716800 Bytes =  2.6 MB
   Load Address: 70008000
   Entry Point:  70008000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK
Starting kernel ...

Uncompressing Linux... done, booting the kernel.
Linux version 2.6.35.3 (user@imx50-bld) (gcc version 4.3.3 (Sourcery G++ Lite 2009q1-203) ) #293
PREEMPT Thu May 9 13:41:07 UTC 2013
CPU: ARMv7 Processor [412fc085] revision 5 (ARMv7), cr=10c53c7d
CPU: VIPT nonaliasing data cache, VIPT nonaliasing instruction cache
Machine: Freescale MX50 Reference Design Platform
Memory policy: ECC disabled, Data cache writeback
Built 1 zonelists in Zone order, mobility grouping on.  Total pages: 65024
Kernel command line: console=ttymxc0,115200 ubi.mtd=1 root=ubi0:nandfs rw rootfstype=ubifs
noinitrd gpmi:nand
PID hash table entries: 1024 (order: 0, 4096 bytes)
Dentry cache hash table entries: 32768 (order: 5, 131072 bytes)
Inode-cache hash table entries: 16384 (order: 4, 65536 bytes)
Memory: 256MB = 256MB total
Memory: 254356k/254356k available, 7788k reserved, 0K highmem
Virtual kernel memory layout:
    vector  : 0xffff0000 - 0xffff1000   (   4 kB)
    fixmap  : 0xfff00000 - 0xfffe0000   ( 896 kB)
    DMA     : 0xf8e00000 - 0xffe00000   ( 112 MB)
    vmalloc : 0x90800000 - 0xf4000000   (1592 MB)
    lowmem  : 0x80000000 - 0x90000000   ( 256 MB)
    pkmap   : 0x7fe00000 - 0x80000000   (   2 MB)
    modules : 0x7f000000 - 0x7fe00000   (  14 MB)
      .init : 0x80008000 - 0x80029000   ( 132 kB)
      .text : 0x80029000 - 0x804ed000   (4880 kB)
      .data : 0x80506000 - 0x80542220   ( 241 kB)
SLUB: Genslabs=9, HWalign=64, Order=0-3, MinObjects=0, CPUs=1, Nodes=1
Hierarchical RCU implementation.
RCU-based detection of stalled CPUs is disabled.
Verbose stalled-CPUs detection is disabled.
NR_IRQS:336
MXC GPIO hardware
MXC IRQ initialized
You should not call the gpmi_set_parent
MXC_Early serial console at MMIO 0x53fbc000 (options '115200')
bootconsole [ttymxc0] enabled
Console: colour dummy device 80x30
Calibrating delay loop... 799.53 BogoMIPS (lpj=3997696)
```

```
pid_max: default: 32768 minimum: 301
Mount-cache hash table entries: 512
CPU: Testing write buffer coherency: ok
regulator: core version 0.5
regulator: dummy:
NET: Registered protocol family 16
i.MX IRAM pool: 128 KB@0x90840000
CPU is i.MX50 Revision 1.1
Using SDMA I.API
MXC DMA API initialized
IMX usb wakeup probe
IMX usb wakeup probe
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
Advanced Linux Sound Architecture Driver Version 1.0.23.
Switching to clocksource mxc_timer1
NET: Registered protocol family 2
IP route cache hash table entries: 2048 (order: 1, 8192 bytes)
TCP established hash table entries: 8192 (order: 4, 65536 bytes)
TCP bind hash table entries: 8192 (order: 3, 32768 bytes)
TCP: Hash tables configured (established 8192 bind 8192)
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
RPC: Registered udp transport module.
RPC: Registered tcp transport module.
RPC: Registered tcp NFSv4.1 backchannel transport module.
LPMode driver module loaded
Static Power Management for Freescale i.MX5
PM driver module loaded
sdram autogating driver module loaded
Bus freq driver module loaded
DVFS driver module loaded
msgmni has been set to 496
alg: No test for stdrng (krng)
cryptodev: driver loaded.
io scheduler noop registered
io scheduler deadline registered
io scheduler cfq registered (default)
Console: switching to colour frame buffer device 128x48
Serial: MXC Internal UART driver
mxcintuart.0: ttymxc0 at MMIO 0x53fbc000 (irq = 31) is a Freescale i.MX
console [ttymxc0] enabled, bootconsole disabled
console [ttymxc0] enabled, bootconsole disabled
mxcintuart.1: ttymxc1 at MMIO 0x53fc0000 (irq = 32) is a Freescale i.MX
mxcintuart.2: ttymxc2 at MMIO 0x5000c000 (irq = 33) is a Freescale i.MX
loop: module loaded
i.MX GPMI NFC
NFC: Version 2, 8-chip GPMI and BCH
Boot ROM: Version 1, Single-chip boot area, block mark swapping supported
Scanning for NAND Flash chips...
-----------------------------
NAND Flash Device Information
-----------------------------
Manufacturer      : Samsung (0xec)
Device Code       : 0xda
Cell Technology   : SLC
Chip Size         : 256 MiB
Pages per Block   : 64
Page Geometry     : 2048+64
ECC Strength      : 4 bits
ECC Size          : 512 B
Data Setup Time   : 20 ns
Data Hold Time    : 10 ns
Address Setup Time: 20 ns
GPMI Sample Delay : 6 ns
tREA              : Unknown
tRLOH             : Unknown
tRHOH             : Unknown
Description       : K9F2G08U0A
------------
NFC Geometry
------------
ECC Algorithm          : BCH
ECC Strength           : 8
Page Size in Bytes     : 2112
Metadata Size in Bytes : 10
ECC Chunk Size in Bytes: 512
ECC Chunk Count        : 4
```

```
Payload Size in Bytes  : 2048
Auxiliary Size in Bytes: 16
Auxiliary Status Offset: 12
Block Mark Byte Offset : 1999
Block Mark Bit Offset  : 0
NAND device: Manufacturer ID: 0xec, Chip ID: 0xda (Samsung NAND 256MiB 3,3V 8-bit)
-----------------
Boot ROM Geometry
-----------------
Boot Area Count            : 1
Boot Area Size in Bytes    : 33554432 (0x2000000)
Stride Size in Pages       : 64
Search Area Stride Exponent: 2
Scanning device for bad blocks
Boot area protection is enabled.
Creating 2 MTD partitions on "gpmi-nfc-main":
0x000000000000-0x000002000000 : "gpmi-nfc-0-boot"
0x000002000000-0x000010000000 : "gpmi-nfc-general-use"
UBI: attaching mtd1 to ubi0
UBI: physical eraseblock size:   131072 bytes (128 KiB)
UBI: logical eraseblock size:    126976 bytes
UBI: smallest flash I/O unit:    2048
UBI: VID header offset:          2048 (aligned 2048)
UBI: data offset:                4096
UBI: attached mtd1 to ubi0
UBI: MTD device name:            "gpmi-nfc-general-use"
UBI: MTD device size:            224 MiB
UBI: number of good PEBs:        1792
UBI: number of bad PEBs:         0
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:             119
UBI: total number of reserved PEBs: 1673
UBI: number of PEBs reserved for bad PEB handling: 17
UBI: max/mean erase counter: 1/0
UBI: image sequence number: 2036678183
UBI: background thread "ubi_bgt0d" started, PID 971
FEC Ethernet Driver
fec_enet_mii_bus: probed
ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
fsl-ehci fsl-ehci.0: Freescale On-Chip EHCI Host Controller
fsl-ehci fsl-ehci.0: new USB bus registered, assigned bus number 1
fsl-ehci fsl-ehci.0: irq 18, io base 0x53f80000
fsl-ehci fsl-ehci.0: USB 2.0 started, EHCI 1.00
hub 1-0:1.0: USB hub found
hub 1-0:1.0: 1 port detected
fsl-ehci fsl-ehci.1: Freescale On-Chip EHCI Host Controller
fsl-ehci fsl-ehci.1: new USB bus registered, assigned bus number 2
fsl-ehci fsl-ehci.1: irq 14, io base 0x53f80200
fsl-ehci fsl-ehci.1: USB 2.0 started, EHCI 1.00
hub 2-0:1.0: USB hub found
hub 2-0:1.0: 1 port detected
usbcore: registered new interface driver cdc_acm
cdc_acm: v0.26:USB Abstract Control Model driver for USB modems and ISDN adapters
Initializing USB Mass Storage driver...
usbcore: registered new interface driver usb-storage
USB Mass Storage support registered.
usbcore: registered new interface driver usbserial
USB Serial support registered for generic
usbcore: registered new interface driver usbserial_generic
usbserial: USB Serial Driver core
USB Serial support registered for GSM modem (1-port)
usbcore: registered new interface driver option
option: v0.7.2:USB Driver for GSM modems
ARC USBOTG Device Controller driver (1 August 2005)
udc: request mem region for fsl-usb2-udc failed
fsl-usb2-udc: probe of fsl-usb2-udc failed with error -16
mice: PS/2 mouse device common for all mice
spi0.0 supply vcc not found, using dummy regulator
ads7846 spi0.0: touchscreen, irq 306
input: ADS7846 Touchscreen as /devices/platform/spi_gpio.0/spi0.0/input/input0
mxc_rtc mxc_rtc.0: rtc core: registered mxc_rtc as rtc0
i2c /dev entries driver
Linux video capture interface: v2.00
pxp-v4l2 pxp-v4l2: initialized
usbcore: registered new interface driver uvcvideo
USB Video Class driver (v0.1.0)
Driver for 1-wire Dallas network protocol.
MXC WatchDog Driver 2.0
MXC Watchdog # 0 Timer: initial timeout 60 sec
gpu mmu enabled
```

```
mxsdhci: MXC Secure Digital Host Controller Interface driver
mxsdhci: MXC SDHCI Controller Driver.
mmc0: SDHCI detect irq 246 irq 1 INTERNAL DMA
mxsdhci: MXC SDHCI Controller Driver.
mmc1: SDHCI detect irq 303 irq 2 INTERNAL DMA
usbcore: registered new interface driver usbhid
usbhid: USB HID core driver
usbcore: registered new interface driver snd-usb-audio
No device for DAI tlv320aic23
No device for DAI imx-ssi-1-0
No device for DAI imx-ssi-1-1
No device for DAI imx-ssi-2-0
No device for DAI imx-ssi-2-1
AIC23 Audio Codec 0.1
DMA Sound Buffer Allocated: Playback UseIram=1 ext_ram=0 buf->addr=f8002000 buf->area=90842000
size=24576
DMA Sound Buffer Allocated: Capture  UseIram=1 ext_ram=0 buf->addr=f8008000 buf->area=90848000
size=24576
asoc: tlv320aic23 <-> imx-ssi-2-0 mapping ok
ALSA device list:
  #0: imx-3stack (tlv320aic23)
TCP cubic registered
NET: Registered protocol family 17
VFP support v0.3: implementor 41 architecture 3 part 30 variant c rev 2
mxc_rtc mxc_rtc.0: setting system clock to 2013-05-13 13:12:32 UTC (1368450752)
UBIFS: mounted UBI device 0, volume 0, name "nandfs"
UBIFS: file system size:   208367616 bytes (203484 KiB, 198 MiB, 1641 LEBs)
UBIFS: journal size:       10412032 bytes (10168 KiB, 9 MiB, 82 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 0:11.
Freeing init memory: 132K
Starting logging: OK
Initializing random number generator... done.
Starting network...
eth0: Freescale FEC PHY driver [SMSC LAN8710/LAN8720] (mii_bus:phy_addr=0:00, irq=-1)
Starting dropbear sshd: OK
Starting sshd: OK
Starting wi-fi network ...
Error for wireless request "Set Mode" (8B06) :
    SET failed on device wlan0 ; Invalid argument.
Cannot read /proc/net/wireless
Error for wireless request "Set ESSID" (8B1A) :
    SET failed on device wlan0 ; Invalid argument.
ifconfig: SIOCSIFADDR: No such device
PHY: 0:00 - Link is Up - 100/Full

Welcome to SK-iMX50-OEM module!
SK-iMX50-OEM login:
```

Это означает, что система успешно загрузилась и готова к работе.

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-ATM0700D4-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 можно запустить ts_test для демонстрации.

Для настройки часов реального времени, предварительно подключив CR1220 батарею на SK-iMX50-MB, необходимо настроить дату-время и сохранить настройки:

```
# date -s 2012.06.05-15:24:10
Tue Jun  5 15:24:10 MSD 2012
# hwclock -w
```

2.1. Подключение модулей расширения

SK-ATM0700D4-Plug – разъем X9

В штатной поставке ядро сконфигурировано на использование данного модуля расширения – /dev/fb0, в качестве контроллера TP включен ADS7846 (или аналог). Обращаю внимание, на самом модуле расширения необходимо разомкнуть джампер J10 (управляет активностью выходных цепей LVDS десериализатора, подключенных в параллель с RGB шиной данных).

EV-ATM5HD-Plug– разъем X9

Настройка системы для данного модуля в точности совпадает с настройками для модуля SK-ATM0700D4-Plug.

SK-TFT1024x768TP-Plug – разъем X9

В штатной поставке ядро не сконфигурировано на использование данного модуля расширения, в качестве контроллера TP включен ADS7846 (или аналог). Для активации необходимо:

а) в свойствах ядра (скрипт menuconfig.sh) зайти в меню «Device Drivers/ Graphics support/Select of LCD plug» и выбрать «SK-TFT1024x768-Plug – 1024x768»

б) пересобрать (скрипт build.sh) и обновить ядро (или просто загрузить ядро по TFTP)

SK-MI0430FT-Plug – разъем X9

В штатной поставке ядро не сконфигурировано на использование данного модуля расширения, в качестве контроллера TP включен ADS7846 (или аналог). Для активации необходимо:

а) в свойствах ядра (скрипт menuconfig.sh) зайти в меню «Device Drivers/ Graphics support/Select of LCD plug» и выбрать «SK-MI0430FT-Plug – 480x272»

б) пересобрать (скрипт build.sh) и обновить ядро (или просто загрузить ядро по TFTP)

SK-HDMI-Plug – разъем X9

В штатной поставке ядро не сконфигурировано на использование данного модуля расширения. Для активации необходимо:

а) в свойствах ядра (скрипт menuconfig.sh) зайти в меню «Device Drivers/ Graphics support/Select of LCD plug» и выбрать «SK-HDMI-Plug – 1280x720»

б) пересобрать (скрипт build.sh) и обновить ядро (или просто загрузить ядро по TFTP)

3. Состав ОС Linux

Ядро 2.6.35.3, включая драйвера:

- Ethernet
- NAND flash
- USB-host
- USB-gadget
- SD/MMC
- I2C
- ISI
- SPI
- UART
- RTC
- Frame Buffer
- TP ADS7846
- …

4. Способы загрузки и содержимое корневой файловой системы

iMX50X подразумевает различные возможные источники загрузки, на модуле предусмотрено два - NAND flash и uSD карта

Штатная загрузка системы осуществляется с NAND flash, после старта загрузчика u-boot из NAND flash, можно настроить загрузку системы на любой из доступных интерфейсов и носителей: NAND, uSD, Ethernet …

Загрузка с micro SD карт осуществляется при замкнутом джампере J2, карта памяти должна быть подготовлена специальным скриптом (с набором исполняемых файлов) находящимся в виртуальной машине «/home/user/src/rootfs/main_fs/sd_fs/sd_prepare.sh» Внимание! В результате работы скрипта все данные на карте будут утеряны. Основные действия скрипта подготовки uSD карты:

1) Создание разделов: 1 – FAT, 2 – EXT3
2) Форматирование разделов
3) Копирование загрузчика u-boot (копируется в пространство не входящее в состав разделов карты памяти)
4) Копирование образа ядра в FAT раздел, в последствии его будет запускать u-boot
5) Копирование корневой файловой системы в EXT3 раздел карты

В штатной поставке, NAND flash содержит загрузчик, ядро, ядро с интегрированной ФС (для «аварийной» загрузки), корневую ФС.

NAND flash разбита на две части:

1) 32M – для хранения загрузчиков, ядра ситемы и смтемы загрузки «safe mode»
   0 – 0x1000000 – область хранения загрузчика u-boot
   0x1000000 – 0x1100000 – область хранения переменных окружения u-boot
   0x1100000 – 0x1900000 – область хранения системы аварийной загрузки
   0x1900000 – 0x2000000 – область хранения ядра Linux
2) 220M – раздел UBI файловой системы, используется в качестве корневой файловой 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.

Восстановить систему к первоначальному состоянию можно с помощью micSD карты подготовленной скриптом «/home/user/src/rootfs/safe_fs/sd_safe/sd_safe_prepare.sh», загрузившись, система отформатирует NAND flash и скопирует на нее все необходимые файлы.

5. Виртуальная машина VMware

Для сборки ядра и корневой ФС используется виртуальная машина VMware с установленной ОС Ubuntu, 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, Samba.

Разархивируйте файл “SK-iMX50-OEM_linux_build_machine.rar“, 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 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 системным временем, особенно при копировании новых файлов (имеющих более позднюю дату создания относительно системы) для сборки.

По умолчанию, в системе присутствует один пользователь: **логин user, пароль 123456**

Суперпользователя root в виртуальной машине нет, для действий с его привилегиями необходимо пользоваться командами su или sudo.

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:

- rootfs/main_fs - пакет сборки корневой файловой системы
- rootfs/safe_fs - пакет сборки корневой файловой системы для «Safe mode»
- kernel – ядро linux, скрипты сборки внутри
- u-boot - загрузчик

В корневом каталоге ядра присутствует два скрипта:

build.sh – собирает ядро и копирует файл в папку TFTP сервера

menuconfig.sh – запускает конфигурационное меню ядра

5.1. Примеры

Обновление ядра Linux, для этого необходимо:

- запустить виртуальную машину
- запустить скрипт /home/user/src/kernel/linux-X.X.XX-sk/build.sh
- включить/перезагрузить плату с подключенным Ethernet (разъем T1) и RS232 кабелями
- прервать в u-boot процесс загрузки нажатием любой клавиши
- выполнить “run system_update”

Загрузка ядра Linux с TFTP сервера, для этого необходимо:

- запустить виртуальную машину
- включаем/перезагружаем плату с подключенным Ethernet (разъем T1) и RS232 кабелями
- прервать в u-boot процесс загрузки нажатием любой клавиши
- выполнить “run tftp_boot”

Обновление u-boot, для этого необходимо:

- запустить виртуальную машину
- включить/перезагрузить плату с подключенным Ethernet (разъем T1) и RS232 кабелями
- прервать в u-boot процесс загрузки нажатием любой клавиши
- выполнить “run safe_boot”
- после загрузки системы, выполняем ”uboot_update”

Обновление корневой ФС, для этого необходимо:

- запустить виртуальную машину
- включить/перезагрузить плату с подключенным Ethernet (разъем T1) и RS232 кабелями

- прервать в u-boot процесс загрузки нажатием любой клавиши
- выполнить “run safe_boot”
- после загрузки системы, выполнить ”rootfs_update_tftp”

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 MfgTool.

Поскольку внешняя DDR2 (или любая другая память не инициализирована), первое запускаемое приложение должно быть загрузчиком. Это приложение (загрузчик u-boot) 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.

Загрузчик u-boot обладает обширными возможностями, например, он умеет копировать файлы по TFTP, SD или SATA, поддерживает целый набор команд и режимов.

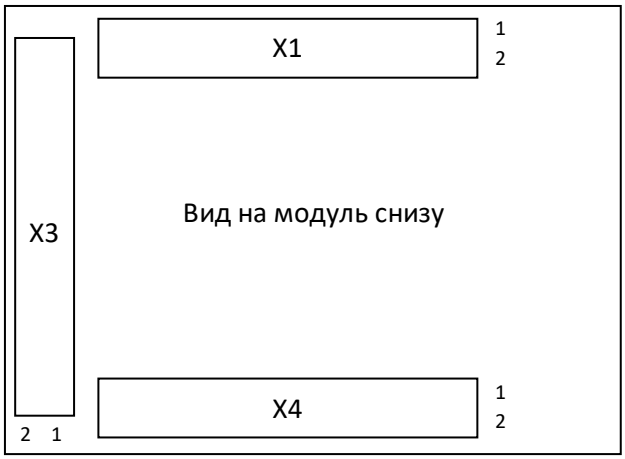
В переменных окружения u-boot есть команда запуска, в которой указано, по какому адресу NAND flash следует прочитать образ ядра, куда этот образ памяти записать и по какому адресу запустить.

Перед запуском ядра Linux, проверяется контрольная сумма собственного архива (в случае safe загрузки, ядро включает в себя еще корневую ФС), иллюстрация:

```
NAND read: device 0 offset 0x1900000, size 0x600000
 6291456 bytes read: OK
## Booting kernel from Legacy Image at 70800000 ...
   Image Name:   Linux-2.6.35.3
   Image Type:   ARM Linux Kernel Image (uncompressed)
   Data Size:    2716800 Bytes =  2.6 MB
   Load Address: 70008000
   Entry Point:  70008000
   Verifying Checksum ... OK
   Loading Kernel Image ... OK
OK
Starting kernel ...
```

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

7. Назначение контактов модуля SK-iMX50-OEM



Ниже перечислены названия выводов процессора, соответствие можно уточнить в «Reference Manual». Жирным шрифтом выделена функция используемая для данного вывода в BSP Linux.

Желтым фоном выделены выводы, назначение которых радикально отличается от назначения в модуле SK-iMX53-OEM.

7.1 Разъем X1

N	Наименование цепи	Дополнительные функции / примечание
1	GND	0B
2	LCD_DTCK	EPDC_D0/GPIO3_0/WEIMV2_D0/ELCDIF_RS/**ELCDIF_DCLK**/USBPHY2_D0
3	LCD_B0	EPDC_SDCE5/GPIO4_30/**ELCDIF_D0**
4	LCD_B1	EPDC_SDCE4/GPIO4_29/**ELCDIF_D1**/USBPHY2_VCTL3
5	LCD_B2	EPDC_SDCE3/GPIO4_28/**ELCDIF_D2**/USBPHY2_VCTL2
6	LCD_B3	EPDC_SDCE2/GPIO4_27/**ELCDIF_D3**/USBPHY2_VCTL1
7	LCD_B4	EPDC_SDCE1/GPIO4_26/**ELCDIF_D4**/USBPHY2_VCTL0
8	LCD_B5	EPDC_SDCE0/GPIO4_25/**ELCDIF_D5**/USBPHY1_VCTL3
9	LCD_B6	EPDC_BDR1/GPIO4_24/**ELCDIF_D6**/USBPHY1_VCTL2
10	LCD_B7	EPDC_BDR0/GPIO4_23/**ELCDIF_D7**/USBPHY1_VCTL1
11	LCD_G0	EPDC_SDLE/GPIO3_24/WEIMV2_D24/**ELCDIF_D8**/AUD5_RXC/USBPHY2_RXERROR
12	LCD_G1	EPDC_SDCLKN/GPIO3_25/WEIMV2_D25/**ELCDIF_D9**/AUD5_RXFS/USBPHY2_SIECLK
13	LCD_G2	EPDC_SDSHR/GPIO3_26/WEIMV2_D26/**ELCDIF_D10**/AUD4_TXD/USBPHY2_LINESTATE0
14	LCD_G3	EPDC_PWRCOM/GPIO3_27/WEIMV2_D27/**ELCDIF_D11**/AUD4_TXC/USBPHY2_LINESTATE1
15	LCD_G4	EPDC_PWRSTAT/GPIO3_28/WEIMV2_D28/**ELCDIF_D12**/AUD4_TXFS/USBPHY2_VBUSVALID
16	LCD_G5	EPDC_PWRCTRL0/GPIO3_29/WEIMV2_D29/**ELCDIF_D13**/AUD4_RXD/USBPHY2_AVALID
17	LCD_G6	EPDC_PWRCTRL1/GPIO3_30/WEIMV2_D30/**ELCDIF_D14**/AUD4_RXC/USBPHY1_ONBIST
18	LCD_G7	EPDC_PWRCTRL2/GPIO3_31/WEIMV2_D31/**ELCDIF_D15**/AUD4_RXFS/USBPHY2_ONBIST
19	LCD_R0	EPDC_GDCLK/GPIO3_16/WEIMV2_D16/**ELCDIF_D16**/AUD6_TXFS/USBPHY2_BISTOK
20	LCD_R1	EPDC_GDSP/GPIO3_17/WEIMV2_D17/**ELCDIF_D17**/AUD6_RXD/USBPHY2_BVALID
21	LCD_R2	EPDC_GDOE/GPIO3_18/WEIMV2_D18/**ELCDIF_D18**/AUD6_RXC/USBPHY2_ENDSESSION

22	LCD_R3	EPDC_GDRL/GPIO3_19/WEIMV2_D19/**ELCDIF_D19**/AUD6_RXFS/USBPHY2_IDDIG
23	LCD_R4	EPDC_SDCLK/GPIO3_20/WEIMV2_D20/**ELCDIF_D20**/AUD5_TXD/USBPHY2_HOSTDNT
24	LCD_R5	EPDC_SDOEZ/GPIO3_21/WEIMV2_D21/**ELCDIF_D21**/AUD5_TXC/USBPHY2_TXRDY
25	LCD_R6	EDPC_SDOED/GPIO3_22/WEIMV2_D22/**ELCDIF_D22**/AUD5_TXFS/USBPHY2_RXVALID
26	LCD_R7	EPDC_SDOE/GPIO3_23/WEIMV2_D23/**ELCDIF_D23**/AUD5_RXD/USBPHY2_RXACTIVE
27	LCD_DE	EPDC_D1/GPIO3_1/WEIMV2_D1/ELCDIF_CS/**ELCDIF_EN**/USBPHY2_D1
28	LCD_VS	EPDC_D2/GPIO3_2/WEIMV2_D2/ELCDIF_WR_RWN/**ELCDIF_VSYNC**/USBPHY2_D2
29	ECSPI1_NCS	**ECSPI1_SS0**/GPIO4_15/CSPI_SS3/ECSPI_SS3/UART4_CTS/EPDC_SDCE9/WEIMV2_D11
30	LCD_HS	EPDC_D3/GPIO3_3/WEIMV2_D3/ELCDIF_RD_E/**ELCDIF_HSYNC**/USBPHY2_D3
31	ECSPI1_MIS	**ECSPI1_MISO**/GPIO4_14/CSPI_SS2/ECSPI2_SS2/UART4_RTS/EPDC_SDCE8/WEIMV2_D10
32	ECSPI1_CLK	**ECSPI1_SCLK**/GPIO4_12/CSPI_RDY/ECSPI2_RDY/UART3_RTS/EPDC_SDCE6/WEIMV2_D8
33	OWIRE	OWIRE_LINE/GPIO6_26/USBH1_OC/CCM_SSI_EXT1_CLK/EPDC_PWRIRQ/GPT_CMPOUT3/OBSRV_INT_OUT4/JTAG_ACT
34	ECSPI1_MOS	**ECSPI1_MOSI**/GPIO4_13/CSPI_SS1/ECSPI2_SS1/UART3_CTS/EPDC_SDCE7/WEIMV_D9
35	GP1_27	EIM_CRE/WEIMV2_CRE/GPIO1_27
36	PWM1	PWM1/PWM1_PWMO/GPIO6_24/USBOTG_OC/GPT_CMPOUT1/OBSRV_INT_OUT2/FAIL
37	I2C1_SCL	I2C1_SCL/GPIO6_18/UART2_TXD_MCU
38	I2C1_SDA	I2C1_SDA/GPIO6_19/UART2_RXD_MCU
39	GND	0B
40	PWM2	PWM2/PWM2_PWMO/GPIO6_25/USBOTG_PWR/DCDC_PWM/GPT_CMPOUT2/OBSRV_INT_OUT3/ANY_PU_PST

7.2 Разъем X3

N	Наименование цепи	Дополнительные функции / примечание
1	USB_OTG_ID	USB OTG порт ID
2	GNG	0B
3	USB_OTG_D	USB OTG порт DN
4	USB_H1_DN	USB Host порт DN
5	USB_OTG_D	USB OTG порт DP
6	USB_H1_DP	USB Host порт DP
7	USB_OTG_V	USB OTG порт VBUS
8	GP6_15	UART3_RXD/GPIO6_15/ESDHC1_D5/ESDHC4_D1/ESDHC2_CD/WEIMV2_D13/USBPHY2_D15
9	SD2_CMD	**SD2_CMD**/ESDHC2_CMD/GPIO5_7/MSHC_BS
10	SD2_CLK	**SD2_CLK**/ESDHC2_CLK/GPIO5_6/MSHC_SCLK
11	SD2_DATA0	**SD2_DATA0**/ESDHC2_D0/GPIO5_8/MSHC_D0/KPP_COL4
12	SD2_DATA2	**SD2_DATA1**/ESDHC2_D1/GPIO5_9/MSHC_D1/KPP_ROW4
13	SD2_DATA1	**SD2_DATA2**/ESDHC2_D2/GPIO5_10/MSHC_D2/KPP_COL5
14	SD2_DATA3	**SD2_DATA3**/ESDHC2_D3/GPIO5_11/MSHC_D3/KPP_ROW5
15	CSPI1_CLK	CSPI_SCLK/GPIO4_8 (в модуле SK-iMX53-OEM этот пин содержит функцию SATA периферии)
16	CSPI1_MISO	CSPI_MISO/GPIO4_10 (в модуле SK-iMX53-OEM этот пин содержит функцию SATA периферии)
17	CSPI1_MOSI	CSPI_MOSI/GPIO4_9 (в модуле SK-iMX53-OEM этот пин содержит функцию SATA периферии)
18	CSPI1_SS0	CSPI_SS0/GPIO4_11 (в модуле SK-iMX53-OEM этот пин содержит функцию SATA периферии)
19	ETH_CP	«Средняя точка» Ethernet трансформатора, см. схему SK-iMX50-MB

20	GND	0B
21	ETH_RXN	Вывод Ethernet PHY RXM
22	ETH_TXN	Вывод Ethernet PHY TXM
23	ETH_RXP	Вывод Ethernet PHY RXP
24	ETH_TXP	Вывод Ethernet PHY TXM
25	AUD_TXC	SD2_CD/ESDHC2_CD/GPIO5_17/**AUD_TXC**/WEIMV2_D5
26	AUDIO_MCL	EPITO/GPIO6_27/USBH1_PWR/**CCM_SSI_EXT2_CLK**/DPLLIP_TOG_EN/GPY_CLKIN/DE_B
27	AUD_TXFS	SD2_DATA7/ESDHC2_D7/GPIO5_15/**AUD_TXFS**/KPP_ROW7/WEIMV2_D3
28	AUD_TXD	SD2_WP/ESDHC2_WP/GPIO5_16/**AUD_TXD**/WEIMV2_D4
29	AUD_RXD	SD2_DATA6/ESDHC2_D6/GPIO5_14/**AUD_RXD**/KPP_COL7/WEIMV2_D2
30	UART5_TX	UART1_CTS/GPIO6_8/UART5_TXD/ESDHC4_D4/ESDHC4_CMD/USBPHY2_D8
31	UART5_RX	UART1_RTS/GPIO6_9/UART5_RXD/ESDHC4_D5/ESDHC4_CLK/USBPHY2_D9
32	UART2_TX	UART2_TXD/GPIO6_10/ESDHC4_D6/ESDHC4_D4/USBPHY2_D10
33	UART2_RX	UART2_RXD/GPIO6_11/ESHDC4_D7/ESDHC4_D5/USBPHY2_D11
34	UART1_TX	UART1_TXD/GPIO6_6/USBPHY1_D4
35	UART1_RX	UART1_RXD/GPIO6_7/USBPHY1_D15
36	I2C2_SCL	**I2C2_SCL**/GPIO6_20/UART2_CTS
37	I2C2_SDA	**I2C2_SDA**/GPIO6_21/UART2_RTS
38	I2C3_SDA	I2C3_SDA/GPIO6_23/FEC_MDIO/PWRFIL_INT/ALARM_DEB/GPT_CAPIN2/OBSRV_INT_OUT1/USBOTG_PWR
39	I2C3_SCL	I2C3_SCL/GPIO6_22/FEC_MDC/GPC_PMIC_RDY/GPT_CAPIN1/OBSRV_INT_OUT0/USBOTG_OC
40	GND	0B

7.3 Разъем X4

N	Наименование цепи	Дополнительные функции / примечание
1	+5V	Питающее напряжение 5В
2	+5V	Питающее напряжение 5В
3	+5V	Питающее напряжение 5В
4	+5V	Питающее напряжение 5В
5	GND	0B
6	GND	0B
7	EIM_DA0	EIM_DA0/WEIMV2_A0/GPIO1_0/KPP_COL4
8	EIM_DA8	EIM_DA8/WEIMV2_A8/GPIO1_8/RAWNAND_CLE
9	EIM_DA1	EIM_DA1/WEIMV2_A1/GPIO1_1/KPP_ROW4
10	EIM_DA9	EIM_DA9/WEIMV2_A9/GPIO1_9/RAWNAND_ALE
11	EIM_DA2	EIM_DA2/WEIMV2_A2/GPIO1_2/KPP_COL5
12	EIM_DA10	EIM_DA10/WEIMV2_A10/GPIO1_10/RAWNAND_CEN0
13	EIM_DA3	EIM_DA3/WEIMV2_A3/GPIO1_3/KPP_ROW5
14	EIM_DA11	EIM_DA11/WEIMV2_A11/GPIO1_11/RAWNAND_CEN1
15	EIM_DA4	EIM_DA4/WEIMV2_A4/GPIO1_4/KPP_COL6
16	EIM_DA12	EIM_DA12/WEIMV2_A12/GPIO1_12/RAWNAND_CEN2/EPDC_SDCE6
17	EIM_DA5	EIM_DA5/WEIMV2_A5/GPIO1_5/KPP_ROW6
18	EIM_DA13	EIM_DA13/WEIMV2_A13/GPIO1_13/RAWNAND_CEN3/EPDC_SDCE7

19	EIM_DA6	EIM_DA6/WEIMV2_A6/GPIO1_6/KPP_COL7
20	EIM_DA14	EIM_DA14/WEIMV2_A14/GPIO1_14/RAWNAND_RDY0/EPDC_SDCE8
21	EIM_DA7	EIM_DA7/WEIMV2_A7/GPIO1_7/KPP_ROW7
22	EIM_DA15	EIM_DA15/WEIMV2_A15/GPIO1_15/RAWNAND_DQS/EPDC_SDCE9
23	EIM_OE	EIM_OE/WEIMV2_OE/GPIO1_24
24	EIM_WAIT	EIM_WAIT/WEIMV2_WAIT/GPIO1_21/WEIMV2_DCLK_B
25	EIM_WE	EIM_RW/WEIMV2_RW/GPIO1_25
26	EIM_BCLK	EIM_BCLK/WEIMV_BCLK/GPIO1_22
27	EIM_EB0	EIM_EB0/WEIMV2_EB0/GPIO1_19
28	EIM_LBA	EIM_LBA/WEIMV2_LBA/GPIO1_26
29	EIM_EB1	EIM_EB1/WEIMV2_EB1/GPIO1_20
30	EIM_CS0	EIM_CS0/WEIMV2_CS0/GPIO1_18
31	RESET	Сигнал сброса для процессора
32	EIM_CS1	EIM_CS1/WEIMV2_CS1/GPIO1_17
33	P_L1	Управление светодиодной индикацией Ethernet PHY. Внимание!!! Этот сигнал участвует в конфигурировании микросхемы Ethernet PHY, необходимо подключать его аналогично включению по схеме SK-iMX50-MB.
34	ECSPI2_NCS	ECSPI2_SS0/GPIO4_19/ELCDIF_CS/ECSPI1_SS3/UART5_RXD/ELCDIF_HSYNC/RAWNAND_CEN7/WEIMV2_D11 В модуле SK-iMX53-OEM этот пин содержит функцию «Аналоговый выход RED»
35	P_L2	Управление светодиодной индикацией Ethernet PHY. Внимание!!! Этот сигнал участвует в конфигурировании микросхемы Ethernet PHY, необходимо подключать его аналогично включению по схеме SK-iMX50-MB.
36	ECSPI2_MOSI	ECSPI2_MOSI/GPIO4_17/ELCDIF_RD_E/ECSPI1_SS1/UART5_CTS/ELCDIF_EN/RAWNAND_CEN5/WEIMV2_D9 В модуле SK-iMX53-OEM этот пин содержит функцию «Аналоговый выход GREEN»
37	VBAT	Автономное питание RTC части, 2,0-5В В модуле SK-iMX53-OEM этот пин содержит функцию «GPIO_7_11, SPDIF IN1, I2C3 SDA …»
38	ECSPI2_MISO	ECSPI2_MISO/GPIO4_18/ELCDIF_RS/ECSPI1_SS2/UART5_TXD/ELCDIF_VSYNC/RAWNAND_CEN6/WEIMV2_D10 В модуле SK-iMX53-OEM этот пин содержит функцию «Аналоговый выход BLUE»
39	ECSPI2_CLK	ECSPI2_SCLK/GPIO4_16/ELCDIF_WR_RWN/ECSPI1_RDY/UART5_RTS/ELCDIF_DCLK/RAWNAND_CEN4/WEIMV2_D8
40	GND	0В

8. Дополнительные материалы

К модулю прилагаются материалы: габаритный чертеж модуля (в формате DXF), структурная схема модуля, схема электрическая принципиальная материнской платы SK-iMX50-MB, проектные файлы материнской платы SK-iMX50-MB (схема, PCB файл печатной платы).