SAMA5D4 PLATFORM USER GUIDE

1 KEYPAD

Keypad is a key input device, and driver report key event from kernel input subsystem, and user can test keypad by tool *getevent*.

- To check keypad input device if succeed register input subsystem and get corresponding handler:

```
# cat /proc/bus/input/device
I: Bus=0019 Vendor=0001 Product=0001 Version=0100
N: Name="gpio_keys"
P: Phys=gpio-keys/input0
S: Sysfs=/devices/gpio_keys/input/input1
U: Uniq=
H: Handlers=event1
B: PROP=0
B: EV=3
B: KEY=1ff 0 0 0 0 0 0 0 0
```

- To get keypad reporting keycode:

```
# getevent -l /dev/input/event1
EV_KEY       BTN_7                DOWN
EV_SYN       SYN_REPORT           00000000
EV_KEY       BTN_7                UP
EV_SYN       SYN_REPORT           00000000
EV_KEY       BTN_4                DOWN
EV_SYN       SYN_REPORT           00000000
EV_KEY       BTN_4                UP
EV_SYN       SYN_REPORT           00000000
```

2 TOUCHSCREEN

Touchscreen is a touch input device, and driver report touchscreen event from kernel input subsystem, and user can test touchscreen by *getevent* tool, but first user should run *ts_calibrate* to calibrate the touchscreen.

- To check touchscreen input device if succeed register input subsystem and get corresponding handler:

```
# cat /proc/bus/input/device
I: Bus=0000 Vendor=0000 Product=0000 Version=0000
```

N: Name="ADS7846 Touchscreen"
P: Phys=spi32766.0/input0
S:Sysfs=/devices/ahb/ahb:apb/f8010000.spi/spi_master/spi32766/spi32766.0
/input/input0
U: Uniq=
H: Handlers=event0
B: PROP=0
B: EV=b
B: KEY=400 0 0 0 0 0 0 0 0 0 0
B: ABS=1000003

- To get touchscreen reporting keycode :

```
# getevent -l /dev/input/event0
EV_KEY      BTN_TOUCH         DOWN
EV_ABS      ABS_X             00000509
EV_ABS      ABS_Y             000007f0
EV_ABS      ABS_PRESSURE       000000ed
EV_SYN      SYN_REPORT        00000000
EV_KEY      BTN_TOUCH         UP
EV_ABS      ABS_PRESSURE       00000000
EV_SYN      SYN_REPORT        00000000
```

- Other touchscreen test tools are build into ts_lib, such as *ts_test, ts_print_raw.*

3 USB MOUSE

USB mouse is a HID input device, and driver report real event from kernel input subsystem, and user can test mouse by tool *getevent* .

- To check mouse input device if succeed register input subsystem and get corresponding handler:

```
# cat /proc/bus/input/device
I: Bus=0003 Vendor=046d Product=c077 Version=0111
N: Name="Logitech USB Optical Mouse"
P: Phys=usb-at91-2/input0
S:Sysfs=/devices/ahb/500000.ohci/usb2/2-2/2-
2:1.0/0003:046D:C077.0001/input/input2
U: Uniq=
```

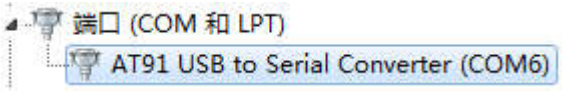
H: Handlers=event2
B: PROP=0
B: EV=17
B: KEY=ff0000 0 0 0 0 0 0 0 0
B: REL=143
B: MSC=10

- To get USB mouse reporting keycode:

```
# getevent -l /dev/input/event2
EV_REL      REL_X           00000001
EV_REL      REL_Y           ffffffff
EV_SYN      SYN_REPORT      00000000
EV_REL      REL_X           00000001
EV_REL      REL_Y           ffffffff
EV_SYN      SYN_REPORT      00000000
```

4 USB DEVICE

USB device port is reserved for system upgrade by USB.
Chip will enter RomBOOT mode, when checked no any bootable media device. If chip succeed entered RomBOOT mode, it will print out "RomBOOT" to console, then connect board and PC with USB cable, PC Device Manager will show out the AT91 USB to serial node, screen cut as below:



Note:User should have installed Atmel PC tool *sam-ba_2.15*, first time connect to RomBOOT mode board, PC will auto install driver for AT91 USB to Serial, but user must manual install the driver again from sam-ba_2.15's install path(for example C:\Program Files (x86)\Atmel\sam-ba_2.15\drv), else will result in USB upgrade failure.

When PC detect AT91 USB to serial, user can run the upgrade script *burn_nandflash.bat* in the upgrade package, then it will show out logfile when upgrade finished.

5 USB STORAGE

- Check if system detected USB disk:

```
# ls /dev/sd*
/dev/sda   /dev/sda1  /dev/sda2
```

If one USB disk have more partitions, node named as *sda1 sda2 ...*

- Another method to check detected USB disk:

```
# cat /proc/partitions |grep sd
 179        0    7703552 sda
 179        1    4998976 sda1
 179        2    2703551 sda2
```

- Manual mount USB disk shell command:

```
# mount /dev/sda1 /mnt
```

6 NAND FLASH

Platform System can bootup from SD or Nand flash, but different bootup method show user different nand partition table.

- SD bootup mode, such as SD card upgrade mode, user have both write and read permission to all partitions.

```
# cat /proc/mtd
mtd0: 00040000 00040000 "at91bootstrap"
mtd1: 00080000 00040000 "bootloader"
mtd2: 000c0000 00040000 "bootloader env"
mtd3: 00080000 00040000 "device tree"
mtd4: 00600000 00040000 "kernel"
mtd5: 1f800000 00040000 "rootfs"
```

- Nand bootup mode, when SD upgrade finished, plug out SD card, then system will bootup form nand flash; when system bootup from nand flash, *env, env_redundent, spare, dtb,rootfs* partition are writeable, and the other partitions are read only.

```
# cat /proc/mtd
mtd0: 00040000 00040000 "bootstrap"
mtd1: 00080000 00040000 "uboot"
mtd2: 00040000 00040000 "env"
mtd3: 00040000 00040000 "env_redundent"
mtd4: 00040000 00040000 "spare"
mtd5: 00080000 00040000 "dtb"
mtd6: 00600000 00040000 "kernel"
mtd7: 1f800000 00040000 "rootfs"
```

mtd8: 1dd98000 0003e000 "rootfs"

7 SPI FLASH

Because of pin conflict between SPI-1 and HSMC, so compile the two device drivers as external module, and do not insert one driver module when have inserted another one.

- Insert SPI-1 bus controller driver(assume the driver module is located in SD card):

```
# insmod /mnt/mmcblk0p1/spi-atmel-1.ko
```

- Check if spi flash device registered MTD succeed:

```
# cat /proc/mtd |grep spi
mtd6: 00200000 00001000 "spi32763.0"
```

- Write SPI Flash:

Erase SPI Flash device first

```
# flash_erase /dev/mtd6 0 0
```

Write binary file(assume FPGA.bin file is located in SD card)

```
# cat /mnt/mmcblk0p1/FPGA.bin >/dev/mtd6
```

8 SD CARD

- Check if system detected SD card:

```
# ls /dev/mmcblk*
/dev/mmcblk0    /dev/mmcblk0p1  /dev/mmcblk0p2
```

If one SD card have more partitions, node named as *mmcblk0p1 mmcblk0p2 ...*

- Another method to check detected card:

```
# cat /proc/partitions |grep mmc
 179        0    7703552 mmcblk0
 179        1    4998976 mmcblk0p1
 179        2    2703551 mmcblk0p2
```

- Manual mount SD shell command:

```
# mount /dev/mmcblk0p1 /mnt
```

9 ETHERNET

- Check matched PHY ID:

 # cat /sys/class/net/eth0/f8020000.etherne\:00/phy_id
 # cat /sys/class/net/eth0/f8020000.etherne\:00/uevent

- Ethernet IP configure:

 # ifconfig eth0 192.168.1.7 up

- Connection check, ping net gateway(assume gateway is 192.168.1.1):

 # ping 192.168.1.1

- TFTP download , run tftpd32 server on PC(192.168.1.3), then run shell to download file on platform.

 # tftp -gr file 192.168.1.3

- Evaluate Ethernet.

Firstly, Install iperf on Ubuntu PC host(192.168.1.6).
ubuntu-host# sudo apt-get install iperf
Secondly, start iperf server on Ubuntu PC host.
ubuntu-host# iperf -s
At last run iperf client on the target below:
 # iperf -c 192.168.1.6 -t 20 -i 1

```
------------------------------------------------------------
Client connecting to 192.168.1.6, TCP port 5001
TCP window size: 43.8 KByte (default)
------------------------------------------------------------
[  3] local 192.168.1.7 port 57248 connected with 192.168.1.6 port 5001
[ ID] Interval       Transfer     Bandwidth
[  3]  0.0-10.0 sec   113 MBytes  94.4 Mbits/sec
```

10 LCD

Platform LCD RGB format is RGB565, LCD test tool is *fb-test.*

- Output red full screen:

 # fb-test -r

- Output green full screen:

 # fb-test -g

- Output blue full screen:

 # fb-test -b

11 PWM BACKLIGHT

Platform LCD backlight is controlled by PWM, backlight level is 0-7.

- Set backlight level shell command:

 # echo 6 >/sys/class/backlight/backlight/brightness

- Get current backlight level shell command:

 # cat /sys/class/backlight/backlight/brightness

- Shutdown backlight shell command:

 # echo 0 >/sys/class/backlight/backlight/brightness

12 USER GPIO

Platform export GPIO to user from user GPIO driver, user can control GPIO in shell, and also can control GPIO in APP by accessing GPIO char device node(/dev/user_gpios).

- Control GPIO in shell(take pb31 for example).

Set pb31 GPIO as output, and output high:
 # echo 0:1 >/sys/devices/user_gpios/user_gpios/pb31/ctrl
Set pb31 gpio output, and output low:
 # echo 0:0 >/sys/devices/user_gpios/user_gpios/pb31/ctrl
Set pb31 gpio as input:
 # echo 1:0 >/sys/devices/user_gpios/user_gpios/pb31/ctrl
Get GPIO input level:
 # cat /sys/devices/user_gpios/user_gpios/pb31/val
User also can simply set GPIO output level, if have set GPIO output.
 # echo 0 >/sys/devices/user_gpios/user_gpios/pb31/val
 # echo 1 >/sys/devices/user_gpios/user_gpios/pb31/val

- Control GPIO in APP, below is user GPIO API code.

```
struct user_gpio {
    const unsigned char *label;
    int gpio;
    int type;
    int value;
};

int gpio_set(const unsigned char *label, int direction, int value)
{
    int fd;
    struct user_gpio ug;

    fd = open(DEV_NODE, O_WRONLY);
    if (fd < 0) {
        printf("open %s failed\n", DEV_NODE);
        return -1;
    }

    ug.label = label;
    ug.type = direction;
    ug.value = value;

    if (write(fd, &ug, sizeof(ug)) != sizeof(ug)) {
        printf("user gpio write failed\n");
    }

    printf("set %s-->%d\n", ug.label, ug.value);

    close(fd);
    return 0;
}

int gpio_get(const unsigned char *label)
{
    int fd;
    struct user_gpio ug;
    int value = -1;

    fd = open(DEV_NODE, O_RDONLY);
    if (fd < 0) {
```

```
        printf("open %s failed\n", DEV_NODE);
        return -1;
    }

    ug.label = label;
    ug.type = GPIO_DIRECTION_IN;

    if (read(fd, &ug, sizeof(ug)) != sizeof(ug)) {
        printf("user gpio read failed\n");
    }

    value = atoi((char *)&ug);
    printf("get %s<--%d\n", label, value);

    close(fd);
    return value;
}
```

- To check valid user GPIO labels, run shell command:

```
# ls /sys/devices/user_gpios/user_gpios/
pb26  pb27  pb29  pb31
```

The labels *pb26 pb27 pb29 pb31 are valid, and user GPIO API can use the labels .*

13 RTC RX8025T

- Check system matched RTC shell command:

```
# cat /sys/class/rtc/rtc0/name
```

- Get system time shell command:

```
# date
```

- Set system time shell command:

```
# date -s "2016-02-02 14:57:40"
```

- Get hardware RTC time shell command:

```
# hwclock
```

- Update system time to hardware RTC:

```
# hwclock  -s
```

- Update hardware RTC time to system time:

 # hwclock -w

- Another method to get hardware RTC date and time:

 # cat /sys/class/rtc/rtc0/date
 # cat /sys/class/rtc/rtc0/time

14 BUZZER

Buzzer is controlled by GPIO, platform control it as LED.

- Enable buzzer shell command:

 # echo 1 >/sys/class/leds/buzzer/brightness

- Disable buzzer shell command:

 # echo 0 >/sys/class/leds/buzzer/brightness

15 LEDS

Platform have d10 ,d8 ,d9 three LEDs, control LEDs as below(take d9 for example).

- Enable LED shell command:

 # echo 1 >/sys/class/leds/d9/brightness

- Disable LED shell command:

 # echo 0 >/sys/class/leds/d9/brightness

16 HSMC

- Insert SMC driver.(assume the driver module is located in SD card)

 # insmod /mnt/mmcblk0p1/atmel_smc.ko

- Get SMC timing register value for all supported NCS

 # cat /sys/bus/platform/devices/fc05c600.smc/smc_reg

ncs:0, setup:0x02020202, pluse:0x06040404, cycle:0x000a0008, mode:0x00020003
ncs:1, setup:0x02020202, pluse:0x06040404, cycle:0x000a0008, mode:0x00020003
ncs:2, setup:0x02020202, pluse:0x06040404, cycle:0x000a0008, mode:0x00020003

- Set HSMC controller timing register value for corresponding NCS.

Set timing register command format:
 # echo ncs:offset:date >/sys/bus/platform/devices/fc05c600.smc/smc_reg

Example for operating NCS0 .
Change setup register value:
 # echo 0:0:0x01010101 >/sys/bus/platform/devices/fc05c600.smc/smc_reg
Change pluse register value:
 # echo 0:4:0x01010101 >/sys/bus/platform/devices/fc05c600.smc/smc_reg
Change cycle register value:
 # echo 0:8:0x00030003 >/sys/bus/platform/devices/fc05c600.smc/smc_reg
Change mode register value:
#echo 0:10:0x00000000 >/sys/bus/platform/devices/fc05c600.smc/smc_reg

- Read and Write SMC device register for corresponding NCS.

Read SMC device register command format:
 # echo ncs:reg >/sys/bus/platform/devices/fc05c600.smc/dev_reg
Write SMC device register command format:
 # echo ncs:reg:data >/sys/bus/platform/devices/fc05c600.smc/dev_reg
Dump SMC device[NCS0] control register command format:
 # cat /sys/bus/platform/devices/fc05c600.smc/dev_reg

- Write zero data.

 # cat /dev/zero >/sys/bus/platform/devices/fc05c600.smc/data

- Read SMC data.

 # cat /sys/bus/platform/devices/fc05c600.smc/data > smc.dat

17 CPU BENCHMARK

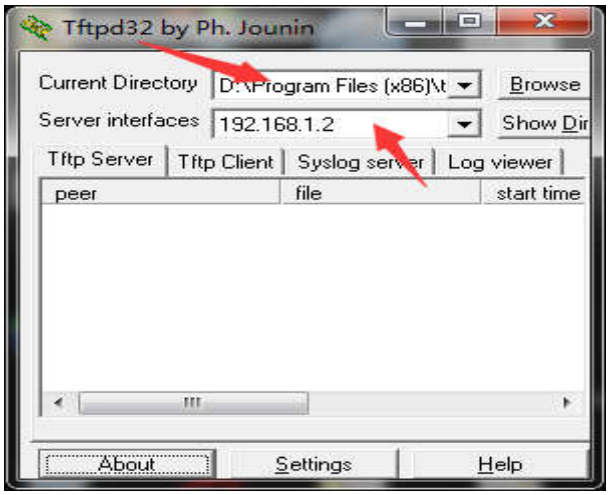
The whetstone benchmark is a synthetic benchmark for evaluating the performance of CPU, so user can evaluate ARM CPU by whetstone.

- Get the MIPS of the CPU

 # whetstone 100000
Loops: 100000, Iterations: 1, Duration: 18 sec.
C Converted Double Precision Whetstones: 555.6 MIPS

18 TFTP

Tftp is a tiny tool for transferring files betwwen cilent and server via network. Firstly, user should download tftpd exe from link: http://tftpd32.jounin.net/tftpd32_download.html, and then run tftpd exe on PC host as tftp server, but don't forget to set directory path include files to be download.



Secondly, Target filesystem should build in tftp client binary, then run command in target console.

- Download from tftp server

 # tftp –gr atmel_smc.ko 192.168.1.2

- Upload file to tftp server

 # tftp –pr atmel_smc.ko 192.168.1.2

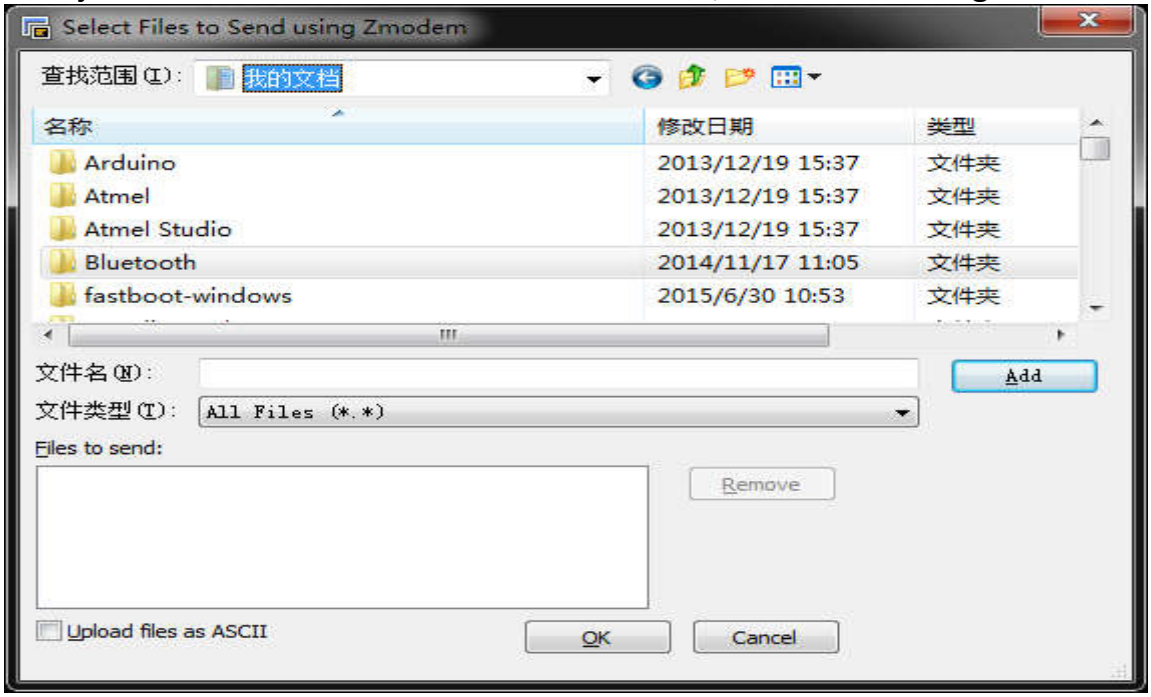
19 RZ/SZ

Platform filesystem also build in rz/sz which is transferring tool via debug serial port.

- Run rz download command on target:

 # rz

Then , PC serial console(Secure CRT) will show user file select dialog as below, user just select the download files and click OK, then transferring will start .



- Run sz upload command on target:

 # sz file_name

20 EEPROM

To access EEPROM CAT24C02, please follow below command:

- Read CAT24C02:

 # hexdump /sys/class/i2c-dev/i2c-1/device/1-0052/eeprom

- Write CAT24C02:

 # tr '\000' '\252' < /dev/zero | dd of=/sys/class/i2c-dev/i2c-1/device/1-0052/eeprom bs=1 count=256

Or

```
# tr '\000' '\125' < /dev/zero | dd of=/sys/class/i2c-dev/i2c-1/device/1-0052/eeprom bs=1 count=256
```

21 TELNET

Run below command in PC to connect target and use name ‘default’ to login :

```
# telnet target_ip
```

22 REMOUNT ROOTFS

To remount rootfs as writable:

```
# mount -o remount,rw /
```