MYS-SAM9G45 SBC Board User Manual

Version V1.3

Version History

Version number	Description	Time
V1.0	Initial Version	2012.03.28
V1.1	Adjust products list	2012.11.02
V1.2	Update Product Image	2012.11.23
V1.3	(add 7.0-inch screen support, modify contact information)	2013.05.15

Directory

Chapter 1 Product Overview

1.1 Product Description

The MYS-SAM9G45 single board computer is launched by MYIR Technology Co., Ltd based on ATMEL Corporation AT91SAM9G45 processor (ARM926EJ-S core). Running at 400 MHz, MYS-SAM9G45 has 256MB Nand-Flash, 4MB DataFlash, 128MB DDR2 SDRAM and supports Linux 2.6.30 as well as Android 2.3.1 operation system. The product provides relevant source and rich peripheral interface: high speed USB2.0, audio input/output, 10/100Mbps Ethernet interface, JTAG debug interface, serial port and Micro SD card interface. All of these provide a greater convenience for development and debug.

MYS-SAM9G45 Applications:

- Portable Data Terminal
- Single Board Computer
- Industrial Control
- Medical Equipment
- Home Automation
- Automation Infotainment
- Test and Measurement Instruments

1.2 Product Features

Based on AT91SAM9G45 processor, MYS-SAM9G45 integrates all the functions and features. The main features are as follows:

Electrical parameters

- Operating Temperature: -40℃~85℃
- Electrical Indicators: +5V power supply
- Mechanical Dimensions: 100mm x 64mm

Processor

- AT91SAM9G45(32 bits ARM RISC processor) runs at 400MHz
- 64KB on chip SRAM
- 64KB on chip ROM

Memory

- 256MB Nand-Flash，512KB EEPROM, 4MB DataFlash
- Two bank of 64MB DDR2 SDRAM

Audio and Video Interface

- A 3.5mm audio input interface
- A 3.5mm Two-channel audio output interface

LCD Touch-Screen Interface

- 24 bits True Colors
- Resolution: Current 4-inch 480 x272, 7-inch 800 x480

Transmission Interface

- Serial Ports
- A High speed USB HOST interface
- A mini USB interface
- Ethernet Interface

Input Interface

- Camera interface
- Standard JTAG interface
- SD/MMC card interface

LED Indicator Light

- A system heartbeat light/power indicator light (Red)
- A user light(Blue)

1.3 Product configuration list

No.	Name	Quantity	Note
1	MYS-SAM9G45 SBC	1	
2	1.5 Meters crosswire	1	
3	1.5 Meters Mini USB 2.0 cable	1	
4	Serial adapter board	1	Connect board to adapter board by this cable
5	Serial cable	1	Connect Serial adapter to PC by this cable
6	DVD	1	Including schematic(PDF), user manual, source code, etc.
7	4.3/7.0 inch LCD touch screen	1	Optional

Table 1-1

Chapter 2 Hardware Resource Introduction

Product preview

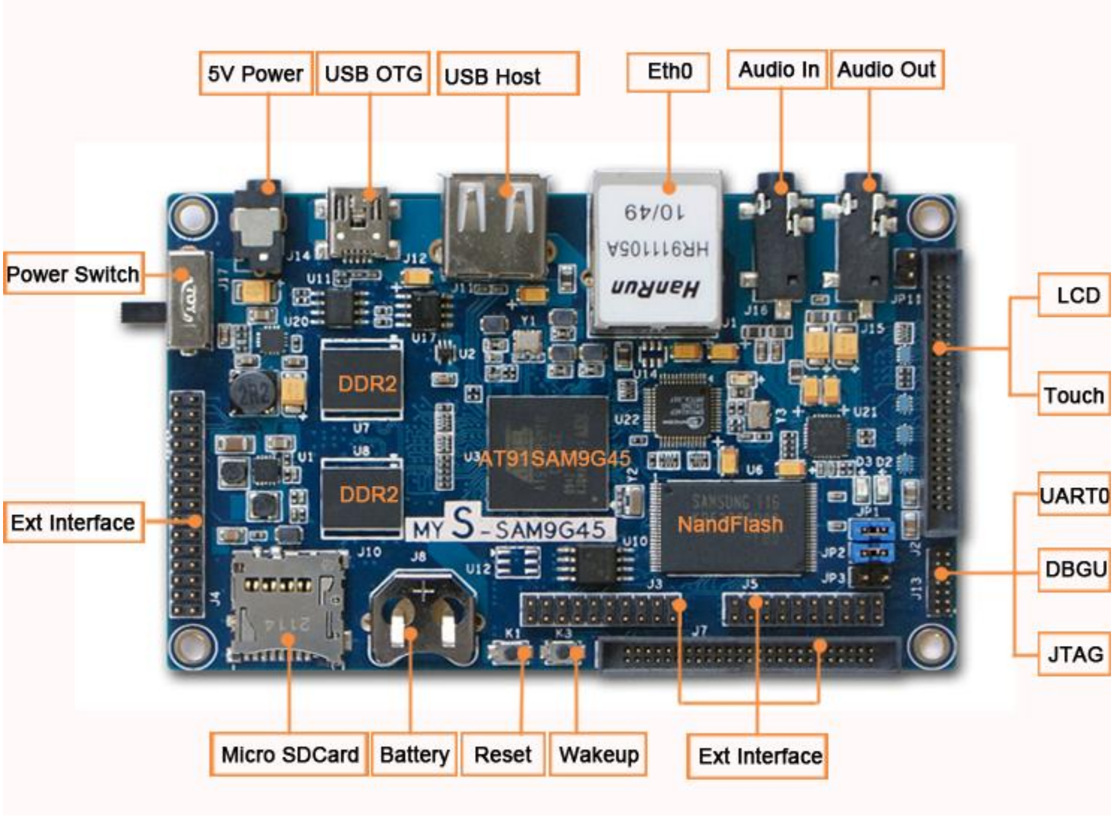


Figure 2-1

Hardware Interface List			
J1	Ethernet Interface	J14	+5V power input interface
J2	LCD interface	J15	Audio output interface
J3	Ext interface	J16	Audio input interface
J4	Ext CMOS camera interface	J17	Power switch
J5	Ext interface	K1	Reset button
J6	Ext interface	K3	Wakeup button
J7	Ext interface	D2	Blue LED
J8	Backup battery	D3	Red LED
J10	SD/MMC card connector	JP1	Nand Flash chip select
J11	USB Host	JP2	Serial Dataflash chip select
J12	Mini USB	JP3	Force Power On

J13	UART / JTAG interface		

Table 2-1

2.1 Power Input Interface

J14		
Pin	Signal	Description
1	GND	GND
2	GND	GND
3	GND	GND
4	+5V	+5V
5	GND	GND

Table 2-2

2.2 Power Switcher

J17		
Pin	Signal	Descripter
1	VBus	+5V
2	Out	Out
3	DC in	DC in

Table 2-3

2.3 Audio Output Interface

J15		
Pin	Signal	Description
1	GND	GND
2	Left	Left output
3	NC	NC
4	Right	Right output
5	NC	NC

Table 2-4

2.4 Audio Input Interface

J16		
Pin	Signal	Description

1	GND	GND
2	Left	Left Input
3	NC	NC
4	Right	Right Input
5	NC	NC

Table 2-5

2.5 Camera Interface

J4		
Pin	Signal	Description
1	+5V	+5V
2	+3V3	+3V3
3	GND	GND
4	SCK2 / PD30	USART2 Serial Clock / PD30
5	TXD2 / PB6	USART2 Transmit Data / PB6
6	RXD0 / PB7	USART0Receive Data / PB7
7	RTS2 / PC9	USART2 Request To Send / PC9
8	CTS2 / PC11	USART2 Clear To Send / PC11
9	TXD3 / PB8	USART3 Transmit Data / PB8
10	RXD3 / PB9	USART3 Receive Data / PB9
11	RTS3 / PA23	USART3 Request To Send / PA23
12	CTS3 / PA24	USART3 Clear To Send / PA24
13	GND	GND
14	SCK3 / PA22	USART3 Serial Clock / PA22
15	SPI1_ MOSI / PB15	Master In Slave Out / PB15
16	SPI1_ MISO / PB14	Master Out Slave In / PB14
17	SPI1_NPCS0/PB17	SPI Peripheral Chip Select0 / PB17
18	SPI1_SPCK / PB16	SPI Serial Clock / PB16
19	TWD1 / PB10	Two-wire Serial Data / PB10
20	TWCK1 / PB11	Two-wire Serial Clock / PB11
21	DTXD / PB13	Debug Transmit Data / PB13
22	+3V3	+3V3
23	DRXD / PB12	Debug Receive Data / PB12
24	GND	GND
25	MCI1_CDA / PA22	Multimedia Card Slot Command / PA22
26	MCI1_CK / PA31	Multimedia Card Clock / PA31
27	MCI1 DA0 / PA23	Multimedia Card 1 Slot A Data / PA23
28	MCI1 DA1 / PA24	Multimedia Card 1 Slot A Data / PA24
29	MCI1 DA2 / PA25	Multimedia Card 1 Slot A Data / PA25
30	MCI1 DA3 / PA26	Multimedia Card 1 Slot A Data / PA26

Table 2-6

2.6 External Interface

2.6.1 J3

J3		
Pin	Signal	Description
1	+5V	+5V
2	+3V3	+3V3
3	GND	GND
4	SCK0 / PB16	USART0 Serial Clock / PB16
5	TXD0 / PB19	USART0 Transmit Data / PB19
6	RXD0 / PB18	USART0 Receive Data / PB18
7	RTS0 / PB17	USART0 Request To Send / PB17
8	CTS0 / PB15	USART0 Clear To Send / PB15
9	TXD1 / PB4	USART1 Transmit Data / PB4
10	RXD1 / PB5	USART1 Receive Data / PB5
11	RTS1 / PD16	USART1 Request To Send / PD16
12	CTS1 / PD17	USART1 Clear To Send / PD17
13	GND	GND
14	SCK1 / PC29	USART1 Serial Clock / PC29
15	SPI0_MISC / PB1	Master In Slave Out / PB1
16	SPI0_MISO / PB0	Master Out Slave In / PB0
17	SPI0 NPCS0 / PB3	SPI Peripheral Chip Select0 / PB3
18	SPI0_SPCK / PB2	SPI Serial Clock / PB2
19	TWD0 / PA20	Two-wire Serial Data / PA20
20	TWCK0 / PA21	Two-wire Serial Clock / PA21

Table 2-7

2.6.2 J5

J5		
Pin	Signal	Description
1	+5V	+5V
2	+3V3	+3V3
3	GND	GND
4	AD0 / PD20	Analog Input / PD20
5	AD1 / PD21	Analog Input / PD21
6	AD2 / PD22	Analog Input / PD22

7	AD3 / PD23	Analog Input / PD23
8	AD4 / PD24	Analog Input / PD24
9	AD5 / PD25	Analog Input / PD25
10	AD6 / PD26	Analog Input / PD26
11	AD7 / PD27	Analog Input / PD27
12	NC	NC
13	PWM1 / PD31	Pulse Width Modulation Output / PD31
14	PWM0 / PD24	Pulse Width Modulation Output / PD24
15	PWM2 / PE31	Pulse Width Modulation Output / PE31
16	GND	GND
17	PWM3 / PD0	Pulse Width Modulation Output / PD0
18	NC	NC
19	ADTRG / PD28	ADC Trigger / PD28
20	ADVREF	ADC Reference

Table 2-8

2.6.3 J6

J6		
Pin	Signal	Description
1	+5V	+5V
2	GND	GND
3	+3V3	+3V3
4	GND	GND
5	Contr1 / PD8	Contr1 / PD8
6	Contr2 / PD9	Contr2 / PD9
7	TWCK0 / PA21	Two-wire Serial Clock / PA21
8	TWD0 / PA20	Two-wire Serial Data / PA20
9	NC	NC
10	MCK / PB31	MCK / PB31
11	GND	GND
12	VSYNC / PB29	Vertical Synchronization / PB29
13	GND	GND
14	HSYNC / PB30	Horizontal Synchronization / PB30
15	GND	GND
16	PCK / PB28	Processor Clock / PB28
17	NC	NC
18	D0 / PB20	D0 / PB20
19	D1 / PB21	D1 / PB21
20	D2 / PB22	D2 / PB22
21	D3 / PB23	D3 / PB23
22	D4 / PB24	D4 / PB24

23	D5 / PB25	D5 / PB25
24	D6 / PB26	D6 / PB26
25	D7 / PB27	D7 / PB27
26	D8 / PB8	D8 / PB8
27	D9 / PB9	D9 / PB9
28	D10 / PB10	D10 / PB10
29	D11 / PB11	D11 / PB11
30	GND	GND
31	GND	GND
32	GND	GND

Table 2-9

2.6.4 J7

J7		
Pin	Signal	Description
1	+5V	+5V
2	GND	GND
3	+3V3	+3V3
4	GND	GND
5	EBI1_A0	External address bus bit0
6	EBI1_A1	External address bus bit1
7	EBI1_A2	External address bus bit2
8	EBI1_A3	External address bus bit3
9	EBI1_A4	External address bus bit4
10	EBI1_A5	External address bus bit5
11	EBI1_A6	External address bus bit6
12	EBI1_A7	External address bus bit7
13	EBI1_A8	External address bus bit8
14	EBI1_A9	External address bus bit9
15	EBI1_A10	External address bus bit10
16	EBI1_A11	External address bus bit11
17	EBI1_A12	External address bus bit12
18	EBI1_A13	External address bus bit13
19	EBI1_A14	External address bus bit14
20	EBI1_A15	External address bus bit15
21	EBI1_A16	External address bus bit16
22	EBI1_A17	External address bus bit17
23	EBI1_A18	External address bus bit18
24	EBI1_A19	External address bus bit19
25	NRST	NRST
26	VBAT	VBAT

27	EBI1_D0	External data bus bit0
28	EBI1_D1	External data bus bit1
29	EBI1_D2	External data bus bit2
30	EBI1_D3	External data bus bit3
31	EBI1_D4	External data bus bit4
32	EBI1_D5	External data bus bit5
33	EBI1_D6	External data bus bit6
34	EBI1_D7	External data bus bit7
35	EBI1_D8	External data bus bit8
36	EBI1_D9	External data bus bit9
37	EBI1_D10	External data bus bit10
38	EBI1_D11	External data bus bit11
39	EBI1_D12	External data bus bit12
40	EBI1_D13	External data bus bit13
41	EBI1_D14	External data bus bit14
42	EBI1_D15	External data bus bit15
43	EBI1_NRD	External data bus bit16
44	EBI1_NWE	External data bus bit17
45	EBI1_NCS2	External data bus bit18
46	IRQ PD19	External data bus bit19
47	PC13	PC31
48	NWAIT / PC15	NWAIT / PC15
49	GND	GND
50	GND	GND

Table 2-10

2.7 USB Interface

USB Host Interface

J11		
Pin	Signal	Description
1	A1	+5V
2	A2	USB Data-
3	A3	USB Data+
4	A4	GND

Table 2-11

Mini USB Interface

J12		
Pin	Signal	Description
1	VBUS	+5V

2	DM	USB Data-
3	DP	USB Data+
4	ID	USB ID
5	GND	GND

Table 2-12

2.8 UART and JTAG Interface

J13-JTAG		
Pin	Signal	Description
1	NTRST	Test system select
2	R1IN	Receive1 data input
3	TDI	Test data input
4	T1OUT	Transit1 data output
5	TMS	Test mode select
6	GND	GND
7	TCK	Test clock
8	NC	NC
9	RTCK	Receive test clock
10	GND	GND
11	TDO	Test data output
12	VDDIO	VDDIO
13	NRST	Microcontroller K1
14	VDDIO	VDDIO

Table 2-13

J13-UART		
Pin	Signal	Description
2	R1IN()	Receive1 data input
4	T1OUT	Transit1 data output
10	GND	GND

Table 2-14

Note: serial adapter board should be inserted into this interface for debug

2.9 LCD Interface

J2		
Pin	Signal	Description
1	+5V	+5V
2	+5V	+5V
3	+3V3	+3V3
4	+3V3	+3V3
5	GND	GND
6	GND	GND
7	PE23 LCDDAT16	PE23 LCDDAT16
8	PE24 LCDDAT17	PE24 LCDDAT17
9	PE25 LCDDAT18	PE25 LCDDAT18
10	PE26 LCDDAT19	PE26 LCDDAT19
11	PE27 LCDDAT20	PE27 LCDDAT20
12	PE28 LCDDAT21	PE28 LCDDAT21
13	PE29 LCDDAT22	PE29 LCDDAT22
14	PE30 LCDDAT23	PE30 LCDDAT23
15	PE15 LCDDAT8	PE15 LCDDAT8
16	PE16 LCDDAT9	PE16 LCDDAT9
17	PE17 LCDDAT10	PE17 LCDDAT10
18	PE18 LCDDAT11	PE18 LCDDAT11
19	PE19 LCDDAT12	PE19 LCDDAT12
20	PE20 LCDDAT13	PE20 LCDDAT13
21	PE21 LCDDAT14	PE21 LCDDAT14
22	PE22 LCDDAT15	PE22 LCDDAT15
23	PE7 LCDDAT0	PE7 LCDDAT0
24	PE8 LCDDAT1	PE8 LCDDAT1
25	PE9 LCDDAT2	PE9 LCDDAT2
26	PE10 LCDDAT3	PE10 LCDDAT3
27	PE11 LCDDAT4	PE11 LCDDAT4
28	PE12 LCDDAT5	PE12 LCDDAT5
29	PE13 LCDDAT6	PE13 LCDDAT6
30	PE14 LCDDAT7	PE14 LCDDAT7
31	GND	GND
32	BL_CN	BL_CN
33	LCD_PWR_EN	PE0 Power Enable
34	LCDDISP PE1	LCDDISP PE1
35	INT PA27	PA27
36	PD6	PD6
37	TWCK1 / PB11	Two-wire Serial Clock

38	TWD1 / PB10	Two-wire Serial Data
39	GND	GND
40	VDEN	LCDDEN PE6
41	VSYNC	LCD VSYNC ,Link to pin-PE3
42	HSYNC	LCD HSYNC ,Link to pin-PE4
43	VCLK	Lcdpck PE5
44	GND	GND
45	TP_XR	AD2_YP PD22
46	TP_XL	AD3_YM PD23
47	TP_YD	AD1_XM PD21
48	TP_YU	AD30XP PD20
49	NC	NC
50	GND	GND

Table 2-15

2.10 SD card Interface

J10		
Pin	Signal	Description
1	DAT2	Card data 2
2	DAT3	Card data 3
3	CMD	Command signal
4	VCC	VCC
5	CLK	Clock
6	VSS	GND
7	DAT0	Card data 0
8	DAT1	Card data 1
9	CD	Card detect
10	GND	GND

Table 2-16

2.11 Ethernet Interface

J1		
Pin	Signal	Description
1	TD+	TD+ output
2	TD-	TD- output
3	RD+	RD+ input
4	CT	AVDDT
5	CT	AVDDT
6	RD-	RD- input

7	NC	NC
8	GND	GND
9	+3V3	+3V3
10	LED1	Left LED
11	LED2	Right LED
12	+3V3	+3V3

Table 2-17

2.12 LED Interface

LED		
Pin	Signal	Description
D2	PD31	Blue, User LED
D3	PD30	Red, User LED and system power reset LED

Table 2-18

2.13 Jumper Setting

No.	Function Description	
	Connect	Disconnect
JP1	Enable NandFlash	Disable Nand Flash
JP2	Enable Serial Dataflash	Disable Serial Dataflash
JP3	Force Power mode	Normal mode(Default)

Table 2-19

Chapter 3 MDK Routine

3.1 Overview

MDK routines are naked programs without operating system and its development tool is Keil uVision 4.01. This chapter describes how to use and write testing procedures. The contents include as follows:

(1) Build and configure MDK development environment;

(2) Debug, compile and download MDK sample program;

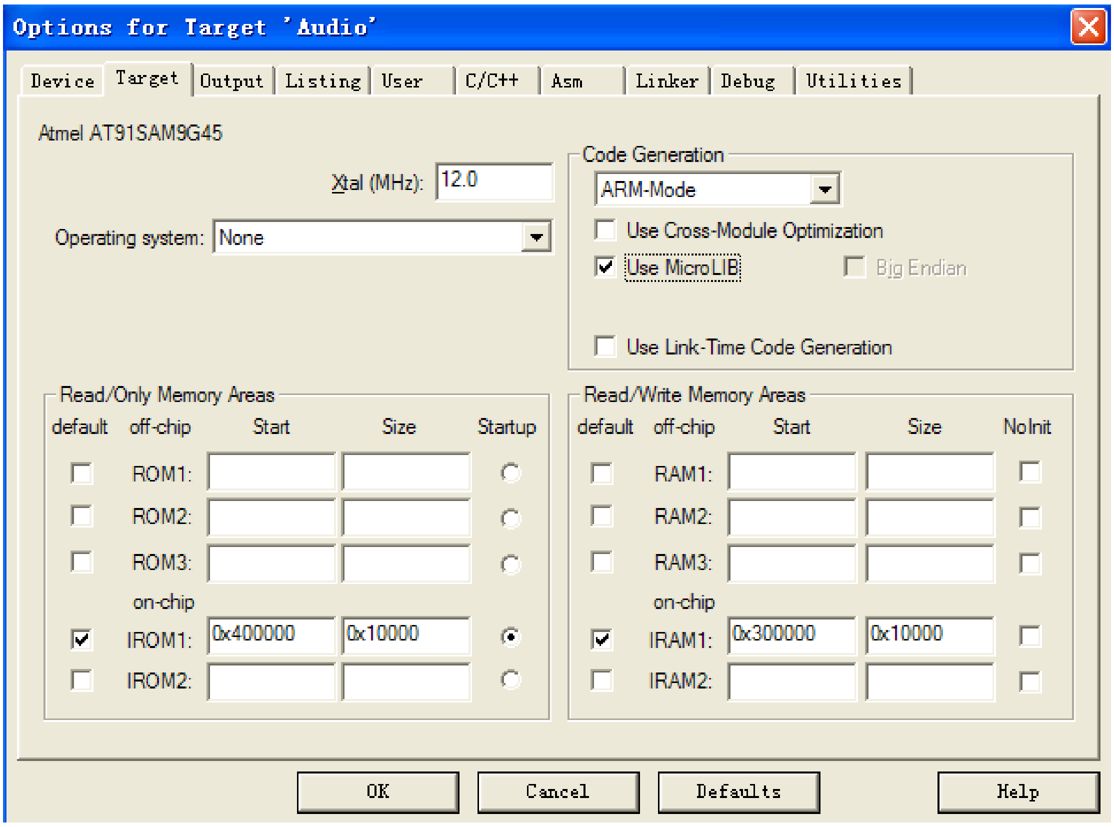
(3) MDK test procedure use, mainly include the following sections:

- User interaction - include voice, image output test and touch screen test
- Storage system - include NandFlash, DataFlash, EEPROM and MMC/SD card test
- System functions-include RTC, TWI (I²C) and RTT etc
- Data communication-include Ethernet, USB and UART test

3.2 Configure and compile MDK Routine

Firstly, install software MDK-ARM (version MDK4.01), and then open project to test. Take test audio for an example, find 04-MDK_Source\01_audio file folder, and double-click to open project Audio.uvproj. Setting download options as following (Noted,Default project setting can download successfully, please recheck if program compile or download fails):

(1) Select menu “**Project**”->”**Options for Target Audio**”, open setting window. Refer to figure 3-1:



Atmel AT91SAM9G45

Xtal (MHz): 12.0

Operating system: None

Code Generation

ARM-Mode

□ Use Cross-Module Optimization

☑ Use MicroLIB □ Big Endian

□ Use Link-Time Code Generation

Read/Only Memory Areas

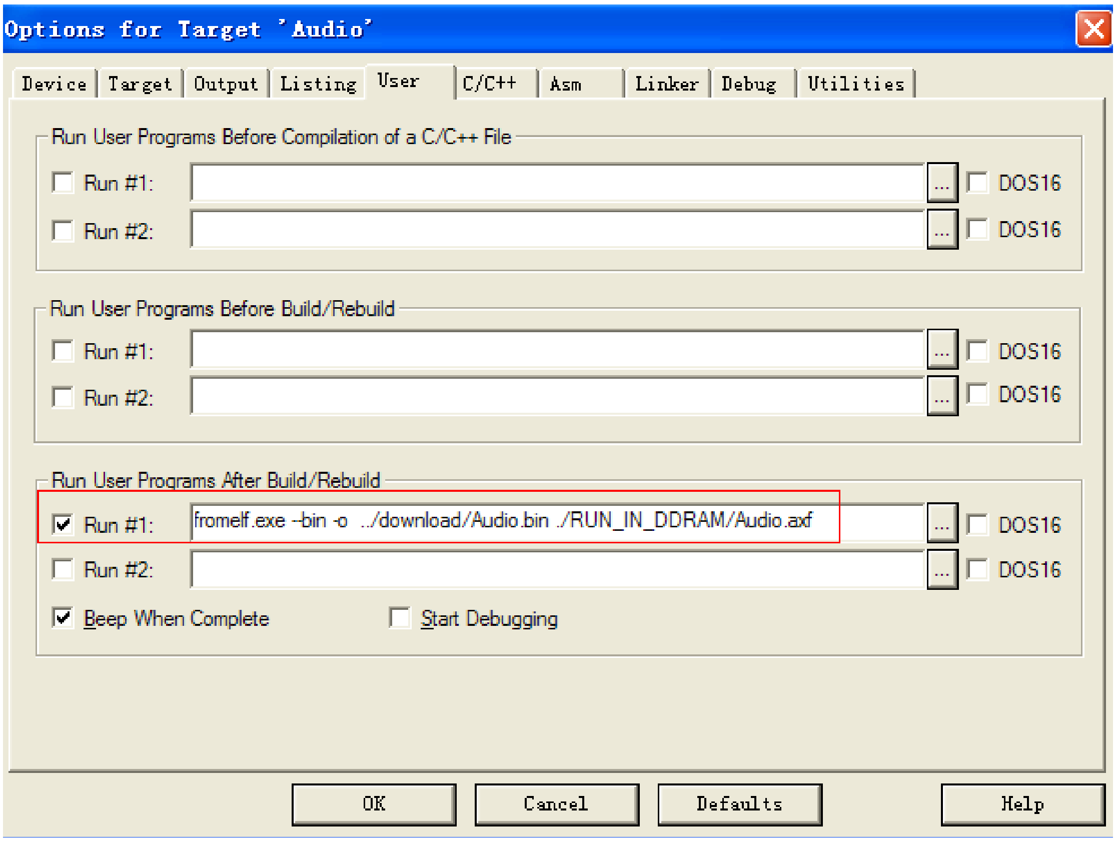
default	off-chip	Start	Size	Startup
□	ROM1:			○
□	ROM2:			○
□	ROM3:			○
	on-chip			
☑	IROM1:	0x400000	0x10000	●
□	IROM2:			○

Read/Write Memory Areas

default	off-chip	Start	Size	NoInit
□	RAM1:			□
□	RAM2:			□
□	RAM3:			□
	on-chip			
☑	IRAM1:	0x300000	0x10000	□
□	IRAM2:			□

Figure 3-1

(2) Click “User” Tab in figure 3-1. Input command which specifies binary file formation in “**Run User Programs After Build/Rebuild**”, as show in figure 3-2:



Run User Programs Before Compilation of a C/C++ File

□ Run #1: □ DOS16

□ Run #2: □ DOS16

Run User Programs Before Build/Rebuild

□ Run #1: □ DOS16

□ Run #2: □ DOS16

Run User Programs After Build/Rebuild

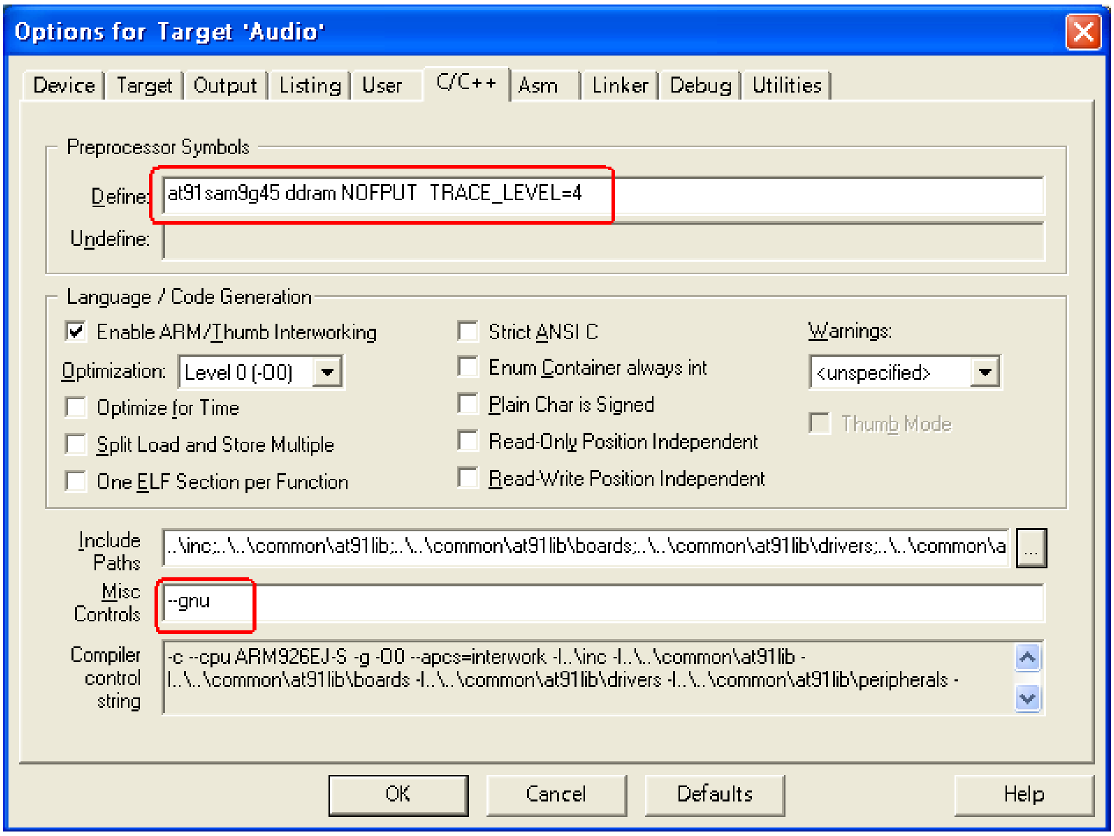
☑ Run #1: fromelf.exe --bin -o ../download/Audio.bin ./RUN_IN_DDRAM/Audio.axf □ DOS16

□ Run #2: □ DOS16

☑ Beep When Complete □ Start Debugging

Figure 3-2

(3) Switch to C/C++ Tab, setting as figure 3-3:



Preprocessor Symbols

Define: at91sam9g45 ddram NOFPUT TRACE_LEVEL=4

Undefine:

Language / Code Generation

☑ Enable ARM/Thumb Interworking

Optimization: Level 0 (-O0)

□ Optimize for Time

□ Split Load and Store Multiple

□ One ELF Section per Function

□ Strict ANSI C

□ Enum Container always int

□ Plain Char is Signed

□ Read-Only Position Independent

□ Read-Write Position Independent

Warnings: <unspecified>

□ Thumb Mode

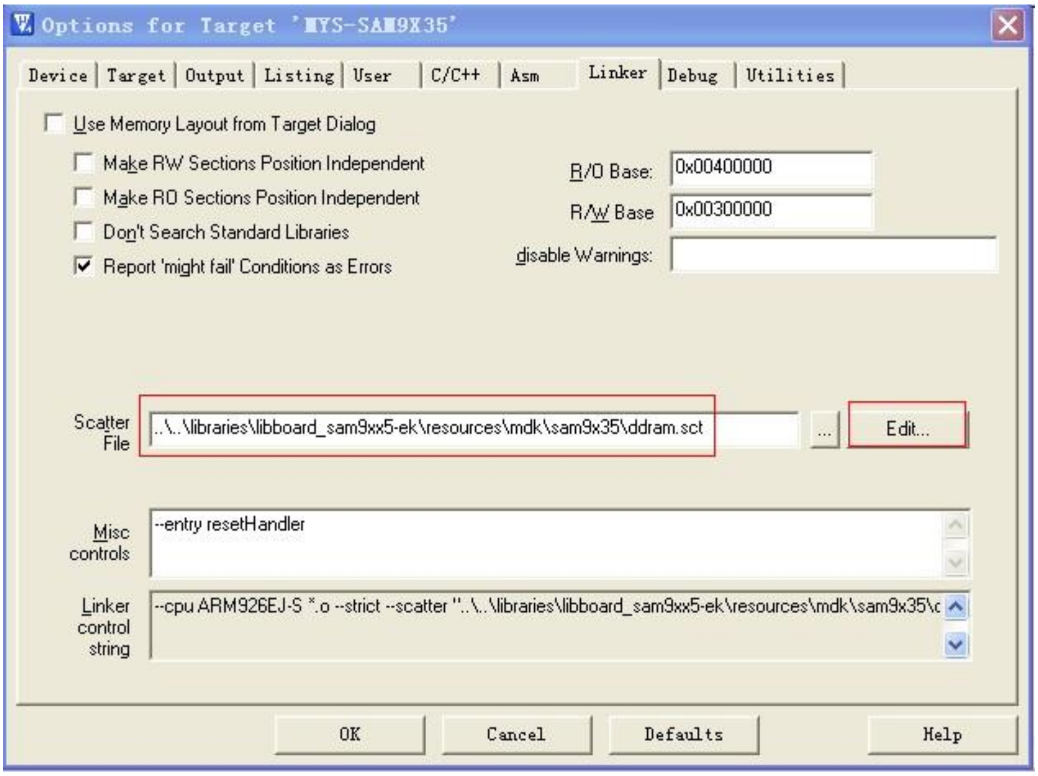
Include Paths: ..\inc;..\..\common\at91lib;..\..\common\at91lib\boards;..\..\common\at91lib\drivers;..\..\common\a

Misc Controls: --gnu

Compiler control string: -c --cpu ARM926EJ-S -g -O0 --apcs=interwork -I..\inc -I..\..\common\at91lib -I..\..\common\at91lib\boards -I..\..\common\at91lib\drivers -I..\..\common\at91lib\peripherals -

Figure 3-3

(4) Switch to Linker Tab, setting as figure 3-4:



□ Use Memory Layout from Target Dialog

□ Make RW Sections Position Independent

□ Make RO Sections Position Independent

□ Don't Search Standard Libraries

☑ Report 'might fail' Conditions as Errors

R/O Base: 0x00400000

R/W Base 0x00300000

disable Warnings:

Scatter File: ..\..\libraries\libboard_sam9xx5-ek\resources\mdk\sam9x35\ddram.sct

Misc controls: --entry resetHandler

Linker control string: --cpu ARM926EJ-S *.o --strict --scatter "..\..\libraries\libboard_sam9xx5-ek\resources\mdk\sam9x35\c

Figure 3-4

(5) "Select menu "Project" -> "Rebuild all target files" to rebuild project, as shown in figure 3-5:

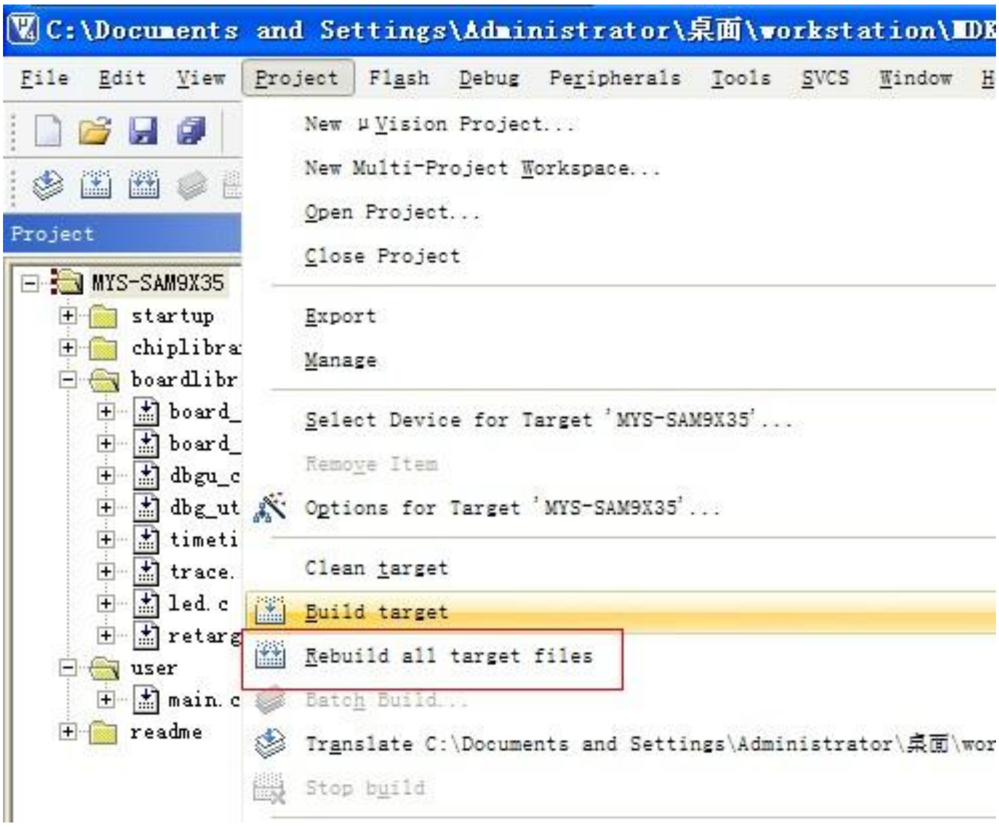


Figure 3-5

3.3 Debug and Download MDK Routine

3.3.1 Debug

The following is MDK program configuration and it has a hardware emulator ULink2 in advance. Take Audio project for an example.

(1) Double click 04-MDK_Source\01_audio\Audio.uvproj and “**Audio**”, and then choose “**Options for Target ‘Audio’**”. Refer to figure 3-6:

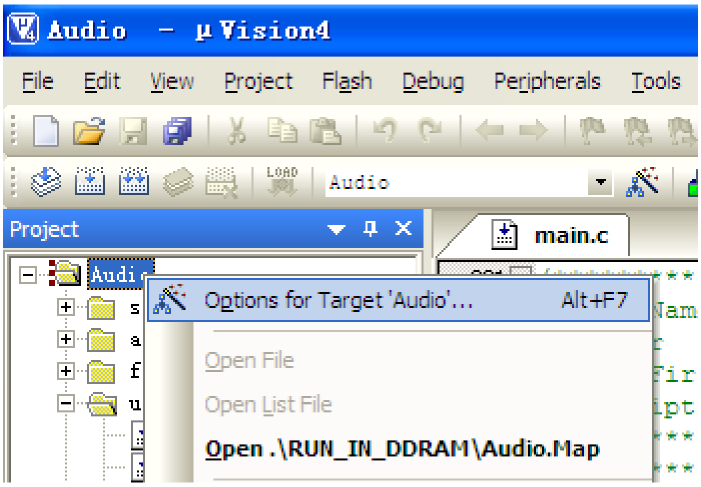
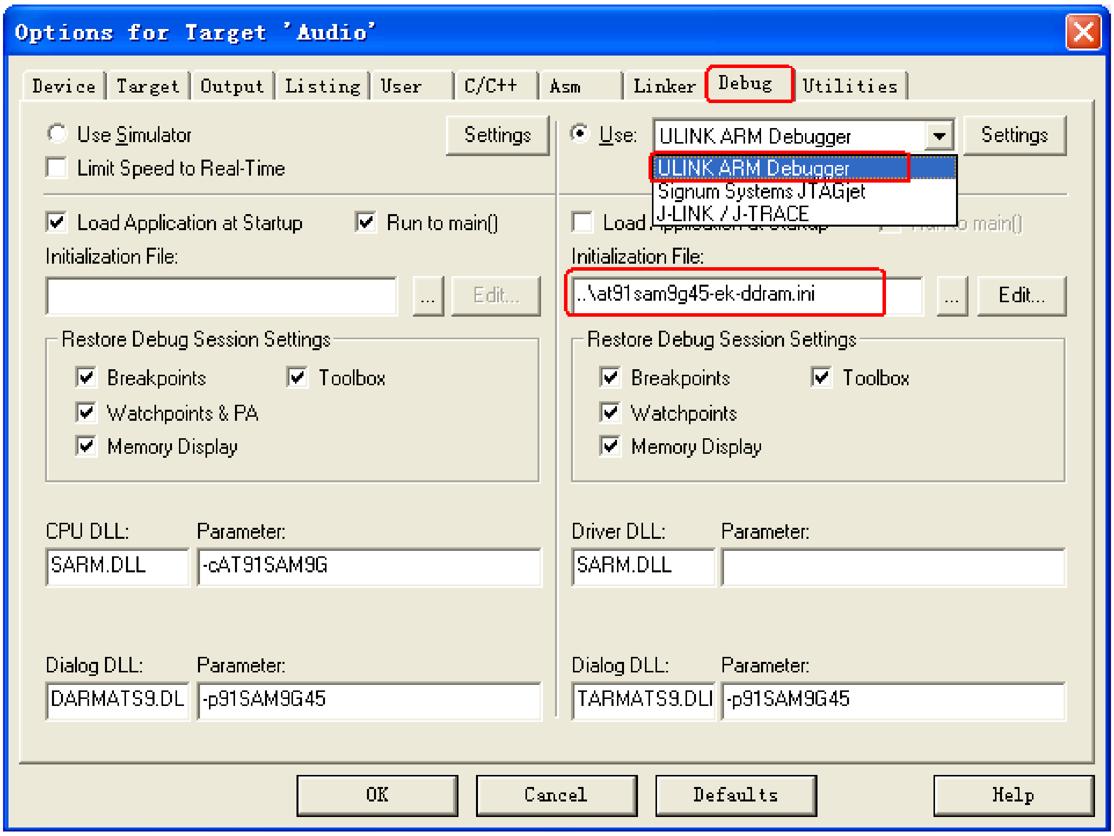

Audio – µVision4
File Edit View Project Flash Debug Peripherals Tools
Audio
Project
main.c
Options for Target 'Audio'... Alt+F7
Open File
Open List File
Open .\RUN_IN_DDRAM\Audio.Map

Figure 3-6

(2) After opening "**Option for Target 'Audio'**" then choose "**Debug**" tab, and select initialization script. Refer to figure 3-7:


Options for Target 'Audio'
Device Target Output Listing User C/C++ Asm Linker Debug Utilities
Settings
Settings
Edit...
Edit...
...
...
OK Cancel Defaults Help

○ Use Simulator

□ Limit Speed to Real-Time

☑ Load Application at Startup ☑ Run to main()

Initialization File:

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

☑ Watchpoints & PA

☑ Memory Display

CPU DLL: SARM.DLL Parameter: -cAT91SAM9G

Dialog DLL: DARMATS9.DL Parameter: -p91SAM9G45

● Use: ULINK ARM Debugger

ULINK ARM Debugger

Signum Systems JTAGjet

J-LINK / J-TRACE

□ Load Application at Startup □ Run to main()

Initialization File: ..\at91sam9g45-ek-ddram.ini

Restore Debug Session Settings

☑ Breakpoints ☑ Toolbox

☑ Watchpoints

☑ Memory Display

Driver DLL: SARM.DLL Parameter:

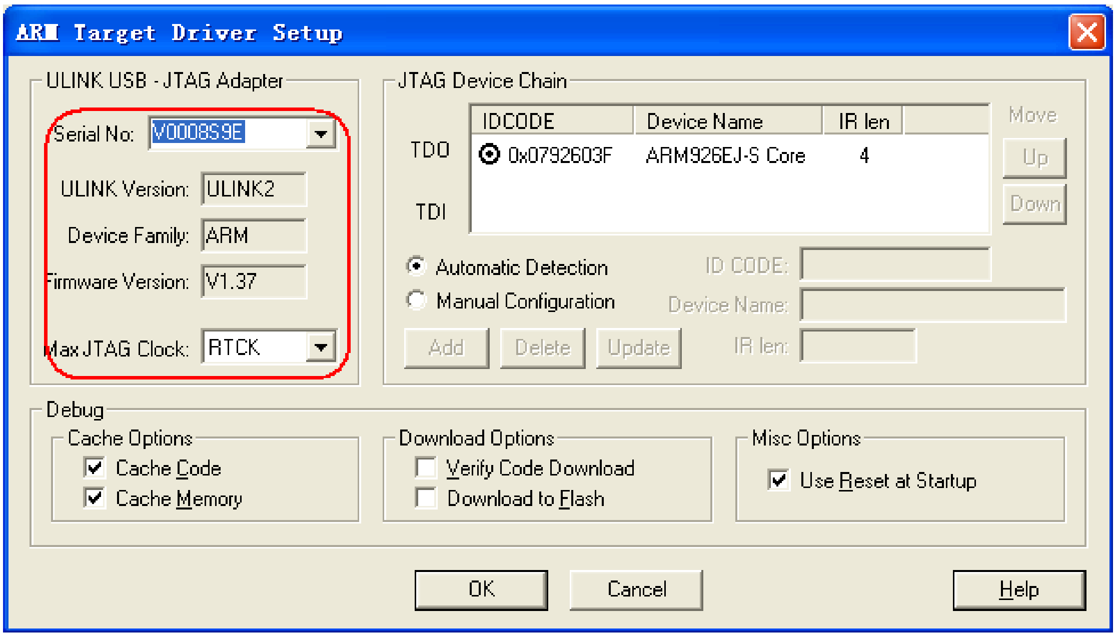
Dialog DLL: TARMATS9.DLl Parameter: -p91SAM9G45

Figure 3-7

(3) Check ULINK2 is good or not. Steps:

When connecting ULink2 to board, the indicator lights of RUN and COM change blue and then turn off, while the indicator lights change red and then remain the same. Thus, it indicates ULink2 is good. In addition, click Debug tab on the right of "Settings" button.

There will be a red mark in the figure which shows ULINK2 is good. Refer to figure 3-8:



ARM Target Driver Setup

ULINK USB - JTAG Adapter

Serial No: V0008S9E

ULINK Version: ULINK2

Device Family: ARM

Firmware Version: V1.37

Max JTAG Clock: RTCK

JTAG Device Chain

	IDCODE	Device Name	IR len
TDO	⊙ 0x0792603F	ARM926EJ-S Core	4
TDI			

● Automatic Detection

○ Manual Configuration

ID CODE:

Device Name:

IR len:

Debug

Cache Options: ☑ Cache Code ☑ Cache Memory

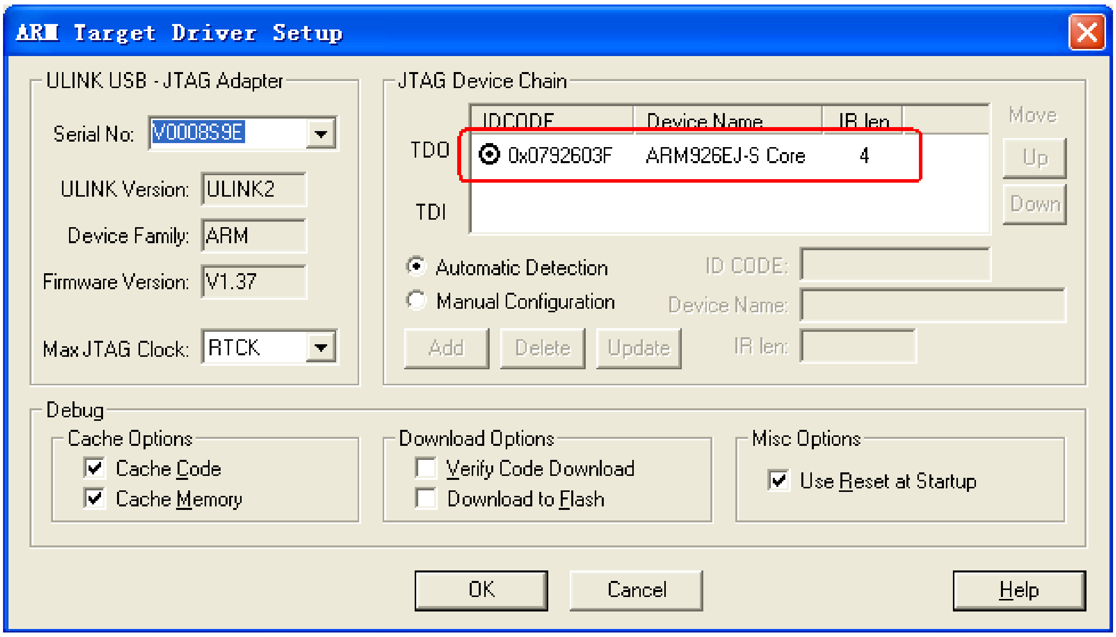
Download Options: ☐ Verify Code Download ☐ Download to Flash

Misc Options: ☑ Use Reset at Startup

Figure 3-8

(4) Check whether ULINK2 is able to detect board, this step is optional.

Clicking Setting in figure 3-10, there will be connection status of ULink2 and board, as well as kernel identification. Click Debug tab on the right of "Settings" button, there will be a red mark in the figure which shows ULINK2 detects board. Refer to figure 3-9:



ARM Target Driver Setup

ULINK USB - JTAG Adapter

Serial No: V0008S9E

ULINK Version: ULINK2

Device Family: ARM

Firmware Version: V1.37

Max JTAG Clock: RTCK

JTAG Device Chain

	IDCODE	Device Name	IR len
TDO	⊙ 0x0792603F	ARM926EJ-S Core	4
TDI			

● Automatic Detection

○ Manual Configuration

ID CODE:

Device Name:

IR len:

Debug

Cache Options: ☑ Cache Code ☑ Cache Memory

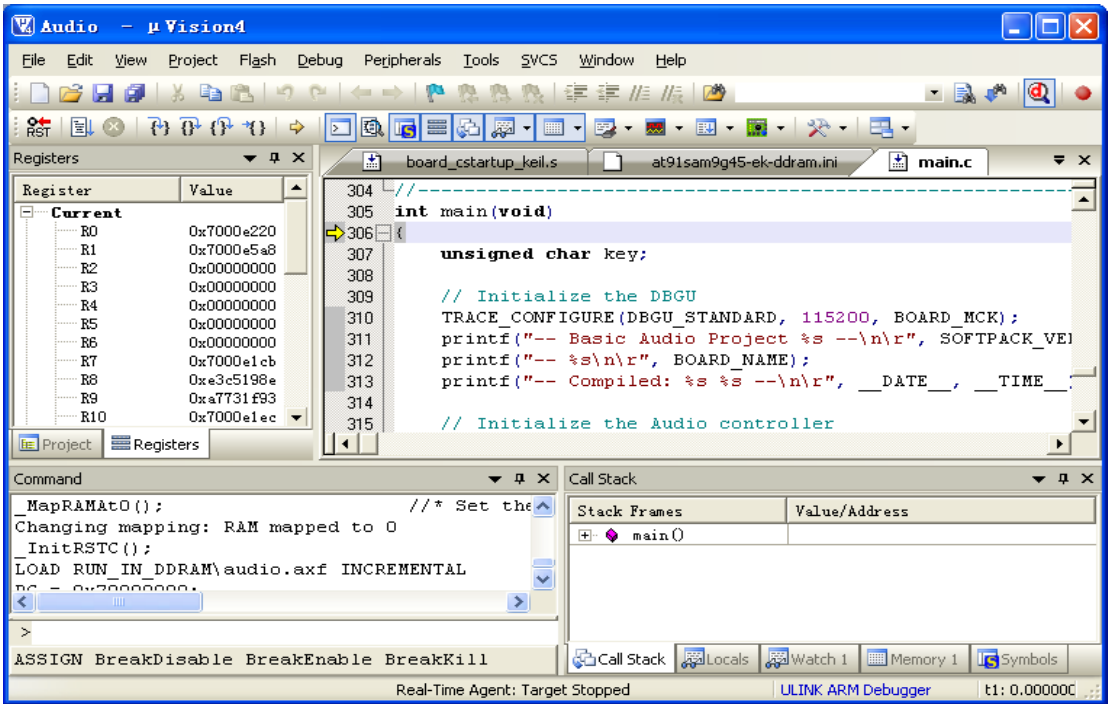
Download Options: ☐ Verify Code Download ☐ Download to Flash

Misc Options: ☑ Use Reset at Startup

Figure 3-9

(5) Click shortcut icon and Debug->Start/Stop Debug Session to debug program.

There will be debug status in figure 3-10:



Register	Value
Current	
R0	0x7000e220
R1	0x7000e5a8
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x7000e1cb
R8	0xe3c5198e
R9	0xa7731f93
R10	0x7000e1ec

```
//-----------------------------------------------------------
int main(void)
{
    unsigned char key;

    // Initialize the DBGU
    TRACE_CONFIGURE(DBGU_STANDARD, 115200, BOARD_MCK);
    printf("-- Basic Audio Project %s --\n\r", SOFTPACK_VEI
    printf("-- %s\n\r", BOARD_NAME);
    printf("-- Compiled: %s %s --\n\r", __DATE__, __TIME__

    // Initialize the Audio controller
```

```
_MapRAMAt0();                          //* Set the
Changing mapping: RAM mapped to 0
_InitRSTC();
LOAD RUN_IN_DDRAM\audio.axf INCREMENTAL
PC = 0x70000000;
>
ASSIGN BreakDisable BreakEnable BreakKill
```

Stack Frames	Value/Address
main()	

Real-Time Agent: Target Stopped ULINK ARM Debugger t1: 0.00000C

Figure 3-10

3.3.2 Download

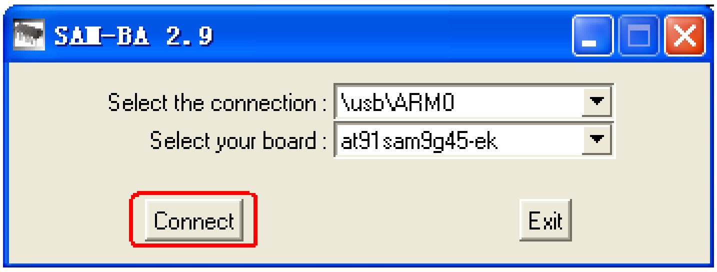
(1) Download manually

① install SAM-BA software. For details, please Refer to 02-Tools\SAM-BA\sam-ba install.

② Connect board to PC by USB, disconnect JP1, JP2 and press K1 to reset board. And here will be prompt that a removable disk inserted.

Note: Connect JP1 enable NANDFLASH; Connect JP2 enable DATAFLASH on board; Disconnect JP3 and JP4 to let the chip not boot from two medias, thus enable to connect to USB.

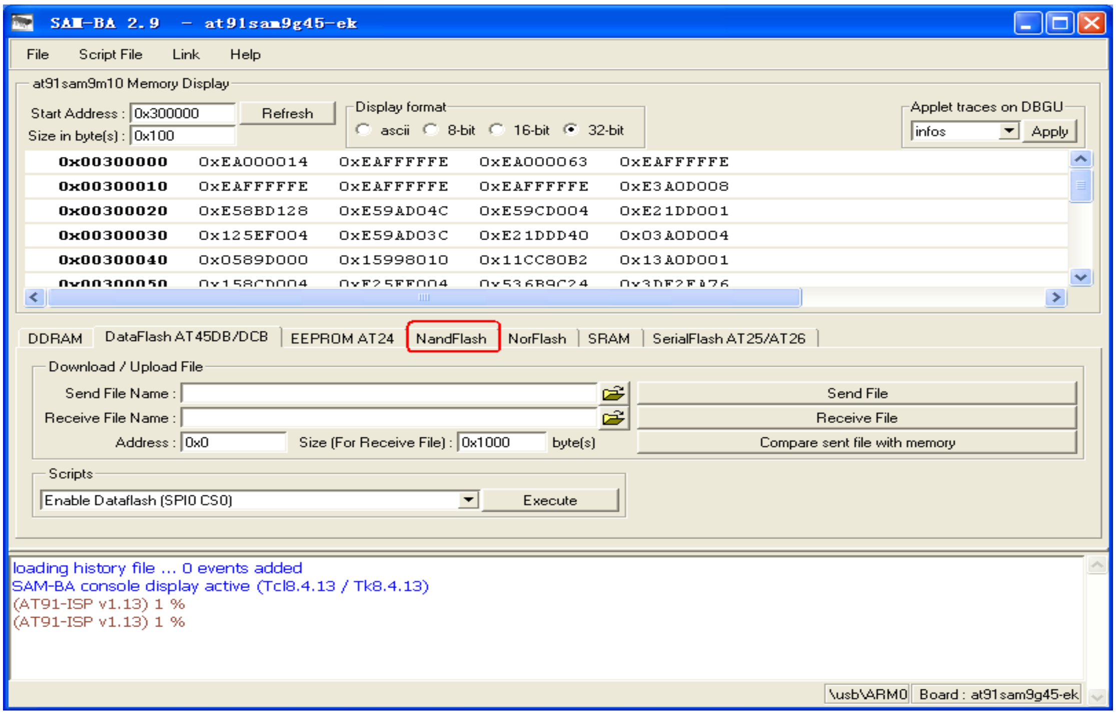
③Click "**start**"->"**All Programs**"-> "**ATMEL Corporation**" -> "**AT91-ISP v1.13**"-> "**SAM-BA v2.9**" and open SAM-BA software. The dialog popped up is saw in figure 3-11:



Select the connection : \usb\ARM0
Select your board : at91sam9g45-ek

Figure 3-11

Then click Connect and pop up SAM-BA interface. Refer to figure 3-12:



SAM-BA 2.9 - at91sam9g45-ek

at91sam9m10 Memory Display

Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0x536B9C24   0x3DF2FA76
```

Download / Upload File

Send File Name :

Receive File Name :

Address : 0x0 Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable Dataflash (SPI0 CS0)

```
loading history file ... 0 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 1 %
(AT91-ISP v1.13) 1 %
```

\usb\ARM0 Board : at91sam9g45-ek

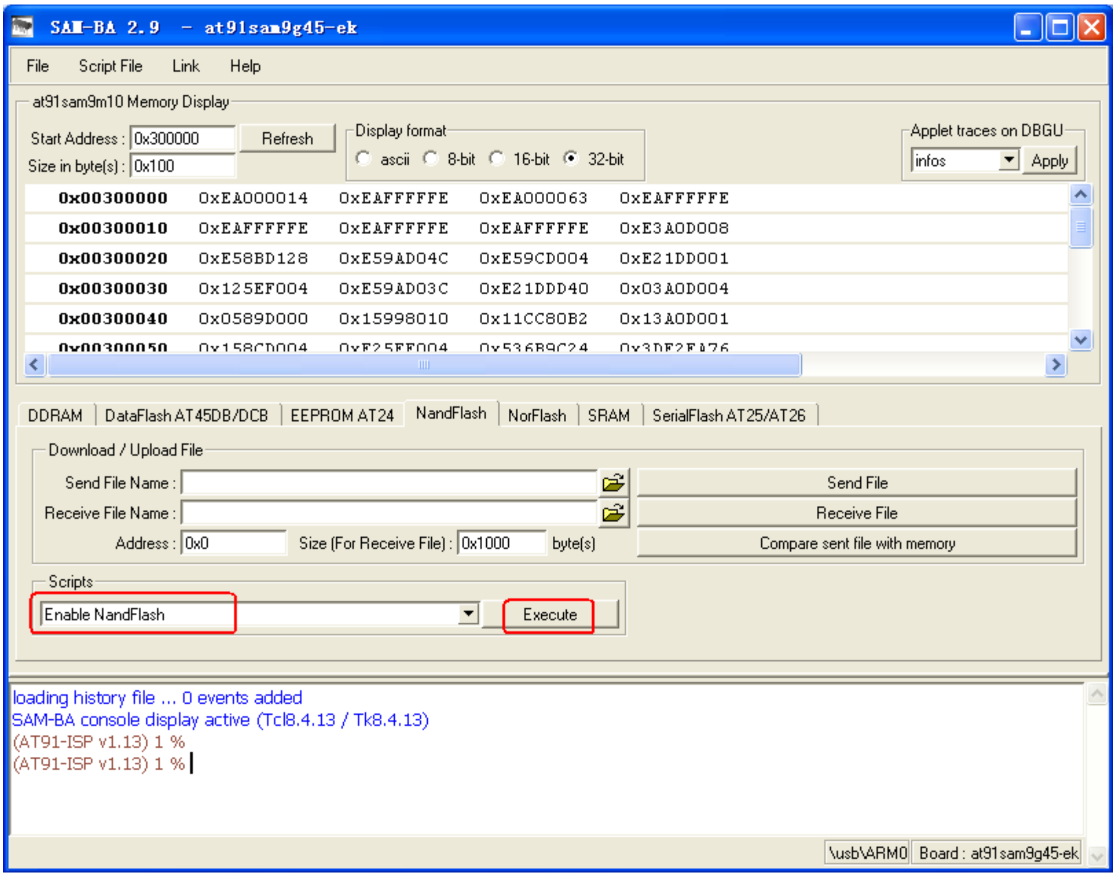
Figure 3-12

④ Connect JP1, Hardware enable NandFlash.

The steps of download program are as follows:

a． Software enables NandFlash

After connect it successfully, choose NandFlash. Refer to figure 3-13. Select “Enable NandFlash” in Script, then click “Execute” to enable NandFlash. Refer to figure 3-13:



Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0x536B9C24   0x3DF2FA76
```

Send File Name :

Receive File Name :

Address : 0x0 Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable NandFlash

```
loading history file ... 0 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 1 %
(AT91-ISP v1.13) 1 %
```

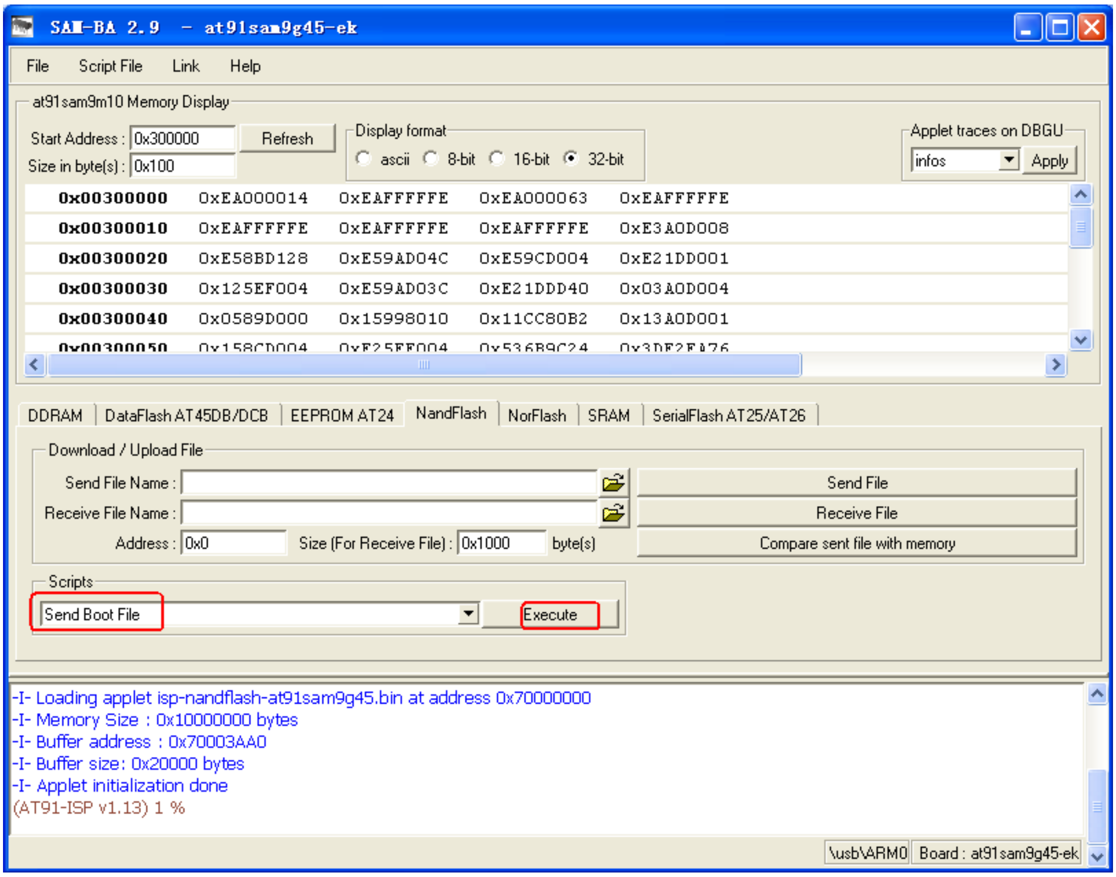
\usb\ARM0 Board : at91sam9g45-ek

Figure 3-13

b. Download nandflash_at91sam9g45ekes.bin.

Note: nandflash_at91sam9g45ekes.bin is the first class boot code. After reset chip, code starts to run from chip SRAM automatically. The first class code has two functions. One set CPU frequency, initial DDRAM and some other basic hardware initiation. The other one copies data (0x2000) from NandFlash to DDRAM. After complete it, it start to run code from entry.

Select “Send Boot File” in Scripts. Refer to figure 3-14:



Start Address : 0x300000
Size in byte(s) : 0x100
○ ascii ○ 8-bit ○ 16-bit ● 32-bit
infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0x536B9C24   0x3DF2FA76
```

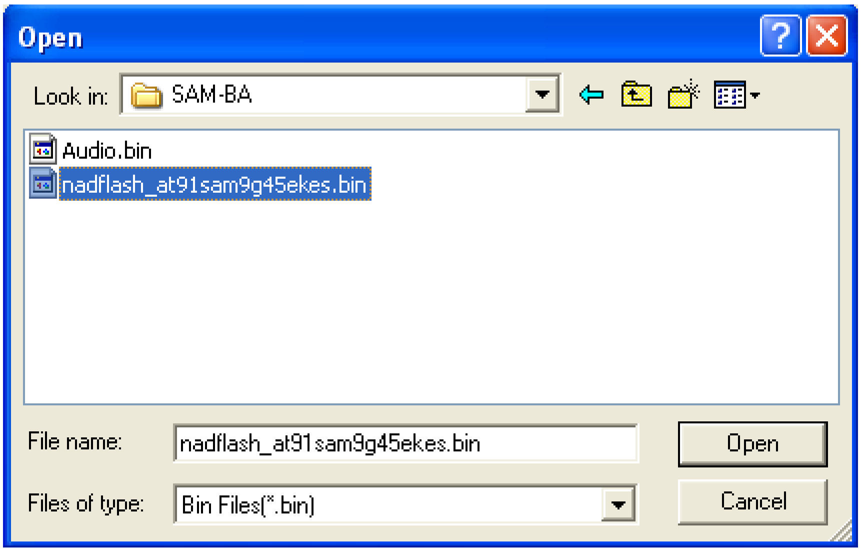
Send File Name :
Receive File Name :
Address : 0x0
Size (For Receive File) : 0x1000 byte(s)
Send Boot File

```
-I- Loading applet isp-nandflash-at91sam9g45.bin at address 0x70000000
-I- Memory Size : 0x10000000 bytes
-I- Buffer address : 0x70003AA0
-I- Buffer size: 0x20000 bytes
-I- Applet initialization done
(AT91-ISP v1.13) 1 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 3-14

Then click “Execute” button and pop up a dialog. Refer to figure 3-14:



Look in: SAM-BA

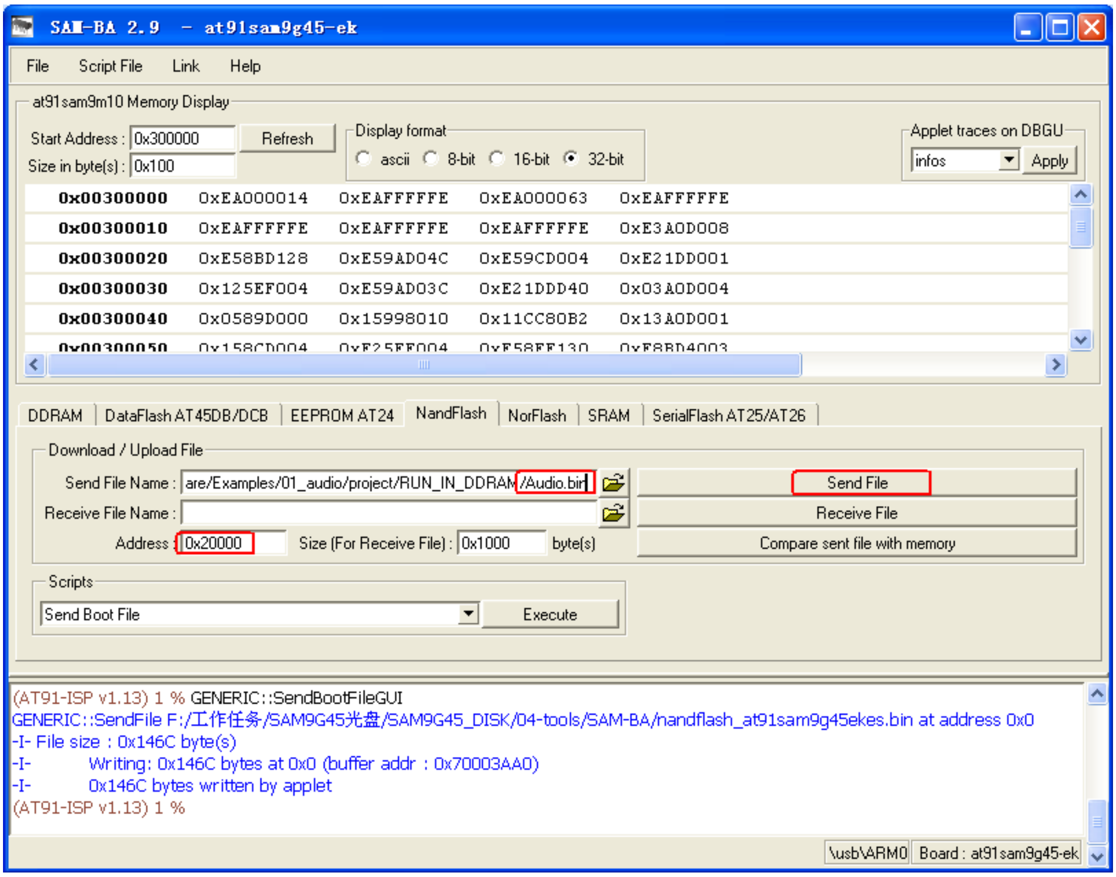
Audio.bin
nadflash_at91sam9g45ekes.bin

File name: nadflash_at91sam9g45ekes.bin
Files of type: Bin Files(*.bin)

Figure 3-15

Select nandflash_at91sam9g45ekes.bin in SAM-BA directory.

c. Download software. Choose compiled Audio.bin in Send File Name. Write 0x20000 in Address text. Refer to figure 3-16:



at91sam9m10 Memory Display

Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xF25FF004   0xF58FF130   0xF8BD4003
```

Download / Upload File

Send File Name : are/Examples/01_audio/project/RUN_IN_DDRAM/Audio.bin

Receive File Name :

Address : 0x20000   Size (For Receive File) : 0x1000 byte(s)

Scripts: Send Boot File

```
(AT91-ISP v1.13) 1 % GENERIC::SendBootFileGUI
GENERIC::SendFile F:/工作任务/SAM9G45光盘/SAM9G45_DISK/04-tools/SAM-BA/nandflash_at91sam9g45ekes.bin at address 0x0
-I- File size : 0x146C byte(s)
-I-        Writing: 0x146C bytes at 0x0 (buffer addr : 0x70003AA0)
-I-        0x146C bytes written by applet
(AT91-ISP v1.13) 1 %
```

Figure 3-16

Then click Send File button to write.

⑤Press K1 to reset board, Audio.bin starts to run.

(2) Download automatically

① Install SAM-BA software. More details please refer to 03-Tools\SAM-BA\sam-ba instal:

② Connect board to PC by USB, disconnect JP1, JP2 and press K1 to reset board. There will be prompt that a disk in task. Refer to figure 3-17:

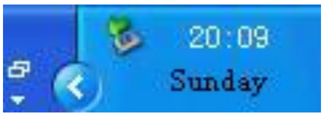


Figure 3-17

③ Connect JP1.

④ Open 04-MDK_Source\01_audio\download and click SAM9G45_MDK_nandflash.bat to download program automatically.

⑤ After download program, and press K1 to reset board, program start running.

3.4 User Interaction

3.4.1 Audio Test

Source code location: 04-MDK_Source\01-audio

Program descriptions: This program describes read wav file in SD card and output it by microphone. To play a sample.wav file in 01_audio directory, .wav file must be loaded into root Micro SD Card directory; then insert SD Card into MicroSD CARD interface. It also needs a microphone connected to J15.

Result: Download program into board, press K1 to reset board and display terminal information:

```
Start AT91Bootstrap...
-- Basic Audio Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: May 22 2012 23:47:01 --
-I- Please connect a SD card ...
-I- SD card connection detected
-I- Init media Sdcard
-I- MEDSdcard init
-I- DMAD_Initialize channel 0
-I- Card Type 1, CSD_STRUCTURE 0
-I- SD/MMC TRANS SPEED 25000 KBit/s
-I- SD 4-BITS BUS
-I- CMD6(1) arg 0x80FFFF01
-I- SD HS Enable
-I- SD/MMC TRANS SPEED 50000 KBit/s
-I- SD/MMC card initialization successful
-I- Card size: 1886 MB
-I- Mount disk 0
-I- File Found!
   Wave file header information
--------------------------------
   - Chunk ID            = 0x46464952
   - Chunk Size          = 8068644
   - Format              = 0x45564157
   - SubChunk ID          = 0x20746D66
   - Subchunk1 Size   = 16
   - Audio Format       = 0x0001
   - Num. Channels      = 2
```

```
    - Sample Rate        = 44100
    - Byte Rate          = 176400
    - Block Align        = 4
    - Bits Per Sample = 16
    - Subchunk2 ID       = 0x61746164
    - Subchunk2 Size    = 8068608
```

Then press any key, terminal displays information:

```
-I- PCM Load to 70100100, size 8068608
Menu :
------
    P: Play the WAV file
    D: Display the information of the WAV file
```

Then press “P” to play it and can hear music by microphone.

User manual: XWM8731EDS.pdf (Audio Hardware schematic) and AT91SAM9G45 Reference Manual.pdf (SSC relevant sections).

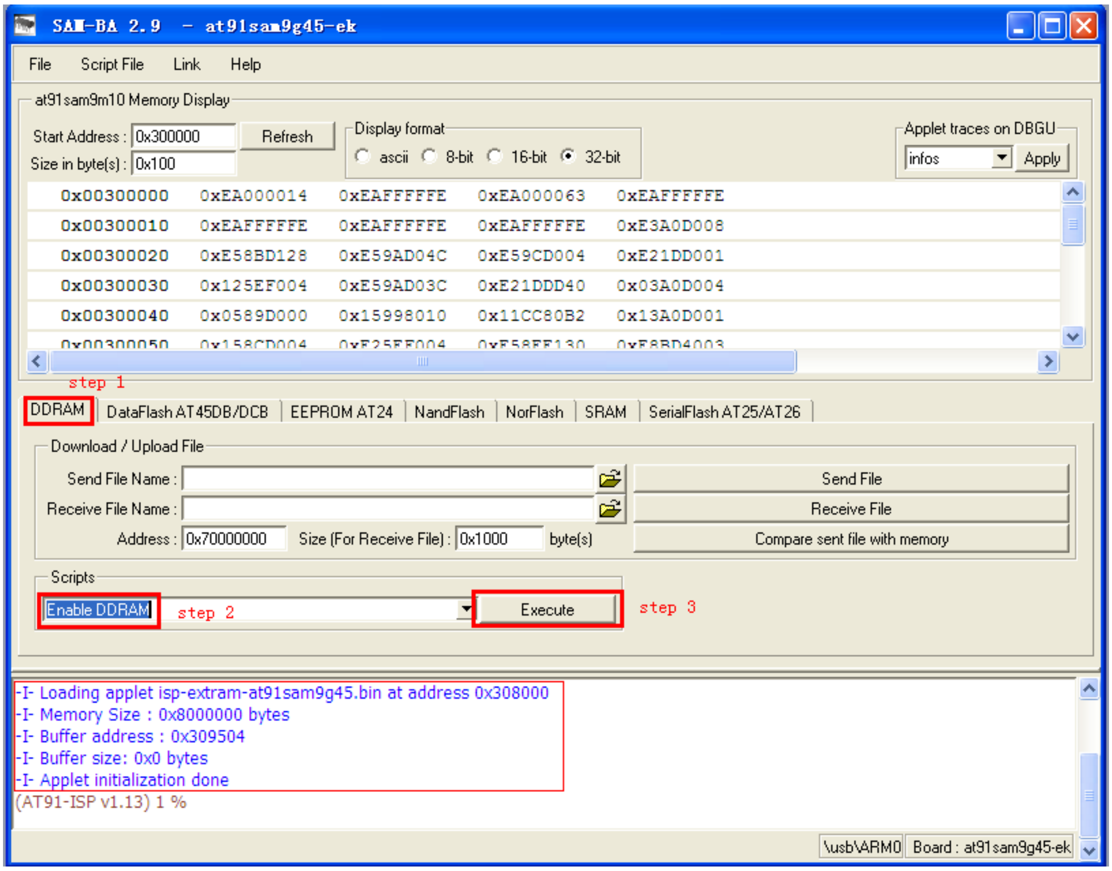
3.4.2 LCD Test

Note: 4.3-inch screen is taken as an example

Source Location: 04-MDK_Source\MDK4.01_Examples\02_lcd_4.3

Program descriptions: This program describes display picture on LCD. Before execute program, image1_rgb.raw and image2_reb.raw are downloaded at offsets 0x100000 and 0x200000. The step is as follow:

(1) Refer to download image ways, start SAM-Ba and connect board. Then select **“DDRAM”** and **“Enable DDRAM”** in Script, click **“Execute”** to initialize DDRAM. Refer to figure 3-18:



at91sam9m10 Memory Display

Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE58FF130   0xE8BD4003
```

Download / Upload File

Send File Name :

Receive File Name :

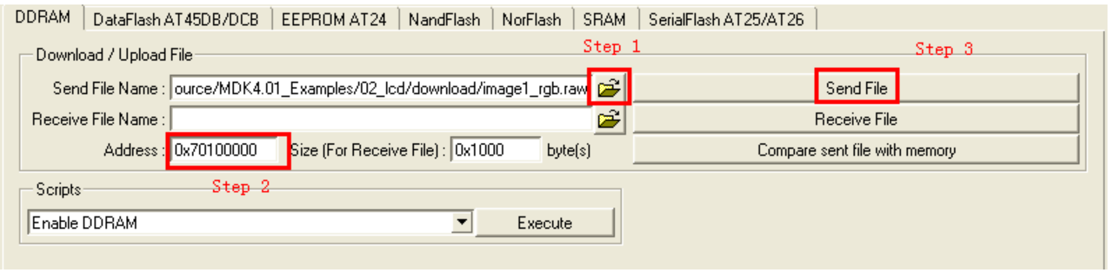
Address : 0x70000000 Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable DDRAM

```
-I- Loading applet isp-extram-at91sam9g45.bin at address 0x308000
-I- Memory Size : 0x8000000 bytes
-I- Buffer address : 0x309504
-I- Buffer size: 0x0 bytes
-I- Applet initialization done
(AT91-ISP v1.13) 1 %
```

Figure 3-18

(2) Download first picture into address: 0x70100000. Refer to figure 3-19:



Download / Upload File

Send File Name : ource/MDK4.01_Examples/02_lcd/download/image1_rgb.raw

Receive File Name :

Address : 0x70100000 Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable DDRAM

Figure 3-19

(3) Download second picture image2_rgb.raw into address: 0x70200000. Then select **“NandFlash”** and download lcd.bin automatically. **“Enable NandFlash”** and **“Send Boot File”** are sent to address: 0x20000.

Result: Press K1 to reset board and execute lcd.bin. Then two pictures are displayed on LCD alternately.

User manual: AT91SAM9G45 Reference Manual.pdf (LCDC section).

3.4.3 Touchscreen Test

Note: here 4.3-inch screen is taken as an example

Source Location: 04-MDK_Source\04_touchscreen_4.3.

Program descriptions: The program describes configure touchscreen. More details please refer to source code.

Result: Download program into board, press K1 to reset board. LCD displays:

```
LCD calibration
Touch the dots to
calibrate the screen
```

Then there will appear five red dots on LCD. Press these dots to configure touchscreen. If configure it successful, LCD will display:

```
LCD calibration
Success !
```

Otherwise LCD display below context for one second. And recover initial status to configure:

```
LCD calibration
Error too big !
```

User manual: AT91SAM9G45 Reference Manual.pdf（TSADCCS section）。

3.5 Memory System

3.5.1 NandFlash Test

Source Location: 04-MDK_Source\06_nandflash

Program description: This program shows API get information about NandFlash. Basic Nandflash operations, such as erase/write/read blocks can help you familiar with nandflash interface. More details please refer to source code.

Result: Firstly display NandFlash ID, Bus width, block size and number, page size and number. Then test block, if one block has problem, it will display “Block is BAD”. If there are no problems, it will display “Test passed”. The test will cost fifteen minutes. Display result:

```
-- Basic NandFlash Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 11:29:19--
-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: 0x8
-I- SkipBlockNandFlash_EraseBlock: Block is BAD
-I- Skip bad block 44:
-I- Test in progress on block: 95
-I- Test passed
```

User manual: AT91SAM9G45 Reference Manual.pdf (SMC section).

3.5.2 Fat File System Test

Source Location: 04-MDK_Source\07_FatFS

Program description: This program describes API gets information about FatFs and test system file. More details please refer to source code.

Results: Firstly initial FatFS and then write, read file. Display result:

```
-- Basic FatFS Full Version with External RAM Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: May 23 2011 20:58:27 --
-I- MEDDdram init
-I- DDRAM initialized
-I- Mount disk 0
-I- Format disk 0
-I- Please wait a moment during formating...
-I- Format disk finished !
-I- Create a file : "0:Basic.bin"
-I- Write file
-I- ByteWritten=512
-I- f_write ok: ByteWritten=512
-I- Close file
-I- Open file : 0:Basic.bin
-I- Read file
-I- Close file
-I- File data Ok !
-I- Test passed !
```

User manual: AT91SAM9G45 Reference Manual.pdf.

3.5.3 File system Test

Source Code Location: 04-MDK_Source\08_filesystem

Program description: This program makes 10M DDRAM into a RAM which is mounted to PC and accessed by USB. RAM can be tested and formatted by FAT file as well as EFSL file system. Detailed process, please refer to source code.

Result: Before run program, it needs to connect board to PC by USB cable. After run program, there will be a 10M disk in “My Computer ". In addition, program can format disk by FAT or EFSL file system. Input "F" by serial port to switch system file and input "R" to read and write system file. Serial display:

```
-- Basic File System Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 17:05:06—
*** Using EFSL ***
--- File System Test (EFSL) ---
FS Mount : PASS
Creat file test.bin : OK
1.Write 4194304 bytes: Done, Speed 5363 KB/s
2.Copy file test.bin to copy.bin: Done, Speed 2728 KB/s
3.Verify file copy.bin: OK, Speed 1518 KB/s
4.Read file test.bin: OK, Speed 5577 KB/s
----------------------------------------------
F to change File System Type
R to run the test again
----------------------------------------------
```

User manual: AT91SAM9G45 Reference Manual.pdf (External Memories section).

3.5.4 Dataflash Test

Source Location: 04-MDK_Source\09_dataflash

Program description: The program describes erase and write, read DataFlash. Firstly erase it, and then write test data, last compare it. If comparison is equal, it shows write it successfully. Otherwise write it unsuccessfully. More details please refer to source code.

Result: download program into board, connect JP2 (disconnect JP1) and open terminal. Press K1 to reset board, the terminal display information.

```
-- Basic Dataflash Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 19 2010 21:13:58 --
```

```
-I- Initializing the SPI and AT45 drivers
-I- At45 enabled
-I- SPI interrupt enabled
-I- Waiting for a dataflash to be connected ...
-I- AT45DB321D detected
-I- Device identifier: 0x0001271F
-I- Test in progress on page: 219
-I- Test passed.
```

User manual: AT91SAM9G45 Reference Manual.pdf (SPI section).

3.5.5 TWI_EEPROM Test

Source Location: 04-MDK_Source\10_twi_eeprom

Program description: The program describes TWI write and read EEPROM. More detail, please refer to source code.

Result: Firstly, prepare for two single board computers. One downloads “08_twi_eeprom” as a master device. The other downloads “13_twi” as a slave. Then find extern interface J6, connect pin 8(TWD), pin 7(TWCK)，pin 30(GND) in J6 by three cables. Open terminal (serial as master device). Press K1 to reset board, the terminal display information.

```
-- Basic TWI EEPROM Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 12 2010 20:50:27 --
-I- Filling page #0 with zeroes ...
-I- Filling page #1 with zeroes ...
-I- Read/write on page #0 (polling mode)
-I- 0 comparison error(s) found
-I- Read/write on page #1 (IRQ mode)
-I- Callback fired !
-I- Callback fired !
-I- 0 comparison error(s) found
-I- Callback fired !
```

It shows TWI communication is successful.

User manual: AT91SAM9G45 Reference Manual.pdf, EM_AT91SAM9G45 Board Schematic.pdf.

3.5.6 SDMMC Test

Source Location: 04-MDK_Source\11_sdmmc

Program description: The program describes write/read SD Card and model change. More details please refer to source code.

Result: Insert SD Card, download program into board. Open terminal and press K1 to observe terminal:

```
-- Basic SD/MMC MCI Mode Project xxx --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 15:58:15 --
-I- Cannot check if SD card is write-protected
-I- DMAD_Initialize channel 0
TC Start ... OK
=========================================
-I- Card Type 1, CSD_STRUCTURE 0
-I- SD 4-BITS BUS
-I- CMD6(1) arg 0x80FFFF01
-I- SD HS Not Supported
-I- SD/MMC TRANS SPEED 25000 KBit/s
-I- SD/MMC card initialization successful
-I- Card size: 483 MB, 990976 * 512B
...
```

SD Card information is displayed in the terminal:

Press "enter" will appear help menu:

```
=========================================
# 0,1,2 : Block read test
# w,W : Write block test(With data or 0)
# b,B : eMMC boot mode or access boot partition change
# i,I : Re-initialize card
# t : Disk R/W/Verify test
# T : Disk performance test
# p : Change number of blocks in one access for test
# s : Change MCI Clock for general test
```

Debug program by help menu.

3.5.7 SD Card Test

Source Location: 04-MDK_Source\12_sdcard

Program description: The program describes speed of write and read SD Card. More details please refer to source code.

Result: Insert MicroSD Card, download program into board. Open terminal and press K1 to reset board, the terminal display information:

```
-- Basic FatFS Full Version with SDCard Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 15 2010 14:22:48 --
-I- Please connect a SD card ...
-I- SD card connection detected
-I- Init media Sdcard
-I- MEDSdcard init
-I- DMAD_Initialize channel 0
-I- Card Type 1, CSD_STRUCTURE 0
-I- SD/MMC TRANS SPEED 25000 KBit/s
-I- SD 4-BITS BUS
-I- CMD6(1) arg 0x80FFFF01
-I- SD HS Enable
-I- SD/MMC TRANS SPEED 50000 KBit/s
-I- SD/MMC card initialization successful
-I- Card size: 972 MB
-I- Mount disk 0
auto_mount_test-I- The disk is already formated.
-I- Display files contained on the SDcard :
auto_mount_test0:/BASIC.bin
-I- Do you want to erase the sdcard to re-format disk ? (y/n)!
```

User manual: AT91SAM9G45 Reference Manual.pdf, EM_AT91SAM9G45 Board Schematic.pdf.

3.5.8 FATFS SD Card Test

Source Location: 04-MDK_Source\13_fatfs_sdcard

Program description: The program describes SD Card by FAT files. More details please refer to source code.

Result: Insert MicroSD Card, download program into board, press K1 to reset board, the terminal display information.

```
-- Basic FatFS Full Version with SDCard Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 15 2010 14:22:48 --
-I- Please connect a SD card ...
-I- SD card connection detected
-I- Init media Sdcard
```

```
-I- MEDSdcard init
-I- DMAD_Initialize channel 0
-I- Card Type 1, CSD_STRUCTURE 0
-I- SD/MMC TRANS SPEED 25000 KBit/s
-I- SD 4-BITS BUS
-I- CMD6(1) arg 0x80FFFF01
-I- SD HS Enable
-I- SD/MMC TRANS SPEED 50000 KBit/s
-I- SD/MMC card initialization successful
-I- Card size: 972 MB
-I- Mount disk 0
auto_mount_test-I- Format disk 0
-I- Please wait a moment during formating...
-I- Format disk finished !
-I- Create a file : "0:Basic.bin"
-I- Write file
-I- ByteWritten=2064
-I- Close file
-I- Open the same file : "0:Basic.bin"
-I- Read file
-I- Close file
-I- File data Ok !
-I- Test passed !
```

User manual: AT91SAM9G45 Reference Manual.pdf, EM_AT91SAM9G45 Board Schematic.pdf.

3.6 System Function

3.6.1 Rtc Test

Source Location: 04-MDK_Source\14_rtc

Program description: This example describes RTC set time and alarm. When up to set time alarm, there will be triggered phenomenon. More details please refer to source code.

Result: Download program into board, press K1 to observe terminal information.

```
-- Basic RTC Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 15:58:15 --
Menu:
```

```
t - Set time
d - Set date
i - Set time alarm
m - Set date alarm
q - Quit!
[Time/Date: 00:08:35, 01/14/2010 Thu ][Alarm status:]
```

Then select “t” set time, select “d” set date, select “i” set time alarm, select “m” set date alarm. When up to the set time, it will trigger alarm and display “Triggered!”. For example, select “I” set time alarm as 00:09:00. When time at 00:09:00, it will display the below information.

```
Menu:
t - Set time
d - Set date
i - Set time alarm
m - Set date alarm
c - Clear alarm notification
q - Quit!
[Time/Date: 00:09:00, 01/14/2010 Thu ][Alarm status:Triggered!]
```

Then select “C” to clear alarm prompt.

Select “q”, there is any response when inputting any key in terminal.

User manual: AT91SAM9G45 Reference Manual.pdf.

3.6.2 TWI Test

Source Location: 04-MDK_Source\15_twi

Program description: This example describes TWI peripheral in slave mode. More details please refer to source code.

Result: Firstly, prepare for two single board computers. One downloads “08_twi_eeprom” as a master device. The other downloads “13_twi” as a slave. Find extern interface J6, connect pin 8(TWD), pin 7(TWCK), pin 30(GND) in J6 by three cables. Open terminal, press K1 to observe terminal information (Note, board as master device).

```
-- Basic TWI Slave Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 15:58:15 --
-I- Configuring the TWI in slave mode
```

It shows configure it successfully.

User manual: AT91SAM9G45 Reference Manual.pdf, MYS_AT91SAM9G45 Board Schematic.pdf.

3.6.3 DMA_screen Test

Source Location: 04-MDK_Source\16_dma_screensaver

Program description: This example shows DMA controller transfers picture. More details please refer to source code.

Result: If use 320x240 LCD, Image 320x240.bmp will be downloaded into DDRAM at offset 0x10000(physical address: 0x70100000). If choose 480x272 LCD, Image 480x272.bmp should be downloaded into DDRAM. Download address is as the same. After download program, there will be effect of changing picture. Terminal displays information:

```
-- Basic DMA Screensaver Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: May 24 2011 09:20:55 --
-I- DMAD_Initialize channel 1
-I- Callback fired !
-I- DMAD_Initialize channel 0
-I- Callback fired !
-I- DMAD_Initialize channel 1
-I- Callback fired !
-I- DMAD_Initialize channel 0
-I- Callback fired !
-I- DMAD_Initialize channel 1
-I- Callback fired !
-I- DMAD_Initialize channel 0
-I- Callback fired !
-I- DMAD_Initialize channel 1
…..
```

It shows that DMA transmit picture successfully.

User manual: AT91SAM9G45 Reference Manual.pdf, MYS_AT91SAM9G45 Board.

3.6.4 RTT Test

Source Location: 04-MDK_Source\17_rtt

Program description: This program describes RTT sets an alarm triggered when timer reaches the corresponding value. More details please refer to source code.

Result: Download program into board, press K1 to observe terminal information.

```
Start AT91Bootstrap...
-- Basic RTT Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 9 2010 17:47:26 –
Time: 2
Menu:
r - K1 timer
s - Set alarm
Choice?
```

Select “r” to reset timer, then timer counts number from “0”. Select “S” set alarm. When timer reaches the corresponding value, it will trigger alarm and display “**!!! ALARM!!**”. For example, Select “S” set alarm time as “8”, when it reaches “8” and display information.

```
Time: 8
!!! ALARM !!!
Menu:
r - K1 timer
s - Set alarm
c - Clear alarm notification
Choice?
```

Then select “C” to clear prompt.

User manual: AT91SAM9G45 Reference Manual.pdf.

3.7 Date Communication

3.7.1 EMAC Test

Source Location: 04-MDK_Source\18_emac

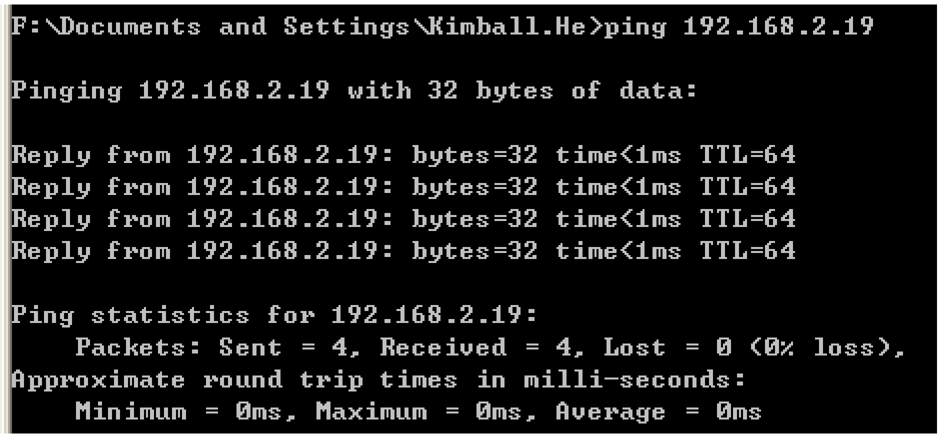
Program description: This program describes Ethernet MAC. Upon startup, configure EMAC by default IP and MAC addresses and then into best mode operation. Once this is done, it will start monitoring incoming packets and processing them whenever appropriate. Two kinds of packets (ARP, ICMP ECHO) can be tested by ping.

Result: Download program into board, connect board to the same network by

Ethernet cable. Or connect board to PC by crosswire, open terminal and press k1 to reset board, program steps into configured mode and receive package. It will display feedback information in terminal and other information when inputting any key. Information is as follows:

```
Start AT91Bootstrap...
-- Basic EMAC Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 13 2010 09:54:58 --
-- MAC 0:45:56:78:9a:ac
-- IP 192.168.2.19
-I- ** Valid PHY Found: 3
-I- MACB_K1Phy
-I- AutoNegotiate complete
P: Link detected
Press a key for statistics
=== EMAC Statistics ===
.tx_packets = 3
.tx_comp = 3
……
```

Open command line, input “ping 192.169.2.19”.



```
F:\Documents and Settings\Kimball.He>ping 192.168.2.19

Pinging 192.168.2.19 with 32 bytes of data:

Reply from 192.168.2.19: bytes=32 time<1ms TTL=64
Reply from 192.168.2.19: bytes=32 time<1ms TTL=64
Reply from 192.168.2.19: bytes=32 time<1ms TTL=64
Reply from 192.168.2.19: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.2.19:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Figure 3-20

User manual: AT91SAM9G45 Reference Manual.pdf, DM9161AEP.pdf.

3.7.2 EMAC Uip Helloworld Test

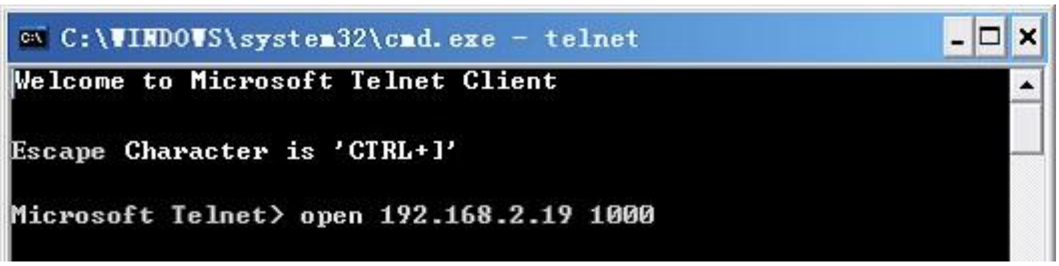
Source Location: 04-MDK_Source\19_emac_uip_helloworld

Program description: This program describes responds to port 1000. More details please refer to source code.

Result: Download program into board, connect board to the same network by Ethernet cable. Or connect board to PC with crosswire, open terminal and press k1 to observe terminal information:

```
Start AT91Bootstrap...
-- Basic EMAC uIP Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 13 2010 11:07:43 --
- MAC 0:45:56:78:9a:ac
- Host IP 192.168.2.19
- Router IP 192.168.2.1
- Net Mask 255.255.255.0
-I- ** Valid PHY Found: 3
-I- MACB_K1Phy
-I- AutoNegotiate complete
P: Link detected
P: clock time initialize - TC0
P: APP Init ... hello-world
```

Open command line and input “Telnet”, then input “open 192.168.2.19 1000”. Refer to figure 3-21:



```
Welcome to Microsoft Telnet Client

Escape Character is 'CTRL+]'

Microsoft Telnet> open 192.168.2.19 1000
```

Figure 3-21

If connection is successful, it will display “Hello. What is your name?” in terminal.

User manual: AT91SAM9G45 Reference Manual.pdf, DM9161AEP.pdf.

3.7.3 EMAC Uip Telnetd Test

Source Location: 04-MDK_Source\20_emac_uip_telnetd

Program description: This program describes telnet application. It can customize shell commands and command functions. “**Stats**”, “**conn**”、“**help/?**”, “**exit**” command represent show network statistics, TCP connection, help，exit shell respectively. Program is aimed to complete four command formats, "exit" command achieves change shell state and "help" command to display available commands menu function. Detailed process, please refer to source code.

Result: Download program into board, connect board to the same network by Ethernet cable. Or connect board to PC with crosswire, open terminal and press k1 to observe terminal information. It will display EMACcount information when inputting any key.

```
Start AT91Bootstrap...
-- Basic EMAC uIP Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 13 2010 11:45:22 --
- MAC 0:45:56:78:9a:ac
- Host IP 192.168.2.19
- Router IP 192.168.2.1
- Net Mask 255.255.255.0
-I- ** Valid PHY Found: 3
-I- MACB_K1Phy
-I- AutoNegotiate complete
P: Link detected
P: clock time initialize - TC0
P: APP Init ... telnetd
=== EMAC Statistics ===
.tx_packets = 0
.tx_comp = 0
.tx_errors = 0
.collisions = 0
.tx_exausts = 0
…….
```

Open Command line, input “telnet”, then input “open 192.168.2.19”, connect port 23 in default.

If connection is successful, there will be prompt. Press “?” will display help information.

Input “**stats**”, “**conn**”, “**help**” display help menu. Input “**exit**’ can’t display information and just shut shell down.

User manual: AT91SAM9G45 Reference Manual.pdf，DM9161AEP.pdf.

3.7.4 EMAC Uip Web Server Test

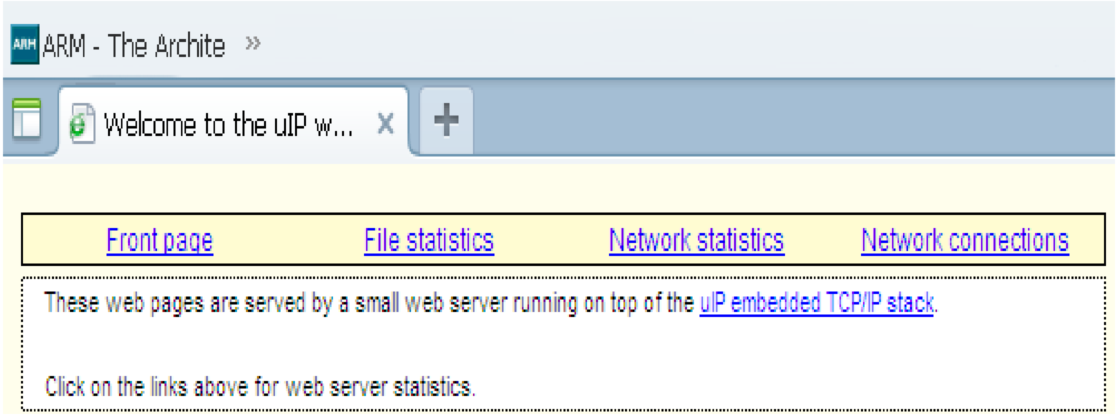
Source Location: 04-MDK_Source\21_emac_uip_webserver

Program description: This example describes webserver application. Program set uip, including IP address, router IP and subnet mask settings. After run program, board can be acted as a Web server accessed by inputting IP. Detailed process, please refer to source code.

Result: Result: Download program into board, connect board to the same network by Ethernet cable. Or connect board to PC by crosswire, open terminal and press k1 to observe terminal information. It will display EMACcount information when inputting any key.

```
Start AT91Bootstrap...
-- Basic EMAC uIP Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 13 2010 17:00:36 --
- MAC 0:45:56:78:9a:ac
- Host IP 192.168.2.19
- Router IP 192.168.2.1
- Net Mask 255.255.255.0
-I- ** Valid PHY Found: 3
-I- MACB_K1Phy
-I- AutoNegotiate complete
P: Link detected
P: clock time initialize - TC0
P: APP Init ... webserver
=== EMAC Statistics ===
.tx_packets = 0
.tx_comp = 0
.tx_errors = 0
.collisions = 0
……
```

If terminal displays “**Link detected**”, it shows connection is success. Then input string “http://192.168..2.19” to open web page. Refer to figure 3-22:



These web pages are served by a small web server running on top of the uIP embedded TCP/IP stack.

Click on the links above for web server statistics.

Figure 3-22

User manual: AT91SAM9G45 Reference Manual.pdf, DM9161AEP.pdf.

3.7.5 USB Device Core Test

Source Location: 04-MDK_Source\22_usb_device_core

Program description: This program describes Enumerate USB and UDP initialization. More details please refer to source code.

Result: Download program into board, connect board to PC by USB cable, open terminal and press k1 to observe terminal information:

```
Start AT91Bootstrap...
-- USB Device Core Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 11 2010 10:51:06 --
```

At the same time, it will notice that USB is found. This program is aimed to enumerate and initialize UDP interface. Then device manger will display corresponding USB device. Refer to figure 3-23:

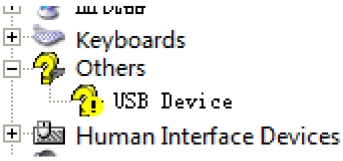


Figure 3-23

Insert USB, LED flashes.

User manual: AT91SAM9G45 Reference Manual.pdf, SP2526A-2EN.pdf

3.7.6 USB Device Hid Transfer Test

Source Location: 04-MDK_Source\23_usb_device_hid_transfer

Program description: This example describes USB HID device transmission, USB HID driver and PIO configuration.

Result: Download program into board, connect board to PC by USB cable. Open terminal and press k1 to observe terminal information:

```
Start AT91Bootstrap...
-- USB Device HID Transfer Project 1.7 --
-- AT91SAM9G45-EK
-- Compiled: Jan 12 2010 17:30:14 --
-W- HIDDTransferDriver_RequestHandler: request 0x0A
-W- Sta 0x8085F400 [0] -W- _
```

At the same time, PC will notice that USB is found, then device manger will display USB device. Refer to figure 3-24:

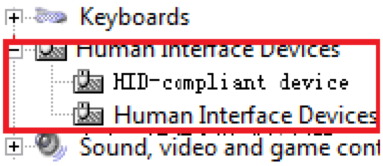
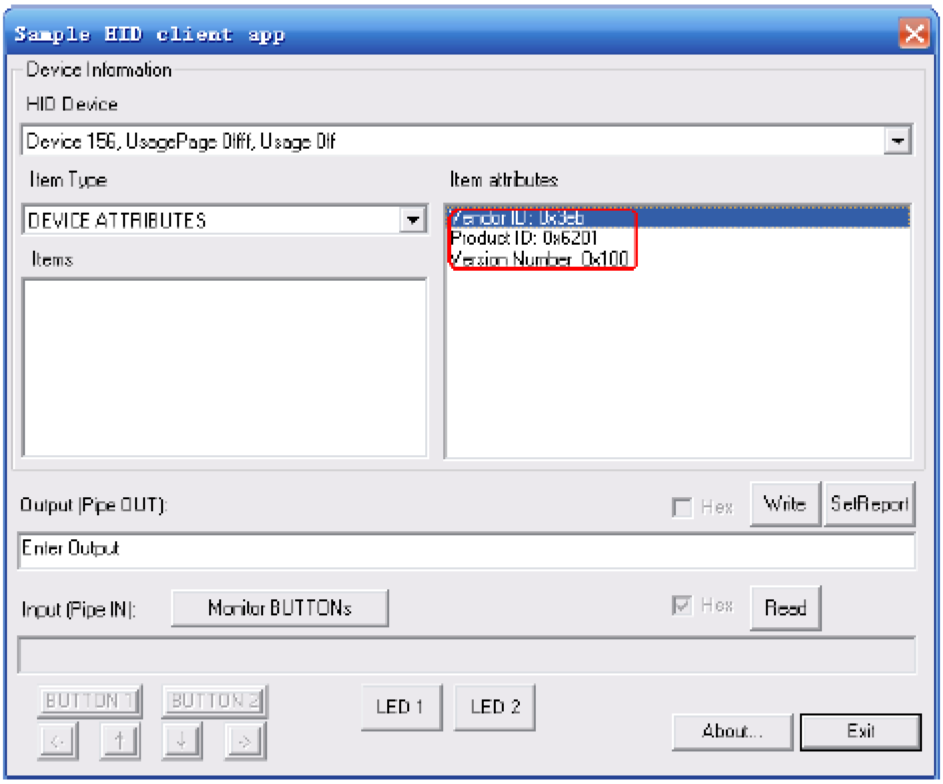


Figure 3-24

Insert USB cable, LED flashes.

Open hidTest.exe to test USB HID.



HID Device: Device 156, UsagePage 0ffff, Usage 0ff

Item Type: DEVICE ATTRIBUTES

Item attributes:
Vendor ID: 0x3eb
Product ID: 0x6201
Version Number 0x100

Output (Pipe OUT): Enter Output

Figure 3-25

Select DEVICE ATTRIBUTES, Input information, and click write button. For example, send message ABCDEFG and click Write, the terminal display information.

```
-W- Sta 0x8085F400 [0] -W- _ Data In(32):
41 42 43 44 45 46 47 00
00 00 00 ……
```

Click Monitor BUTTON to detect device data. Press K3, BUTTON2 changes in hidTest.exe.

User manual: AT91SAM9G45 Reference Manual.pdf, SP2526A-2EN.pdf.

3.7.7 USB Device CDC Serial Test

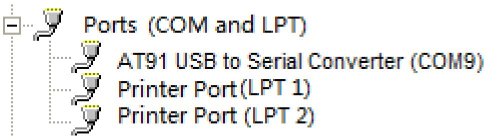
Source Location: 04-MDK_Source\24_usb_device_cdc_serial

Program description: This program describes virtual serial function. More details please refer to source code.

Result: Download program into board, press K1 to reset board, PC will notice prompt.

Location of driver directory: 04-MDK_Source\23_usb_device_cdc_serial\drive. After installing driver, open “port (COM and LPT)” on the left of “Computer Manager”->“Device

Manager", there will be "AT91 USB to Serial Converter (COM9)". Refer to figure 3-26:



Ports (COM and LPT)
AT91 USB to Serial Converter (COM9)
Printer Port(LPT 1)
Printer Port (LPT 2)

Figure 3-26

Open serial 9 and 1 in SSCOM3.2, set serial: Baud Rate: 115200; Data Bits: 8; Parity Bit: None; Stop Bit: 1; Hardware flow: None. Then send character to serial 9 from serial 1, and serial 9 can receive data.

Note: This program use "AT91 USB to Serial Converter (COM9)". Different PC may have different COM and user can select right serial according to host.

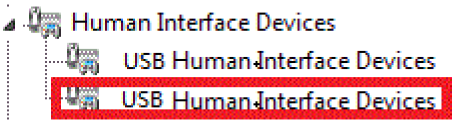
User Manual: AT91SAM9G45 Reference Manual.pdf, SP2526A-2EN.pdf.

3.7.8 USB Device Hid Keyboard Test

Source Location: 04-MDK_Source\25_usb_device_hid_keyboard

Program description: This program describes USB HID device transmission, USB HID driver, PIO configuration, and UDB interface initialization.

Result: Download program into board, press K1 to observe information in terminal. It will notice that "USB HID Keyboard Device" is found and there will be USB Device. Refer to figure 3-27:



Human Interface Devices
USB Human Interface Devices
USB Human Interface Devices

Figure 3-27

Inserting USB, D2 flashes.

Press K3 to control Num Lock light, and D2 flashes.

User manual: AT91SAM9G45 Reference Manual.pdf, SP2526A-2EN.pdf.

Chapter 4 Linux System Guide

4.1 Outline

This chapter describes how to install and run Linux system on MYS-SAM9G45 board. The contents include how to build development environment, compile source codes and download images. Product in the factory has been programed Linux system, and NandFlash Mapping is as follows:

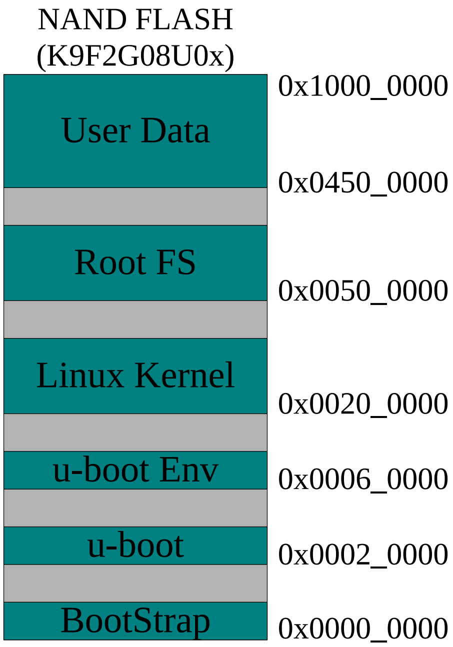


Figure 4-1

(1) BootStrap

After power on system, the first class boot program is copied automatically to internal SRAM and begins to run. The main role is to initialize CPU and external RAM, copy u-boot from NandFlash to the external RAM, and then jump to u -boot entry and start u-boot.

(2) u-boot

Secondary boot program, which can interact with the user, is used for updating kernel image, loading kernel and booting kernel starts.

(3) u-boot Env

Provide u-boot parameters, such as ip address, start command, boot parameters

(4) Linux Kernel

Design Linux 2.6.39 kernel for MYS-SAM9G45.

(5) Root FS

Angstrom-X11 GUI system file.

4.2 Software Resources

Category	Name	Remark
Boot program	Bootstrap	First boot program
	u-boot	Secondary boot program
Linux kernel	Linux 2.6.30	Linux kernel only for MYD-SAM9X5 hardware
Device Drivers	USB Host	USB Host driver supports the mode of OHCI and EHCI transmission
	USB Device	USB Device Driver (Gadget)
	Ethernet	Ethernet driver
	MMC / SD	MMC/SD Card driver
	NandFlash	NandFlash/SmartMedia driver
	TWI(I2C)	I2C driver
	SPI	SPI driver
	AC97	AC97 Audio driver
	LCD Controller	LCD driver, support 4.3 inch, 7 inch，10.2 inch
	RTC	RTC clock driver
	TouchScreen	4 -wire resistive touch screen driver
	PWM	PWM (pulse width modulation) driver
	UART	Serial port driver
	LED	LED driver, include GPIO LED PWM LED driver
System Files	Angstrom-X11	X11 file system with a graphical interface

Table 4-1

4.3 Linux Development Environment Structure

This section, please refer to "VirtualBox's Linux - based development environment to build pdf".

4.4 Installation and Compile

4.4.1 Create work directory

```
# mkdir   /home/MYIR_SAM9G45
# cd   /home/MYIR_SAM9G45
```

Copy 05-Linux_Source folder in CD to /home/MYIR_SAM9X5 (Users can also customize the working directory):

```
# cp   -r   /media/cdrom/05-Linux_SAM9G45   ./
```

4.4.2 Install Cross Compiler Tools

Decompress cross compiler tool to /usr/local directory

```
# tar   xvjf \
05-Linux_Source/CrossTool/   \
arm-2007q1-10-arm-none-linux-gnueabi.tar.bz2   -C   /usr/local
```

4.4.3 Install AT91Bootstrap Source and Compile

Extract Bootstrap-v1.14.zip to working directory：

```
# unzip   05-Linux_Source/AT91Bootstrap/Bootstrap-v1.14.zip
```

After the extraction is completed, compile Bootstrap-v1.14.zip:

```
# cd   Bootstrap-v1.14/board/at91sam9g45ekes/nandflash/
# sudo   make   clean
# sudo   make   \
CROSS_COMPILE=/usr/local/arm-2007q1/bin/arm-none-linux-gnueabi-
```

After completion, nandflash_at91sam9g45ekes.bin is AT91Bootstrap.

Note: AT91Bootstrap is a bootloader for ATMEL chip, which initialize necessary hardware (GPIO Clock, SDRAM, etc), then copy uboot to SDRAM to run.

4.4.4 Install uboot Source and Compile

Decompress u-boot to work directory:

```
# tar   xvjf   05-Linux_Source/u-boot/u-boot-1.3.4.tar.bz2
# cd   u-boot-1.3.4/
```

Compile u-boot:

```
# sudo   make   distclean
# sudo   make   clean
# sudo   make   at91sam9g45ekes_nandflash_config
# sudo   make   \
CROSS_COMPILE=/usr/local/arm-2007q1/bin/arm-none-linux-gnueabi-
```

After completion, u-boot.bin is U -boot that we program.

U-Boot is developed by open source project PPCBoot Development. ARMboot incorporated PPCBoot, collectively known as the U- Boot with some other arch Loader. The first version of the U-Boot-0.2.0 release in December 17, 2002, at the same time, PPCBoot and ARMboot stop maintenance.

After release, U-Boot has been updated several times and the latest version is U-Boot-1.3.4. U-Boot support is persistent. Release URL: http://sourceforge.net/projects/u-boot/.

U-Boot support for the processor architecture, including PowerPC (MPC5xx，MPC8xx，MPC82xx，MPC7xx，MPC74xx，4xx), ARM (ARM7，ARM9，StrongARM，Xscale),MIPS (4Kc，5Kc),x86. U-Boot(Universal Bootloader), As can be Refer ton from the name, it is the most complete resource code under the GPL general the Boot Loader.

U-Boot provides two modes of operation: the boot loader (Boot loading) mode and download mode (Downloading), and has all the features of the large-scale Boot Loader. The main features are:

- Support SCC / FEC Ethernet
- BOOTP/TFTP boot
- IP,MAC preset function
- Online reading and writingFLASH,DOC, IDE,IIC,EEROM,RTC；
- Supports serial port kermit, S-record to download code；
- Identify binary, ELF32, pImage format Image, special support for Linux boot
- Set Monitoring (minitor) command: read and write I/O, memory, registers, memory, peripherals test.
- Support Scripting language (like BASH scripts)
- Support WatchDog, LCD logo, status indication function.

U-Boot function is so powerful, covering most of the processor architecture, providing a large number of peripheral drivers, supportting for multiple file systems, coming with debugging, script guidance tools, in particular to support Linux for board-level transplant a lot of work.

4.4.5 Install Source Code and Compile

(1) Manual compile

Unzip the Linux kernel to the working directory:

```
# tar   xvjf   05-Linux_Source/linux_kernel_2.6.30/mys-sam9g45-linux-2.6.30.tar.bz2
# cd   mys-sam9g45-linux-2.6.30
```

Configure file. (Choose a different configuration file depending LCD size) as shown in

Table 4-2:

LCD Model	Configuration file
LCD_4.3	MYS-SAM9G45_4.3lcd_defconfig
LCD_7.0	MYS-SAM9G45_7.0lcd_defconfig
LCD_10.2	MYS-SAM9G45_10.2lcd_defconfig

Table 4-2

Select appropriate configuration files by actual size and execute following command:

```
# sudo   make   ARCH=arm
```

For example, 4.3 -inch LCD should execute following command:

```
# sudo   make   ARCH=arm   MYS-SAM9G45_4.3lcd_defconfig
```

Enter following command to compile Linux kernel:

```
# sudo   make   uImage   \
  ARCH=arm   \
  CROSS_COMPILE=/usr/local/arm-2007q1/bin/arm-none-linux-gnueabi-
```

(2) Use script to compile automatically

Directly compile make_image.sh script in source root directory:

```
# sudo   chmod   a+x   make_image.sh
```

# ./make_image.sh xx Note: make uImage command requires your Ubuntu system has been installed mkimage tools. Otherwise, use the following command to install the tool:

# **apt-get install uboot-mkimage**

After compile kernel, uImage directory is Linux kernel file in arch/arm/boot/.

4.5 Program Linux Image

4.5.1 Install Download Tool

Here we use samba v2.11 provided by Atmel, specific installation method, please refer to Tools\SAM -BA\sam- ba install.pdf.

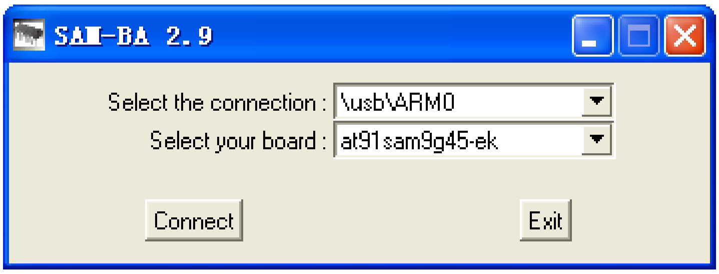
4.5.2 Connect Board and SAM-BA

(1) Install USB driver of MYS-SAM9G45

Refer to 03-Tools\SAM-BA\The Board Driver Installation Guide.pdf。

(2) Connect board

Connect board by USB cable, disconnect JP1, JP2 jumper and then reset board by pressing K1 key. Start software sam-ba v2.9, interface is shown in figure 4-2:



SAM-BA 2.9

Select the connection : \usb\ARM0

Select your board : at91sam9g45-ek

Figure 4-2

Then connect JP1 jumper, and click “Connect”.

4.5.3 Automatical Download

Complete 4.5.1, connect t board by USB cable, disconnect JP1 and JP2, press K1 key to reset board, and then reconnect JP1 jumper. Board identifies PC successfully.

Open CD-ROM directory: 02-Images\ Linux_Image\ MYS-SAM9G45_Linux_4.3_LCD, double click download.bat, SAM-BA will download Linux image automatically to board. Entire download process takes about three minutes. Connect board by serial cable, and reset board to observe Linux start information.

4.5.4 Manual Download

NandFlash demo Memory map, refer to figure 4-3:

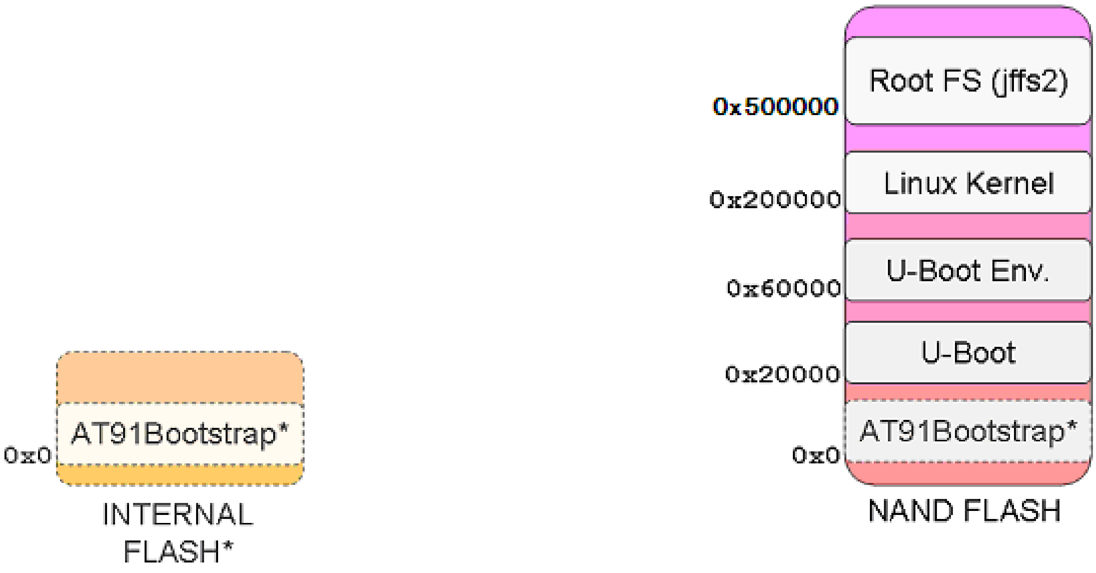
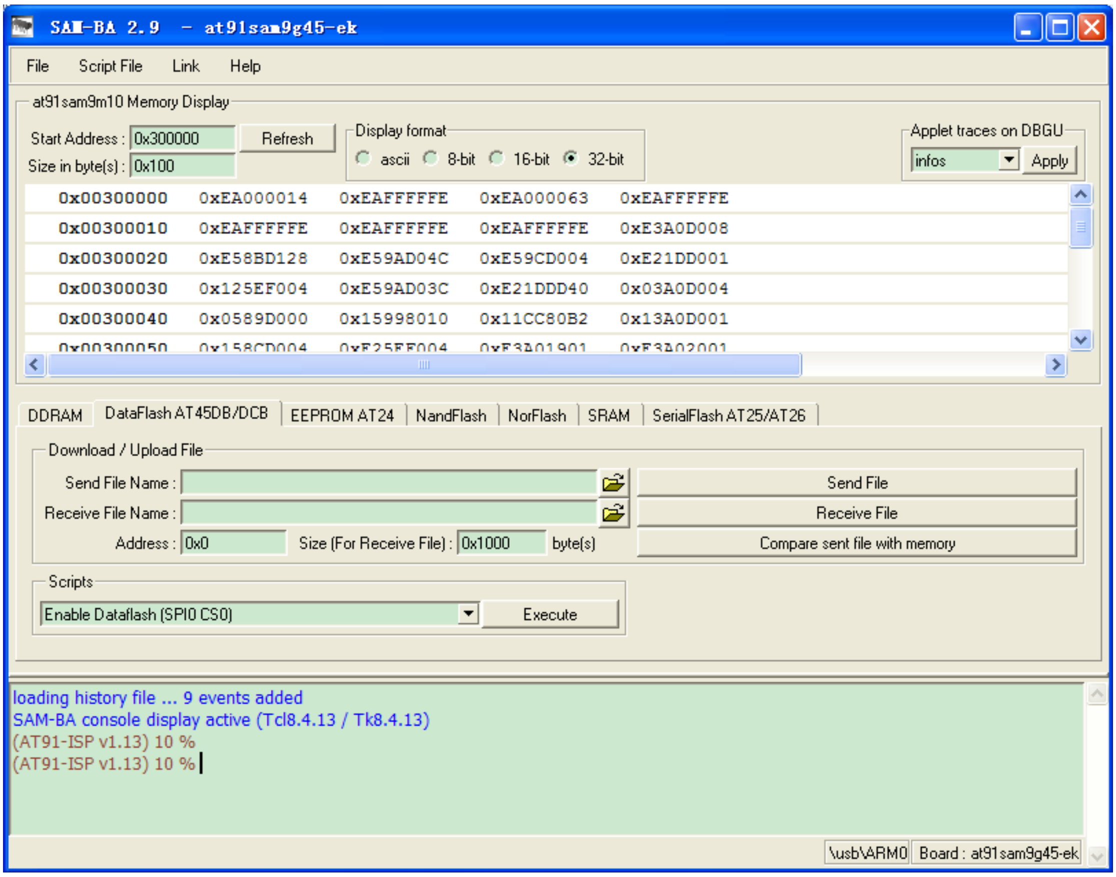


Figure 4-3

Download Linux manually by SAM-BA

(1) Complete chapter 4.5.2, Connect button to enter SAM-BA main interface, refer to figure 4-4:


SAM-BA 2.9 - at91sam9g45-ek
File Script File Link Help
at91sam9m10 Memory Display
Refresh
Display format
Applet traces on DBGU
Apply
DDRAM DataFlash AT45DB/DCB EEPROM AT24 NandFlash NorFlash SRAM SerialFlash AT25/AT26
Download / Upload File
Send File
Receive File
Compare sent file with memory
Scripts
Execute

Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

Send File Name :
Receive File Name :
Address : 0x0 Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable Dataflash (SPI0 CS0)

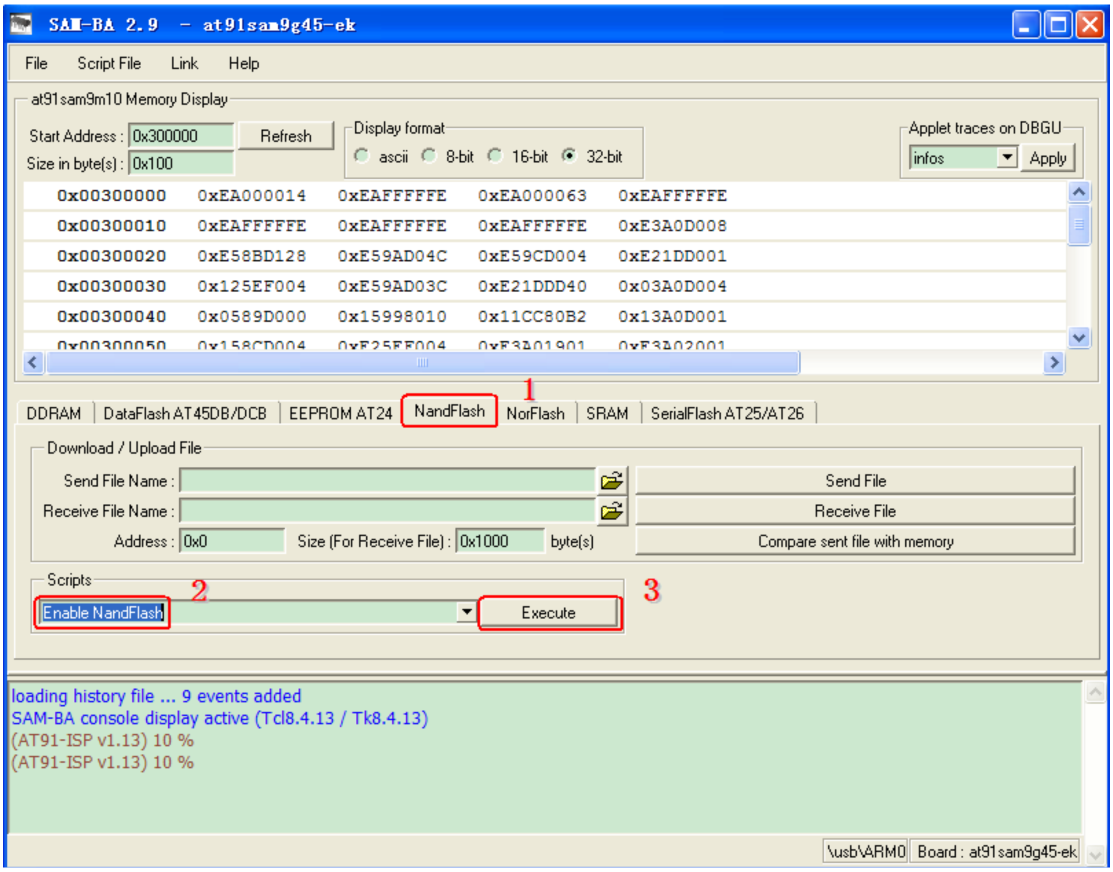
```
loading history file ... 9 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 10 %
(AT91-ISP v1.13) 10 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 4-4

(2) Select "Enable NandFlash" in Scripts tab, and then click "Execute" to Enable

NandFlash. Refer to figure 4-5:



Start Address : 0x300000
Size in byte(s) : 0x100
○ ascii ○ 8-bit ○ 16-bit ● 32-bit
infos

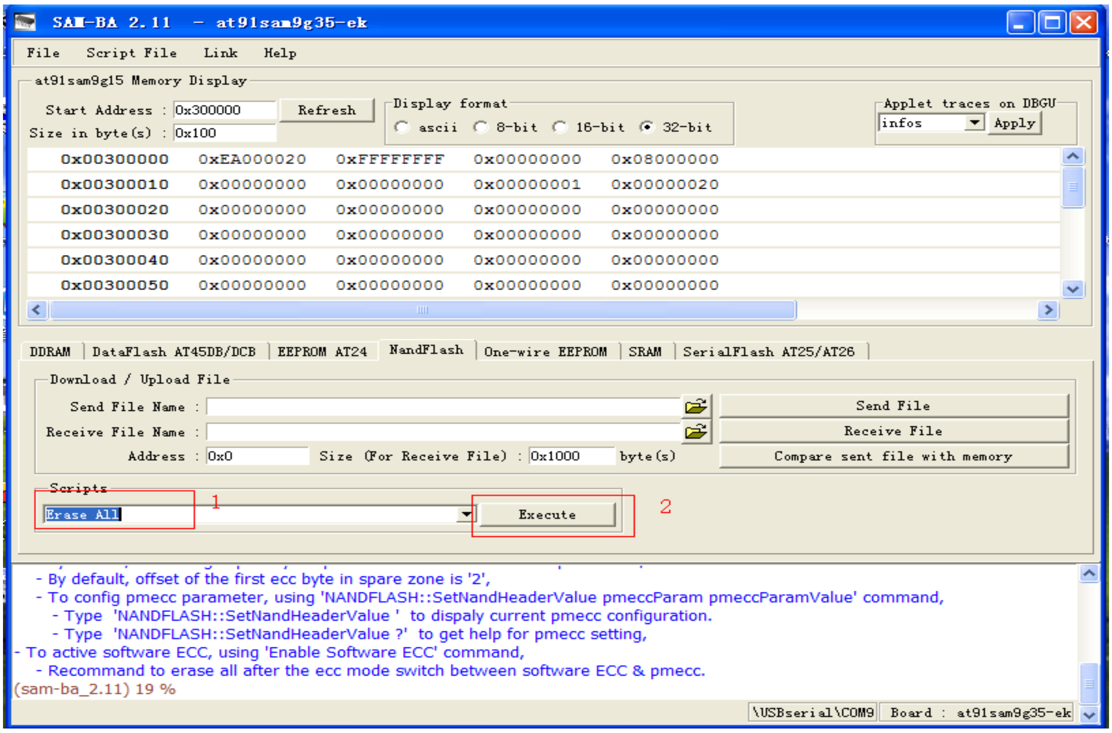
```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE3A01901   0xE3A02001
```

Send File Name :
Receive File Name :
Address : 0x0   Size (For Receive File) : 0x1000 byte(s)
Scripts: Enable NandFlash

```
loading history file ... 9 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 10 %
(AT91-ISP v1.13) 10 %
```

Figure 4-5

(3) Select "Erase All" in Scripts tab, then click "Execute", refer to figure 4-6:



Start Address : 0x300000
Size in byte(s) : 0x100
○ ascii ○ 8-bit ○ 16-bit ● 32-bit
infos

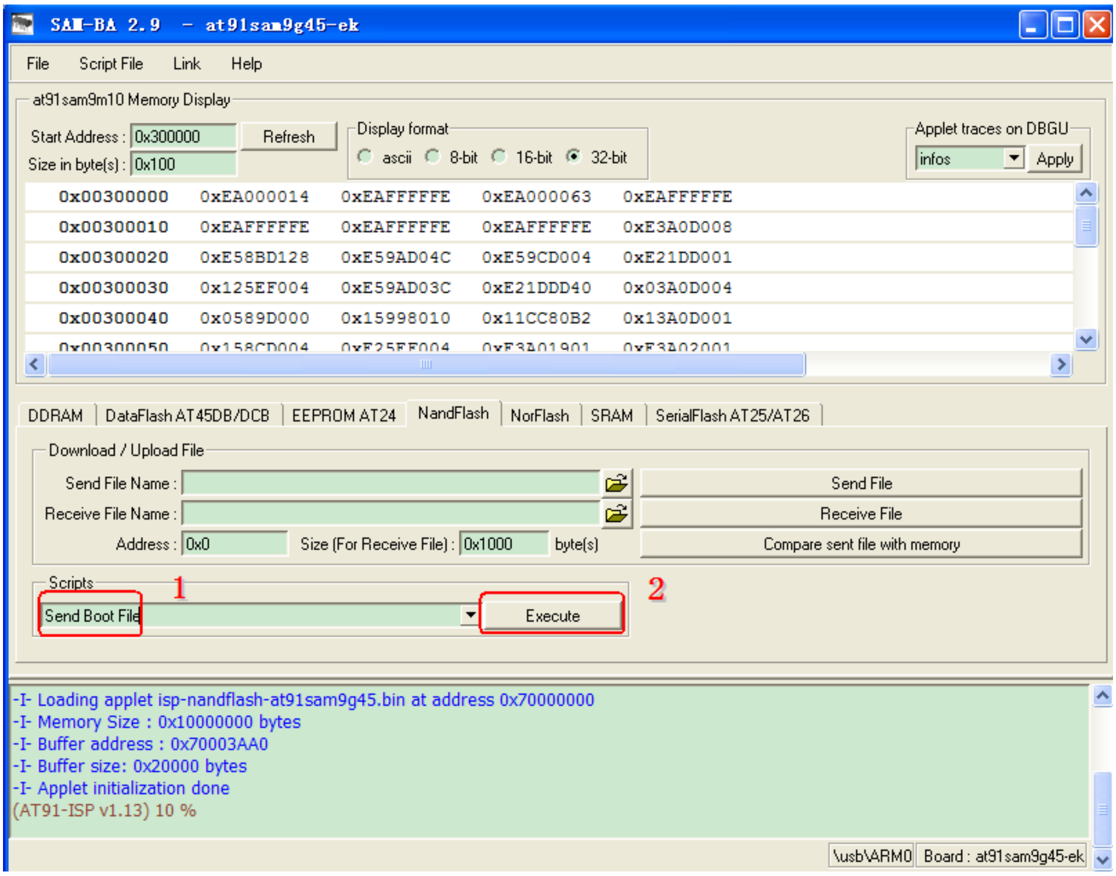
```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x00000000   0x00000000   0x00000001   0x00000020
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
0x00300040   0x00000000   0x00000000   0x00000000   0x00000000
0x00300050   0x00000000   0x00000000   0x00000000   0x00000000
```

Send File Name :
Receive File Name :
Address : 0x0   Size (For Receive File) : 0x1000 byte(s)
Scripts: Erase All

```
   - By default, offset of the first ecc byte in spare zone is '2',
   - To config pmecc parameter, using 'NANDFLASH::SetNandHeaderValue pmeccParam pmeccParamValue' command,
     - Type  'NANDFLASH::SetNandHeaderValue '  to dispaly current pmecc configuration.
     - Type  'NANDFLASH::SetNandHeaderValue ?'  to get help for pmecc setting,
- To active software ECC, using 'Enable Software ECC' command,
   - Recommand to erase all after the ecc mode switch between software ECC & pmecc.
(sam-ba_2.11) 19 %
```

Figure 4-6

(4) Download nandflash_at91sam9g45ekes.bin. Refer to figure 4-7:


SAM-BA 2.9 – at91sam9g45-ek
File Script File Link Help
at91sam9m10 Memory Display
Refresh
Display format
○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU
Apply
DDRAM DataFlash AT45DB/DCB EEPROM AT24 NandFlash NorFlash SRAM SerialFlash AT25/AT26
Download / Upload File
Send File
Receive File
Compare sent file with memory
Scripts
1
Execute
2

Start Address : 0x300000

Size in byte(s) : 0x100

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

Send File Name :

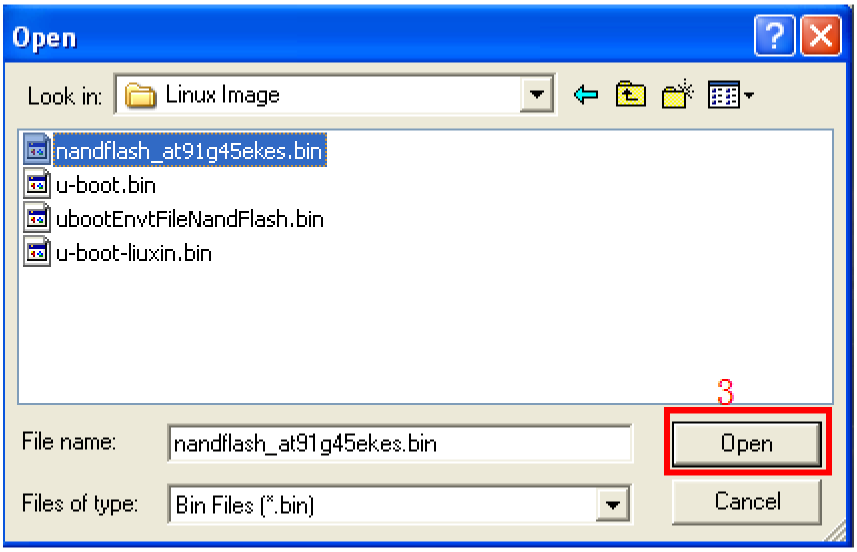
Receive File Name :

Address : 0x0 Size (For Receive File) : 0x1000 byte(s)

Scripts: Send Boot File

```
-I- Loading applet isp-nandflash-at91sam9g45.bin at address 0x70000000
-I- Memory Size : 0x10000000 bytes
-I- Buffer address : 0x70003AA0
-I- Buffer size: 0x20000 bytes
-I- Applet initialization done
(AT91-ISP v1.13) 10 %
```

\usb\ARM0 Board : at91sam9g45-ek



Look in: Linux Image

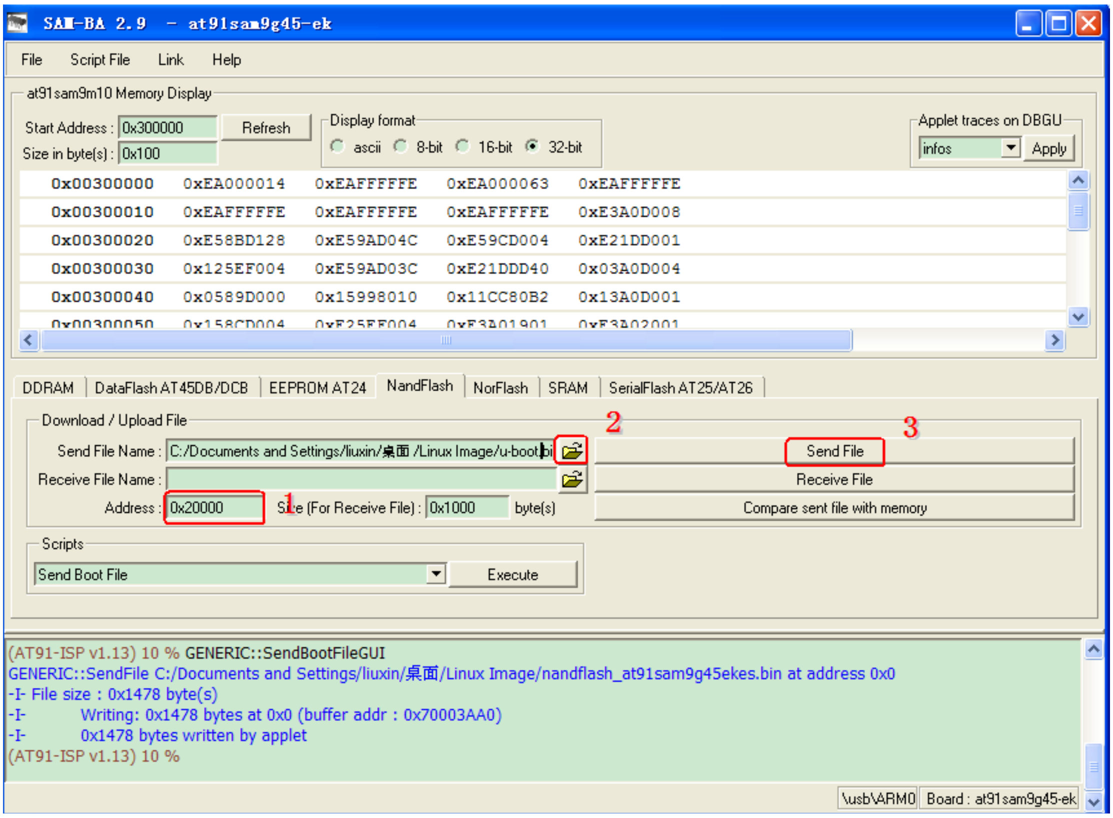
nandflash_at91g45ekes.bin
u-boot.bin
ubootEnvtFileNandFlash.bin
u-boot-liuxin.bin

File name: nandflash_at91g45ekes.bin

Files of type: Bin Files (*.bin)

Figure 4-7

(5) Download u-boot.bin to 0x20000. Refer to figure 4-8:



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

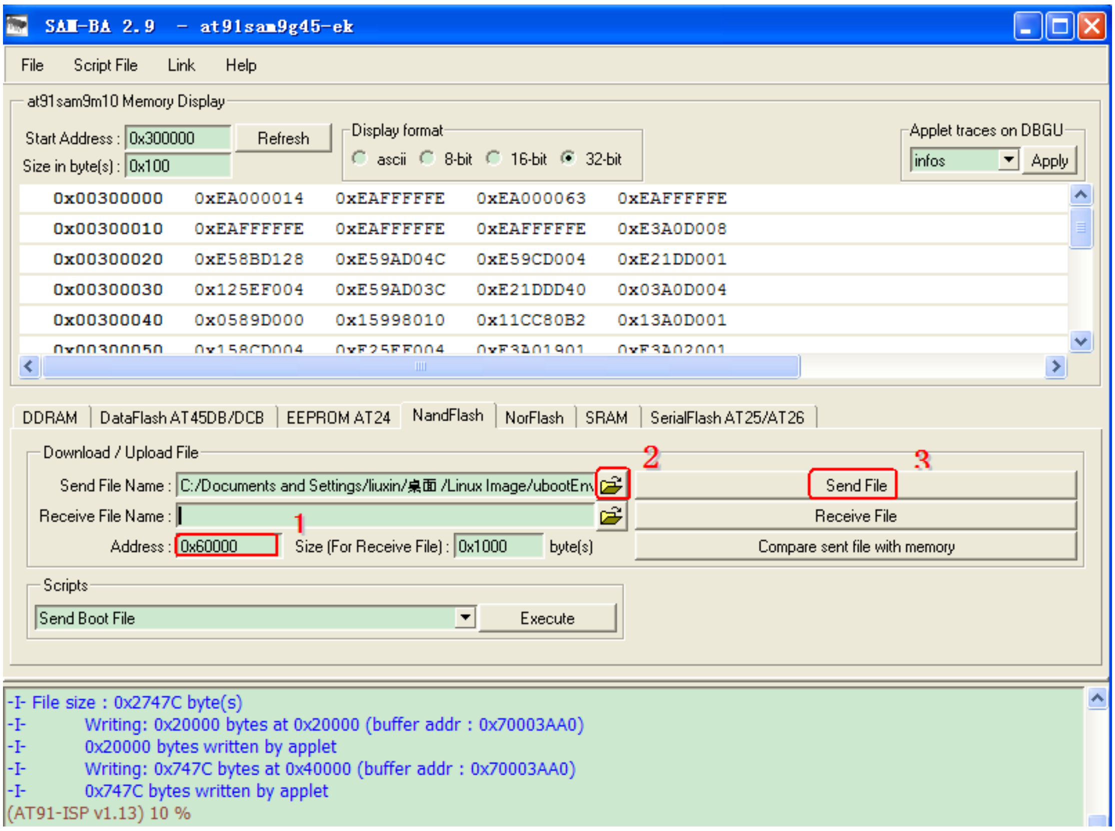
```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE3A01901   0xE3A02001
```

Send File Name : C:/Documents and Settings/liuxin/桌面 /Linux Image/u-boot.bi
Receive File Name :
Address : 0x20000
Size (For Receive File) : 0x1000 byte(s)
Scripts: Send Boot File

```
(AT91-ISP v1.13) 10 % GENERIC::SendBootFileGUI
GENERIC::SendFile C:/Documents and Settings/liuxin/桌面/Linux Image/nandflash_at91sam9g45ekes.bin at address 0x0
-I- File size : 0x1478 byte(s)
-I-         Writing: 0x1478 bytes at 0x0 (buffer addr : 0x70003AA0)
-I-         0x1478 bytes written by applet
(AT91-ISP v1.13) 10 %
```

Figure 4-8

(6) Download ubootEnvtFileNandFlash.bin to 0x60000. Refer to figure 4-9:



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

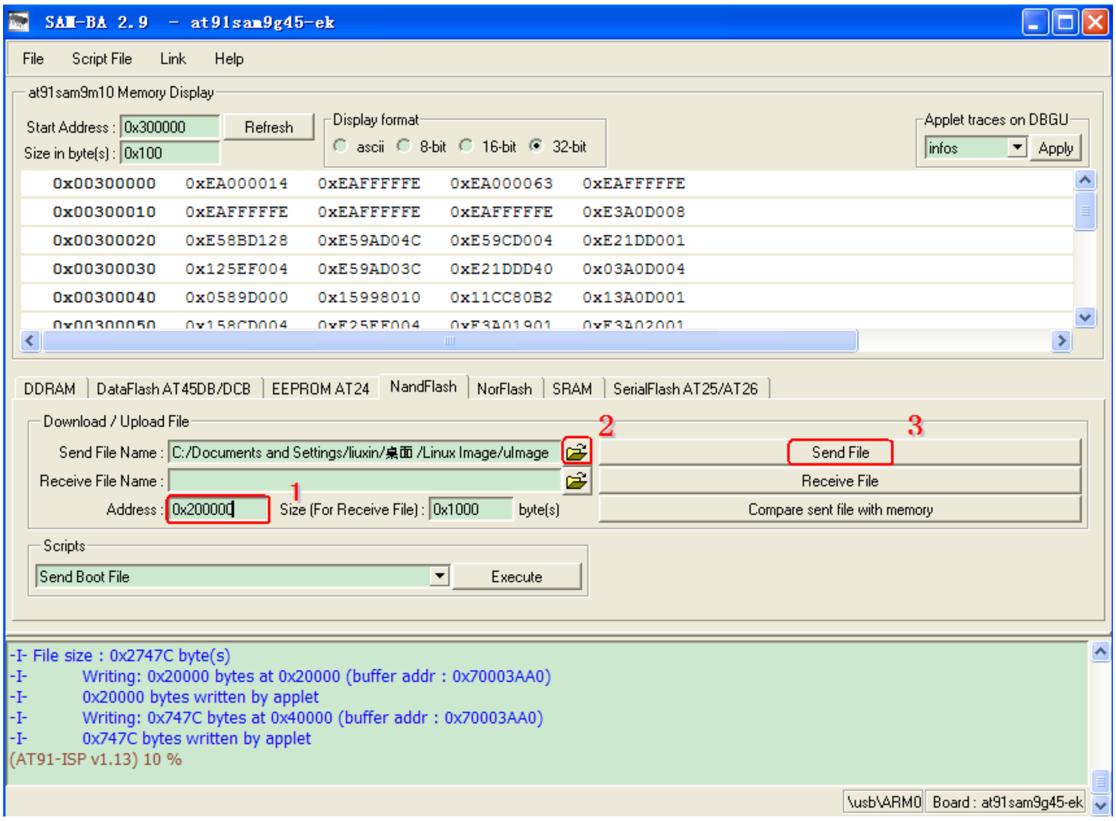
```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE3A01901   0xE3A02001
```

Send File Name : C:/Documents and Settings/liuxin/桌面 /Linux Image/ubootEn
Receive File Name :
Address : 0x60000
Size (For Receive File) : 0x1000 byte(s)
Scripts: Send Boot File

```
-I- File size : 0x2747C byte(s)
-I-         Writing: 0x20000 bytes at 0x20000 (buffer addr : 0x70003AA0)
-I-         0x20000 bytes written by applet
-I-         Writing: 0x747C bytes at 0x40000 (buffer addr : 0x70003AA0)
-I-         0x747C bytes written by applet
(AT91-ISP v1.13) 10 %
```

Figure 4-9

(7) Download Linux kernel uImage to 0x200000. Refer to figure 4-10:



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

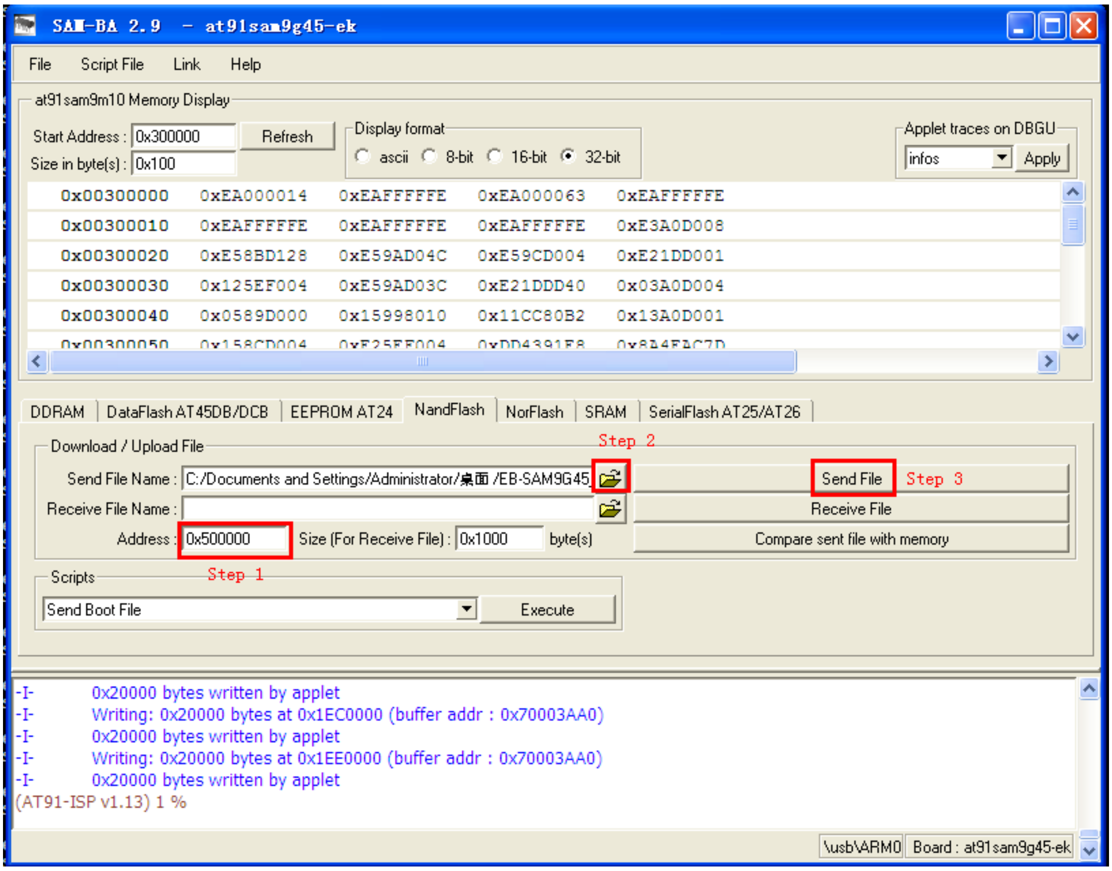
Send File Name : C:/Documents and Settings/liuxin/桌面 /Linux Image/uImage
Receive File Name :
Address : 0x200000
Size (For Receive File) : 0x1000 byte(s)
Scripts: Send Boot File

```
-I- File size : 0x2747C byte(s)
-I-         Writing: 0x20000 bytes at 0x20000 (buffer addr : 0x70003AA0)
-I-         0x20000 bytes written by applet
-I-         Writing: 0x747C bytes at 0x40000 (buffer addr : 0x70003AA0)
-I-         0x747C bytes written by applet
(AT91-ISP v1.13) 10 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 4-10

(8) Download Andstrom-x11-image-demo-glibc-at91.rootfs.jffs2 to 0x500000. Refer to figure 4-11:



Start Address : 0x300000
Size in byte(s) : 0x100
Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xDD4391F8   0x8A4FAC7D
```

Send File Name : C:/Documents and Settings/Administrator/桌面/EB-SAM9G45 — Send File

Receive File Name :

Address : 0x500000 Size (For Receive File) : 0x1000 byte(s)

Scripts: Send Boot File

```
-I-      0x20000 bytes written by applet
-I-      Writing: 0x20000 bytes at 0x1EC0000 (buffer addr : 0x70003AA0)
-I-      0x20000 bytes written by applet
-I-      Writing: 0x20000 bytes at 0x1EE0000 (buffer addr : 0x70003AA0)
-I-      0x20000 bytes written by applet
(AT91-ISP v1.13) 1 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 4-11

4.6 Make Linux File System

Andstrom-x11-image-demo-glibc-at91.rootfs.jffs2 can be made a simple formulation and revision in system file. Here, take helloworld for an example, add an application to file system root directory, show detailed steps of making system file.

4.6.1 Program helloworld

Firstly, write a simple program helloworld:

(1) Create and compile helloworld.c

```
# vi   helloworld.c
```

Enter the following in the helloworld.c, save and exit:

```
#include <stdio.h>

int main(int argc, char *argv[])
{
     int i;
```

```
    printf("========== Hello World ==========\n");
    printf("argc: %d\n", argc);
    for(i = 0; i < argc; i++)
    {
        printf("argv[%d]: %s\n", i, argv[i]);
    }
    return 0;
}
```

(2) Compile helloworld.c

Add cross-compiler tools path to PATH:

```
# export   PATH=$PATH: /usr/local/arm-2007q1/bin/
```

Use cross compiler tool to compile:

```
# arm-none-linux-gnueabi-gcc   -o   helloworld   helloworld.c
```

Helloworld application is generated

4.6.2 Mount jffs2 file system

Jffs2 file system is designed for embedded mobile device design. Mounting jffs2 must have mtd interface. Normal PC does not have mtd manage disk, so here needs mtdram driver. Use RAM space analog mtd equipment and then mount jiffs2. Concrete steps are as follows:

(1) Load jffs2 and mtd driver, including mtdblock and mtdram

Enter following command by turn:

```
# sudo   modprobe   jffs2
# sudo   modprobe   mtdblock
# sudo   modprobe   mtdram   total_size=65536   erase_size=128
```

Need to pass a few parameters to loaded mtdram:

- total_size   Analog mtd partition size, units of KB
- erase_size   mtd partition erase block size, units of KB. Usually it is associated with the BlockSize of nandflash in target board. the BlockSize of NANDFLASH is 128KB in MYS-the SAM9G45.

If executed successfully .we can find mtd0 and mtd0r0 devices in /dev/directory:

```
# ls   /dev/mtd*
/dev/mtd0   /dev/mtd0ro   /dev/mtdblock0
```

(2) Load Andstrom-x11-image-demo-glibc-at91.rootfs.jffs2 into new mtd0 partition.

Erase mtd0:

```
# sudo   flash_eraseall   /dev/mtd0
```

Load by dd command, as follows:

```
# sudo   dd   if= Angstrom-x11-image-demo-glibc-at91.rootfs.jffs2   of=/dev/mtd0
53248+0 records in
53248+0 records out
27262976 bytes (27 MB) copied, 0.455085 s, 59.9 MB/s
```

(3) Load jffs2

After completing the above step, we can mount jffs0 file system like mounting ordinary mtd device. Create a new mount point:

```
# mkdir   fsmount
```

Mount by the following command:

```
# sudo   mount   -t   jffs2   /dev/mtdblock0   fsmount/
# ls   fsmount/
bin   boot   dev etc   home   lib   media   mnt   proc   sbin   sys   tmp   usr   var
```

Now that, we have mounted jffs2 system file successfully.

4.6.3 Modify jffs2 System Files

After mount jffs2 file system successfully, it can modify file content. It should add compiled demo helloworld to root directory. The operation is as follows:

```
# sudo   cp   helloworld   fsmount/
# sync
# ls   fsmount
bin boot dev etc helloworld home lib media   mnt   proc   sbin   sys   tmp   usr   var
```

4.6.4 Regenerate jffs2 System File

Regenerate file system by mkfs.jffs2 tool. Using the following command if not install mkfs.jffs2 tools:

```
# sudo   apt-get   install   mtd-utils
```

Enter the following command to generate a new jffs2 file system:

```
# sudo   mkfs.jffs2   -n   -l   -s   0x800   -e   0x20000   -p   0x4000000   \
  -d   fsmount/   -o   rootfs.jffs2
# sync
```

mkfs.jffs2 Parameter Description:

-n   Not write "CLEANMARKER" at the beginning of each block. NANDFLASH has

CLEANMAKER

-l Little-endian format (little-ending)

-s Specify page size, 0x800 (2KB)

-e Specify erase block size. The block size of NANDFLASH is 0x20000 (128KB)

-p Pre-filled size is 0x4000000 (64MB).

-d Specify the enter directory of generated file system

-o Specify the output file

After completion, download fsimage.ubi file to 0x500000 by chapter 4.5.4.

After download, reset board and input root to login, there will be newly added helloworld file in the root directory:

```
at91sam login: root
root@at91sam:~$ cd /
root@at91sam:/$ ls
bin           etc           lib           proc           tmp
boot          helloworld    media         sbin           usr
dev           home          mnt           sys            var
```

Run helloworld, as follows:

```
root@at91sam:/$ ./helloworld
========== Hello World ==========
argc: 1
argv[0]: ./helloworld
root@at91sam:/$
```

4.7 Linux Use

Run Linux system, and operate it by touch screen or serial. The following describes operate Linux, such as mount U disk, SD card, network interface test and music test.

4.7.1 Touch Screen Calibration

After download, boot will run touch screen calibration procedure automatically (after calibration is complete, the boot will no longer run automatically), then LCD screen will show five points in turn. Press corresponding calibration point to calibrate touch screen, If pass the calibration, it will enter Linux system GUI interface. After booting Linux, enter "root" to log:

```
at91sam login:root
```

4.7.2 Disk Use

(1) Enter Linux by terminal, U disk is inserted to any of a USB host port, and there will be the follow information:

```
usb 1-2: new high speed USB device using atmel-ehci and address 3
usb 1-2: New USB device found, idVendor=1005, idProduct=b113
usb 1-2: New USB device strings: Mfr=1, Product=2, SerialNumber=3
usb 1-2: Product: USB FLASH DRIVE
usb 1-2: Manufacturer:
usb 1-2: SerialNumber: 19891C540920
usb 1-2: configuration #1 chosen from 1 choice
scsi1 : SCSI emulation for USB Mass Storage devices
scsi 1:0:0:0: Direct-Access USB FLASH DRIVE PMAP PQ: 0 ANSI: 0 CCS
sd 1:0:0:0: [sda] 7831552 512-byte hardware sectors: (4.00 GB/3.73 GiB)
sd 1:0:0:0: [sda] Write Protect is off
sd 1:0:0:0: [sda] Assuming drive cache: write through
sd 1:0:0:0: [sda] Assuming drive cache: write through
sda: sda1
sd 1:0:0:0: [sda] Attached SCSI removable disk
```

(2) System will mount U disk automatically to /media/sdaX (X will be different in different U disk mount directory. find x in output information in the serial by insertting U disk.), entering the follow command to view U disk contents.

```
root@at91sam:~$ cd /media
root@at91sam:/media$ ls
card        hdd         mmcblk0    ram          sda1
cf          mmc1        net        realroot   union
root@at91sam:/media$ cd sda1
```

(3) Enter LS command to view U disk contents

```
root@at91sam:/media/sda1$ ls
```

4.7.3 SD Card Use

(1)Insert MicroSD card to MicroSD card interface and system will automatically mount it.

(2) When MicroSD card is inserted, HyperTerminal displays SD card information:

```
root@at91sam:~$ mmc0: host does not support reading read-only switch. assuming
write-enable.
mmc0: new SD card at address e624
mmcblk0: mmc0:e624 SU02G 1.84 GiB
```

(3) Enter MicroSD card mount directory by the following commands:

```
root@at91sam:~$ cd /media/mmcblk0/
```

(4) Enter the LS command to view MicroSD card content:

```
root@at91sam:/media/mmcblk0$ ls
01 - What Are Words.mp3      Epson_LQ630K_PrinterDriver_60vista.exe
GoneNutty.mp4                MobileQQ_2012_v1_0.apk          arm_ip
c_1024_768_3079.jpg          c_1024_768_3081.jpg
c_1024_768_3094.jpg          cantest                         cantest.x
```

4.7.4 Play MP3 Music

Before playing music, it needs to connect headphones or stereo to J7. U disk storages an mp3 music and is inserted into USB interface. Next, enter SD card by chapter 4.3.3.

play music by mplayer command

```
root@at91sam:/media/mmcblk0p1$ mplayer demo.mp3
```

At this point, it can hear music from headphones. Enter Ctrl + C to end playing music:

4.7.5 Network port test

Connect board to PC with a crossover cable, or to connect the board to a switch or router by straight through cable. Note that for a PC using a crossover cable to connect the way, if development boards need to access the external network ,it requires the PC side has dual network cards, one card connected to development board, the other card connected to the external network and the two card set to bridge mode. The following test is in the cross-link + PC NIC bridged mode. ⑴ the PC's "Network Connections" window, select the need to bridge two NICs, right-click and select "bridge" so the two network cards will be bridged.

(1) View current network configuration information by Ifconfig eth0 command, as follows:

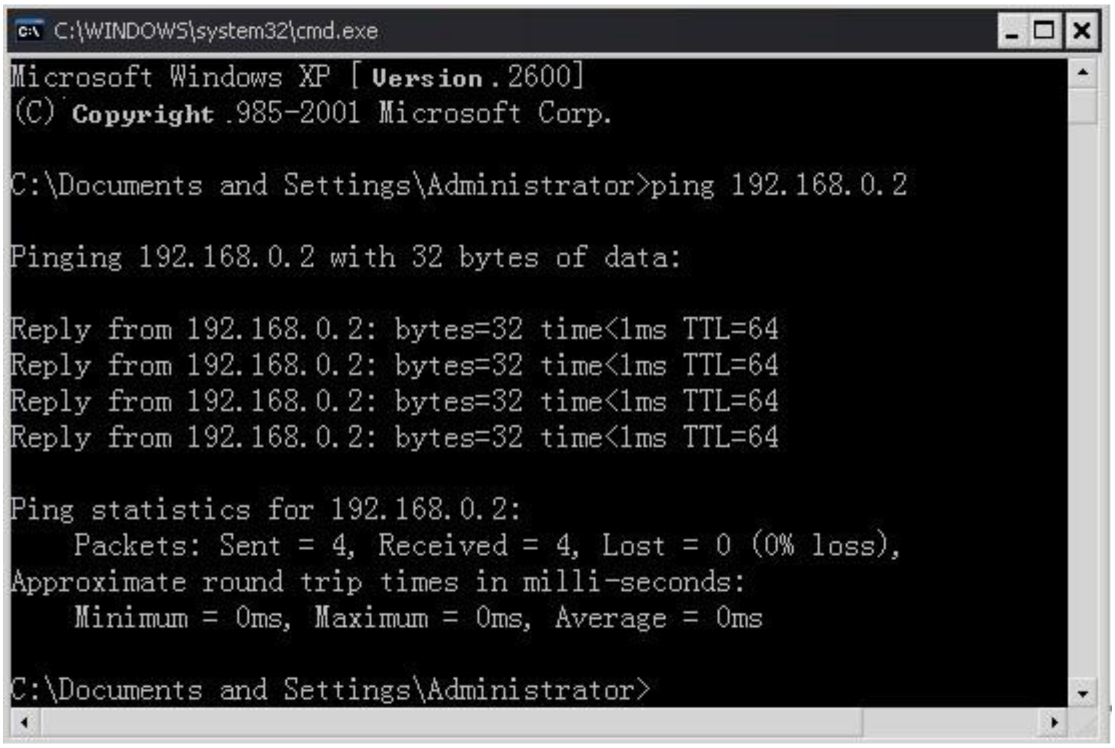
```
root@at91sam:/media/mmcblk0$ ifconfig eth0
```

```
eth0      Link encap:Ethernet   HWaddr 3A:1F:34:08:54:54
          BROADCAST MULTICAST   MTU:1500   Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)   TX bytes:0 (0.0 B)
          Interrupt:25 Base address:0xc000
```

(2) Configure IP address by Ifconfig command: 192.168.0.2, as shown below:

```
root@at91sam:/media/mmcblk0$ ifconfig eth0 192.168.0.2
root@at91sam:/media/mmcblk0$ ifconfig eth0
eth0       Link encap:Ethernet   HWaddr 3A:1F:34:08:54:54
           inet addr:192.168.0.2   Bcast:192.168.0.255   Mask:255.255.255.0
           UP BROADCAST MULTICAST   MTU:1500   Metric:1
           RX packets:0 errors:6 dropped:0 overruns:0 frame:0
           TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:0 (0.0 B)   TX bytes:1797 (1.7 KiB)
           Interrupt:25 Base address:0xc000
```

(3) Ping board. Refer to figure 4-12:



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>ping 192.168.0.2

Pinging 192.168.0.2 with 32 bytes of data:

Reply from 192.168.0.2: bytes=32 time<1ms TTL=64
Reply from 192.168.0.2: bytes=32 time<1ms TTL=64
Reply from 192.168.0.2: bytes=32 time<1ms TTL=64
Reply from 192.168.0.2: bytes=32 time<1ms TTL=64

Ping statistics for 192.168.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Documents and Settings\Administrator>
```

Figure 4-12

4.7.6 Telnet test

(1) View by Ifconfig eth0 command, as follows:

```
root@at91sam:/$ ifconfig eth0
eth0      Link encap:Ethernet   HWaddr 3A:1F:34:08:54:54
          UP BROADCAST RUNNING MULTICAST   MTU:1500   Metric:1
          RX packets:55 errors:6 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7943 (7.7 KiB)   TX bytes:2177 (2.1 KiB)
          nterrupt:25 Base address:0xc000
```

(2) Configure board ip address as 192.168.0.2 by actual situation and enter the following command:

```
root@at91sam:~$ ifconfig eth0 192.168.0.2
root@at91sam:/$ ifconfig
eth0        Link encap:Ethernet   HWaddr 3A:1F:34:08:54:54
            inet addr:192.168.0.2   Bcast:192.168.0.255   Mask:255.255.255.0
            UP BROADCAST RUNNING MULTICAST   MTU:1500   Metric:1
            RX packets:58 errors:6 dropped:0 overruns:0 frame:0
            TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:8684 (8.4 KiB)   TX bytes:2177 (2.1 KiB)
            Interrupt:25 Base address:0xc000
```

(3) Configure Gateway

Firstly, test network is connected successfully by ping command, as follows:

```
root@at91sam:/$ ping 192.168.0.1
PING 192.168.0.1 (192.168.0.1): 56 data bytes
64 bytes from 192.168.0.1: icmp_seq=0 ttl=64 time=9.7 ms
64 bytes from 192.168.0.1: icmp_seq=1 ttl=64 time=0.3 ms
64 bytes from 192.168.0.1: icmp_seq=2 ttl=64 time=0.3 ms
```

Then use route add default gw command to set default gateway (by actual situation): 192.168.0.1, the operation is as follows:

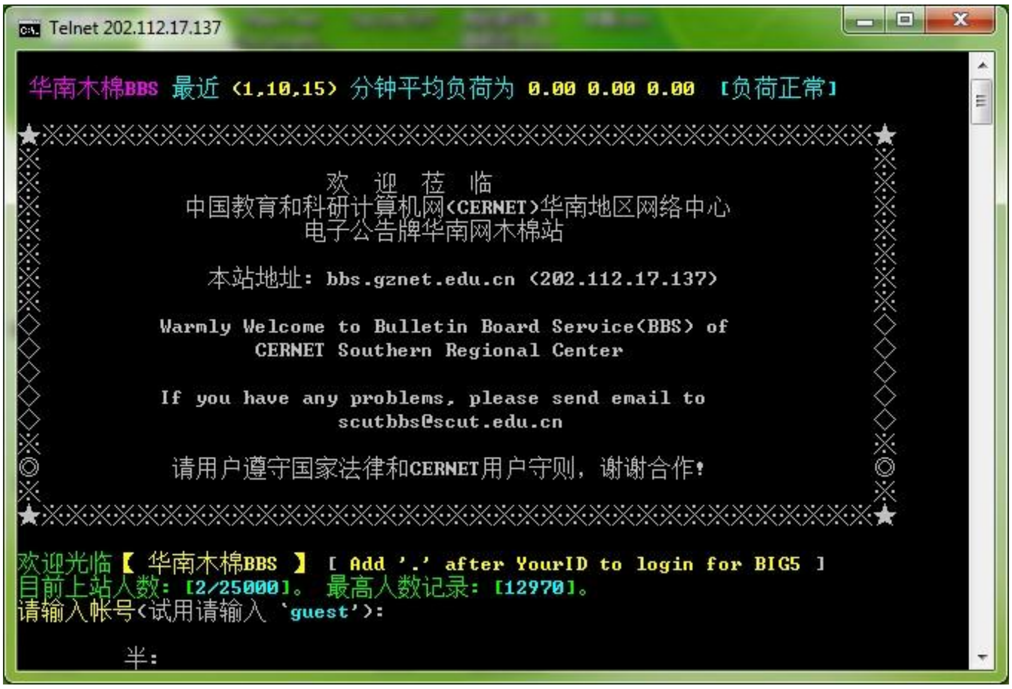
```
root@at91sam:/$ route add default gw 192.168.0.1
```

Ping 202.112.17.137 to confirm connectivity, operating as follows:

```
root@at91sam:/$ ping 202.112.17.137
PING 202.112.17.137 (202.112.17.137): 56 data bytes
64 bytes from 202.112.17.137: icmp_seq=0 ttl=52 time=28.9 ms
64 bytes from 202.112.17.137: icmp_seq=1 ttl=52 time=117.7 ms
64 bytes from 202.112.17.137: icmp_seq=2 ttl=52 time=210.6 ms
```

(4) Use telnet to access BBS forum:

```
root@at91sam:/$ telnet 202.112.17.137
```



Telnet 202.112.17.137

```
华南木棉BBS 最近 (1,10,15) 分钟平均负荷为 0.00 0.00 0.00  [负荷正常]

                    欢  迎  莅  临
        中国教育和科研计算机网(CERNET)华南地区网络中心
                 电子公告牌华南网木棉站

          本站地址: bbs.gznet.edu.cn (202.112.17.137)

        Warmly Welcome to Bulletin Board Service(BBS) of
                CERNET Southern Regional Center

        If you have any problems, please send email to
                    scutbbs@scut.edu.cn

         请用户遵守国家法律和CERNET用户守则，谢谢合作!

欢迎光临【 华南木棉BBS 】 [ Add '.' after YourID to login for BIG5 ]
目前上站人数: [2/25000]。 最高人数记录: [12970]。
请输入帐号(试用请输入 `guest'):

     半:
```

Telnet test is successful.

(5) Configure DNS to access external network (specific DNS settings by actual situation).

① View DNS server address by ipconfig/all command

② Use touch to create /etc/resolv.conf file and write DNS into resolv.conf file, as shown:

```
root@at91sam:/$ touch /etc/resolv.conf
root@at91sam:/$ echo nameserver 202.103.24.68 > /etc/resolv.conf
```

③ Ping www.baidu.com:

```
root@at91sam:/$ ping www.baidu.com
PING www.a.shifen.com (220.181.111.148): 56 data bytes
64 bytes from 220.181.111.148: icmp_seq=0 ttl=53 time=58.9 ms
64 bytes from 220.181.111.148: icmp_seq=1 ttl=53 time=55.9 ms
64 bytes from 220.181.111.148: icmp_seq=2 ttl=53 time=59.5 ms
64 bytes from 220.181.111.148: icmp_seq=3 ttl=53 time=55.7 ms
64 bytes from 220.181.111.148: icmp_seq=4 ttl=53 time=61.1 ms
64 bytes from 220.181.111.148: icmp_seq=5 ttl=53 time=61.3 ms
64 bytes from 220.181.111.148: icmp_seq=6 ttl=53 time=56.1 ms
```

Access extranet successfully.

4.7.7 RTC Use

(1) Install button battery to board.

(2) This system will set initial value at first start time, so it needs to set time after system startup.

Set system time: Note that setup time must be set to hardware clock (RTC)

```
root@at91sam:~$ date -s 2012.03.15-10:36:00 ; hwclock –w
```

Query time: the following command to query system time and hardware RTC time.

```
root@at91sam:~$ date
root@at91sam:~$ hwclock -r
```

4.8 Linux driver development examples

This section describes a simple character device driver development, achieving the function to control LED lights.

4.8.1 Hardware schematic

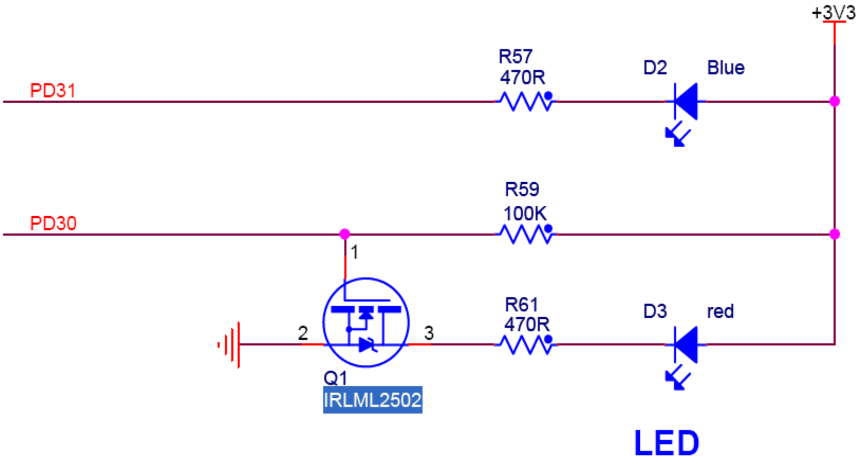


Figure 4-13

Use PD30 interface control red D3 by IRLML2502. When it is high, LED turns on. Port PD31 control blue D2. When it is low, LED turns on. Refer to figure 4-13:

4.8.2 Driver source code

(1) Create driver file in kernel

Create drive files:

```
# cd   linux-2.6.39
# vi   drivers/char/ledtest.c
```

(2) Driver source code ledtest.c is as follows:

```
#include <linux/string.h>
#include <linux/cdev.h>
#include <linux/fs.h>
#include <mach/gpio.h>
#include <linux/device.h>

#define DEVICE_NAME       "MYS-SAM9G45-ledtest"

static int LED_Major = 0;
struct cdev cdev;

#define   LED_OFF        0
#define   LED_ON         1

static unsigned long led_table [] =
{
        AT91_PIN_PD31,   /**led_blue**/
        AT91_PIN_PD30,   /**led_red**/
};
static int MYS_SAM9G45_ledtest_open(struct inode *inode, struct file *file)
{
        printk("MYS-SAM9G45-ledtest Driver Open Called!\n");
        return 0;
}

static long MYS_SAM9G45_ledtest_ioctl(struct file *filp, unsigned int cmd, unsigned long arg)
{
        if((cmd != 1 && cmd != 0) || (arg != 1 && arg != 0))
                return -1;

        switch(cmd)
        {
                case LED_ON:
                        if(arg)
                        {
                                at91_set_gpio_value(led_table[arg], 1);
                        }
                        else
                        {
                                at91_set_gpio_value(led_table[arg], 0);
```

```
            }
            break;
        case LED_OFF:
            if(arg)
            {
                at91_set_gpio_value(led_table[arg], 0);
            }
            else
            {
                at91_set_gpio_value(led_table[arg], 1);
            }
            break;
        default:
            return -EINVAL;
    }
    return 0;
}

static int MYS_SAM9G45_ledtest_release(struct inode *inode, struct file *file)
{
    printk("MYS_SAM9G45_LED Driver Release Called!\n");
    return 0;
}
static struct file_operations MYS_SAM9G45_ledtest_fops =
{
    .owner              =    THIS_MODULE,
    .open               =    MYS_SAM9G45_ledtest_open,
    .release            =    MYS_SAM9G45_ledtest_release,
    .unlocked_ioctl     =    MYS_SAM9G45_ledtest_ioctl,
};

static struct class *MYS_SAM9G45_ledtest_class = NULL;
static int __init MYS_SAM9G45_ledtest_init(void)
{
    int result,err;
    dev_t devno   =   MKDEV(LED_Major, 0);

    if (LED_Major)
    {
        result = register_chrdev_region(devno, 1, DEVICE_NAME);
        printk("Got the Major number by register_chrdev_region !\n ");
    }
    else
    {
```

```
                result = alloc_chrdev_region(&devno, 0, 1, DEVICE_NAME);
                LED_Major=MAJOR(devno);
                printk("Got the Major number by alloc_chrdev_region !\n");
        }

        if (result < 0)
        {
                printk(DEVICE_NAME " can't register major number\n");
                return result;
        }

        printk("register MYS_SAM9G45_ledtest Driver OK! Major = %d\n", LED_Major);
        cdev_init(&cdev,&MYS_SAM9G45_ledtest_fops);
        cdev.owner=THIS_MODULE;
        cdev.ops=&MYS_SAM9G45_ledtest_fops;

        err=cdev_add(&cdev, MKDEV(LED_Major, 0), 1);
        if (err)
        {
                printk("error %d adding led \n ", err);
                goto fail_cdev_add;
        }
                MYS_SAM9G45_ledtest_class = class_create(THIS_MODULE,
DEVICE_NAME);
        if(IS_ERR(MYS_SAM9G45_ledtest_class))
        {
                printk("Err: failed in MYS_SAM9G45_ledtest class. \n");
                goto fail_create_class;
        }
        device_create(MYS_SAM9G45_ledtest_class, NULL, MKDEV(LED_Major, 0),
NULL, DEVICE_NAME);
        at91_set_gpio_output(AT91_PIN_PD31, 1);
        at91_set_gpio_output(AT91_PIN_PD30, 1);

        at91_set_deglitch(AT91_PIN_PD31, 1);
        at91_set_deglitch(AT91_PIN_PD30, 1);

        printk(DEVICE_NAME " initialized\n");
        return 0;

    fail_create_class:
        cdev_del(&cdev);
    fail_cdev_add:
        unregister_chrdev_region(devno, 1);
        return -1;
    }
    static void __exit MYS_SAM9G45_ledtest_exit(void)
    {
        printk("MYS_SAM9G45 LED DRIVER MODULE EXIT\n");
        device_destroy(MYS_SAM9G45_ledtest_class, MKDEV(LED_Major, 0));
        class_destroy(MYS_SAM9G45_ledtest_class);
        cdev_del(&cdev);
        unregister_chrdev(LED_Major, DEVICE_NAME);
    }

    module_init(MYS_SAM9G45_ledtest_init);
    module_exit(MYS_SAM9G45_ledtest_exit);

    MODULE_LICENSE("GPL");
    MODULE_AUTHOR("Alvin");
    MODULE_DESCRIPTION("This is an example of MYS_SAM9G45_LEDTEST drivers");
    MODULE_ALIAS("A simplest module.");
```

4.8.3 Compile Driver

(1) Modify Kconfig and Makefile in driver/char/directory.

① Kconfig

Use vi editor to open Kconfig file:

```
# vi drivers/char/Kconfig
```

The last of document (before endmenu) plus the following:

```
    config LEDTEST
        tristate "ledtest for MYS-SAMG45"
        default n
        help
          this is a driver for MYS-SAM9G45
```

Then save it and exit.

② Makefile

Use vi editor to open t Makefile:

```
# vi drivers/char/Makefile
```

At the end of file add the following:

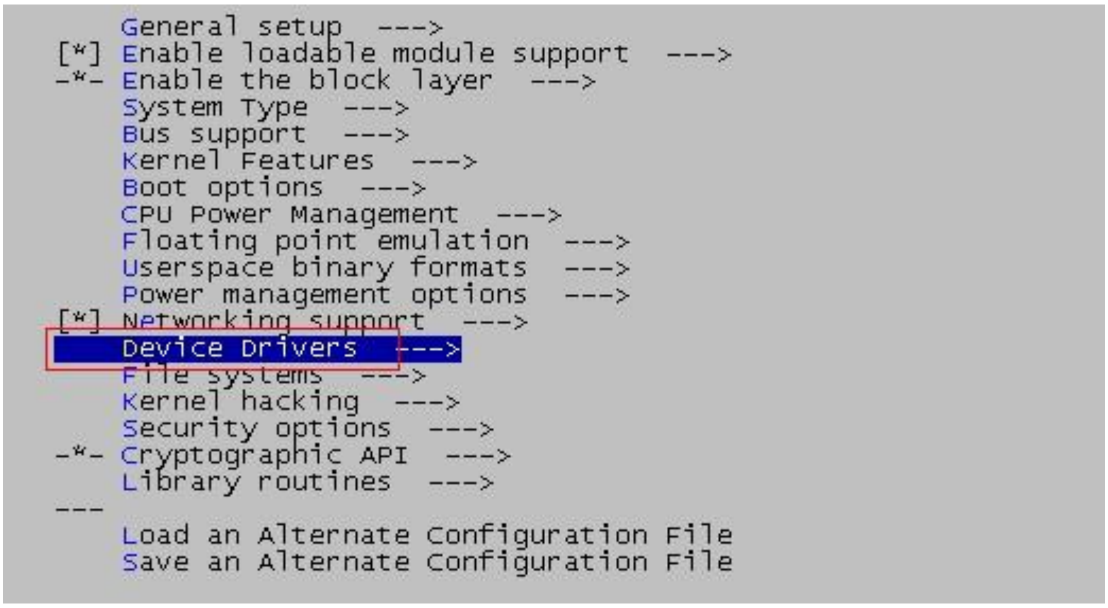
```
obj-$(CONFIG_LEDTEST)                += ledtest.o
```

Then save it and exit.

(2) Configure driver as module to compile:

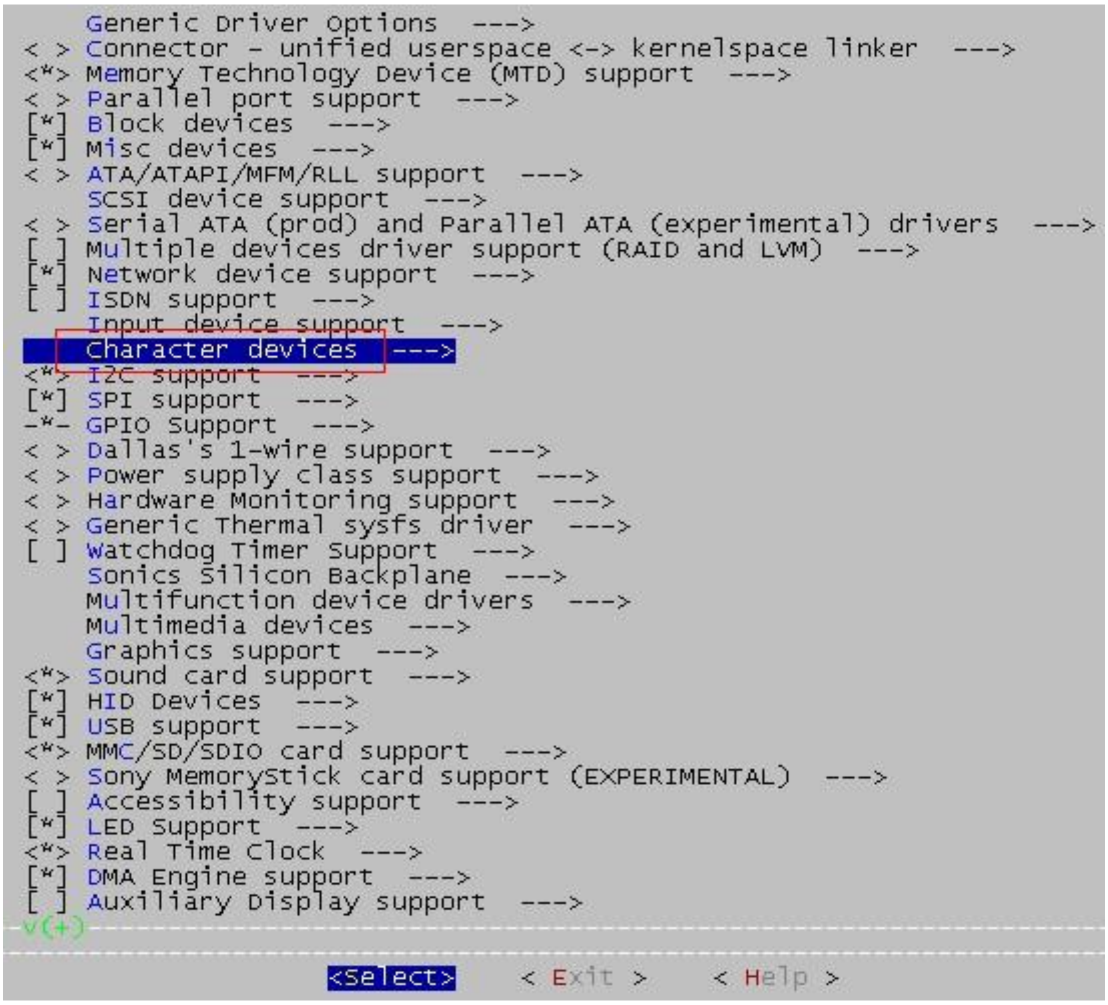
```
# make    ARCH=arm    menuconfig
```

Select Device Drivers---> Character devices---> the <M> ledtest for MYD- SAM9G45, and then press M which is said as a module. Specific operating screenshot is in figure 4-14, 4-15 and 4-16:



```
    General setup  --->
[*] Enable loadable module support  --->
-*- Enable the block layer  --->
    System Type  --->
    Bus support  --->
    Kernel Features  --->
    Boot options  --->
    CPU Power Management  --->
    Floating point emulation  --->
    Userspace binary formats  --->
    Power management options  --->
[*] Networking support  --->
    Device Drivers  --->
    File systems  --->
    Kernel hacking  --->
    Security options  --->
-*- Cryptographic API  --->
    Library routines  --->
---
    Load an Alternate Configuration File
    Save an Alternate Configuration File
```

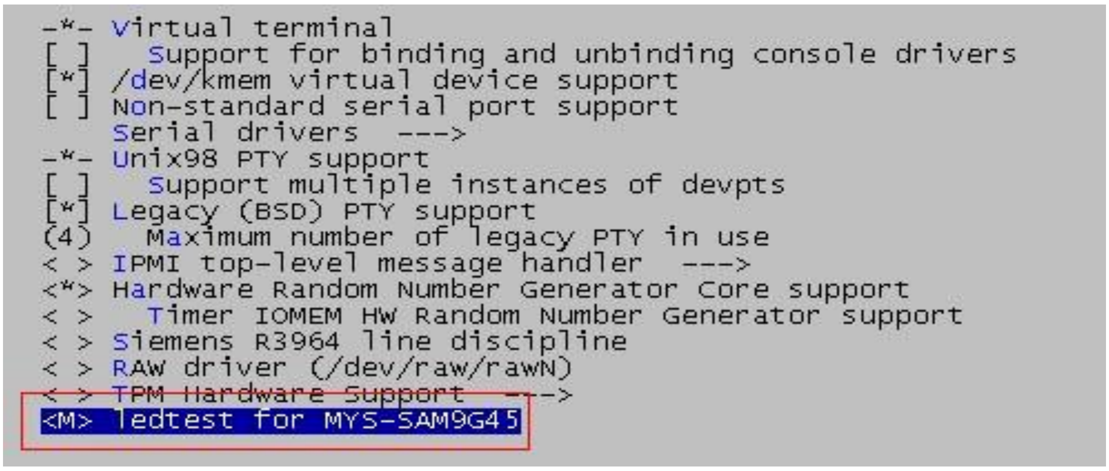
Figure 4-14



```
    Generic Driver Options  --->
< > Connector - unified userspace <-> kernelspace linker  --->
<*> Memory Technology Device (MTD) support  --->
< > Parallel port support  --->
[*] Block devices  --->
[*] Misc devices  --->
< > ATA/ATAPI/MFM/RLL support  --->
    SCSI device support  --->
< > Serial ATA (prod) and Parallel ATA (experimental) drivers  --->
[ ] Multiple devices driver support (RAID and LVM)  --->
[*] Network device support  --->
[ ] ISDN support  --->
    Input device support  --->
    Character devices  --->
<*> I2C support  --->
[*] SPI support  --->
-*- GPIO Support  --->
< > Dallas's 1-wire support  --->
< > Power supply class support  --->
< > Hardware Monitoring support  --->
< > Generic Thermal sysfs driver  --->
[ ] Watchdog Timer Support  --->
    Sonics Silicon Backplane  --->
    Multifunction device drivers  --->
    Multimedia devices  --->
    Graphics support  --->
<*> Sound card support  --->
[*] HID Devices  --->
[*] USB support  --->
<*> MMC/SD/SDIO card support  --->
< > Sony MemoryStick card support (EXPERIMENTAL)  --->
[ ] Accessibility support  --->
[*] LED Support  --->
<*> Real Time Clock  --->
[*] DMA Engine support  --->
[ ] Auxiliary Display support  --->
v(+)

              <Select>    < Exit >    < Help >
```

Figure 4-15



```
-*- Virtual terminal
[ ]   Support for binding and unbinding console drivers
[*] /dev/kmem virtual device support
[ ] Non-standard serial port support
    Serial drivers  --->
-*- Unix98 PTY support
[ ]   Support multiple instances of devpts
[*] Legacy (BSD) PTY support
(4)   Maximum number of legacy PTY in use
< > IPMI top-level message handler  --->
<*> Hardware Random Number Generator Core support
< >   Timer IOMEM HW Random Number Generator support
< > Siemens R3964 line discipline
< > RAW driver (/dev/raw/rawN)
< > TPM Hardware Support  --->
<M> ledtest for MYS-SAM9G45
```

Figure 4-16

(3) Compile driver module

Operation as follows:

```
# touch    drivers/char/ledtest.c
# make    ARCH=arm    modules    \
  CROSS_COMPILE=/usr/local/arm-2007q1/bin/arm-none-linux-gnueabi-
```

After complete the compilation, it will generate driver file ledtest.ko in drivers/char/.

4.8.4 Download driver

ledtest.ko file compiled successfully is copied to SD card or U disk, specific actions are as follows:

(1) Cancel trigger by tother drivers to two LEDs.

```
root@at91sam:/# cd /sys/class/leds/d7
root@at91sam:/sys/class/leds/d1# echo none > trigger
root@at91sam:/sys/class/leds/d1# cd ../d8
root@at91sam:/sys/class/leds/d2# echo none > trigger
```

(2) load driver module into kernel

```
root@at91sam:/# cd    /media/sda4/MYS-SAM9G45
root@at91sam:/media/sda4/MYS-SAM9G45# ls
ledtest.ko      ledtest_app
root@at91sam:/media/sda4/MYS-SAM9G45# insmod    ledtest.ko
MYS_SAM9G45_LEDTEST DRIVER MODULE INIT
register MYD_SAM9G45_ledtest Driver OK! Major = 249
MYD-SAM9G45-ledtest initialized
```

At this point, LED driver has loaded into kernel successfully. In the next section, we will write a simple application to test driver, verifying whether drive is working properly.

4.9 Application Development Instance

This section describes the upper layer of the Linux system application development, and a simple instance tells application development process and driver invocation. Achieve the function: control LED by passed parameter.

4.9.1 Source code compilation

Create a new directory and a new ledtest_app.c file by vi editor or copy compiled file to current directory directly, ledtest_app.c source is as follows:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
```

```
    #include <sys/ioctl.h>

    #define LED_DEV "/dev/MYS-SAM9G45-ledtest"

    int main(int argc, char **argv)
    {
        int fd, ret, led_num, led_status;
        if (argc!=3 || sscanf(argv[1],"%d", &led_num)!=1
            || sscanf(argv[2],"%d", &led_status)!=1)
        {
            printf("\r\nPlease input correct parameters !\r\n\n");
            printf("usage:\r\n%s <led_num> <led_status>\r\n", argv[0]);
        printf("\r\nOptions:\r\n");
            printf("   led_num\t-   1 for red led, 0 for blue led.\r\n   led_status\t-   1 for
ON, 0 for OFF.\r\n\n");
            exit(1);
        }

        if((led_status!=1 && led_status!=0) || (led_num!=0 && led_num!=1))
        {
             printf("\r\nError: The parameter value must be '0' or '1' !\r\n");
             printf("\r\nPlease try again !!! !\r\n\n");
             exit(1);
        }

        fd = open(LED_DEV, 0);
        if (fd < 0)
        {
            printf("\r\nFail to open device '%s'!\r\n\n", LED_DEV);
            exit(1);
        }

        ret = ioctl(fd, led_status, led_num);
        if(ret < 0)
        {
            printf("\r\nFail calling ioctl !\r\n\n");
        }

        close(fd);
        return 0;
    }
```

4.9.2 Compile

Because of one source files, so do not write Makefile, if more source file to be compiled, recommend writing Makdfile file.

Add path of cross-compiler tools to PATH:

```
# export PATH=$PATH:/usr/local/arm-2007q1/bin/
```

Use cross compiler tool to compile:

```
# arm-none-linux-gnueabi-gcc   -o   ledtest   ledtest.c
```

Perform the above operation, if no error is generated in current directory, ledtest_app will be generated.

4.9.3 Application Use

After compilation, copy ledtest copied to board by SD card or U disk, and then run tfile in terminal. Control two LED by passed two parameters. The first parameter controls which LED lights ("0" is blue led, "1" is red led), and the second parameter controls LED lights ("0" is ON, "1" is OFF). The specific operation is as follows:

When input parameters is not enough:

```
root@at91sam:/$ ./ledtest   1

Please input correct parameters !

usage:
./ledtest_app_mys <led_num> <led_status>

Options:
   led_num          -   1 for red led, 0 for blue led.
   led_status      -   1 for ON, 0 for OFF.
```

When input parameter is out of range:

```
root@at91sam:/$ ./ledtest   2   1

Error: The parameter value must be '0' or '1' !

Please try again !!! !
```

Input parameters Correct:

```
root@at91sam:~# /media/sda4/ledtest   1   1
MYS-SAM9G45-ledtest Driver Open Called!
MYS_SAM9G45_LED Driver Release Called!
```

The application is executed in right way. Passing the above parameters, blue LED is off.

Chapter 5 Android System Guidelines

5.1 Overview

Android is a Linux system based open source operating system, mainly used in portable devices. Android operating system originally developed by Andy Rubin development, initially mainly support mobile phone. In 2005 Android is purchased by Google, formatting the open mobile phone alliance to improvement it, gradually extended to the tablet computer and other area. Since its first release Welcomed by the majority of consumers, Android’s market shares around the world more than Symbian system for the first time in the first quarter of 2011, ranking first in the world. The data show that in February 2012, Android accounted for 52.5% of the share of the global smartphone operating system market.

Android system is running based on Linux system, mainly made by Linux Kernel, system libraries, Dalvik virtual machine, application framework, and applications written mainly by JAVA. Its framework is as shown in figure 5-1:

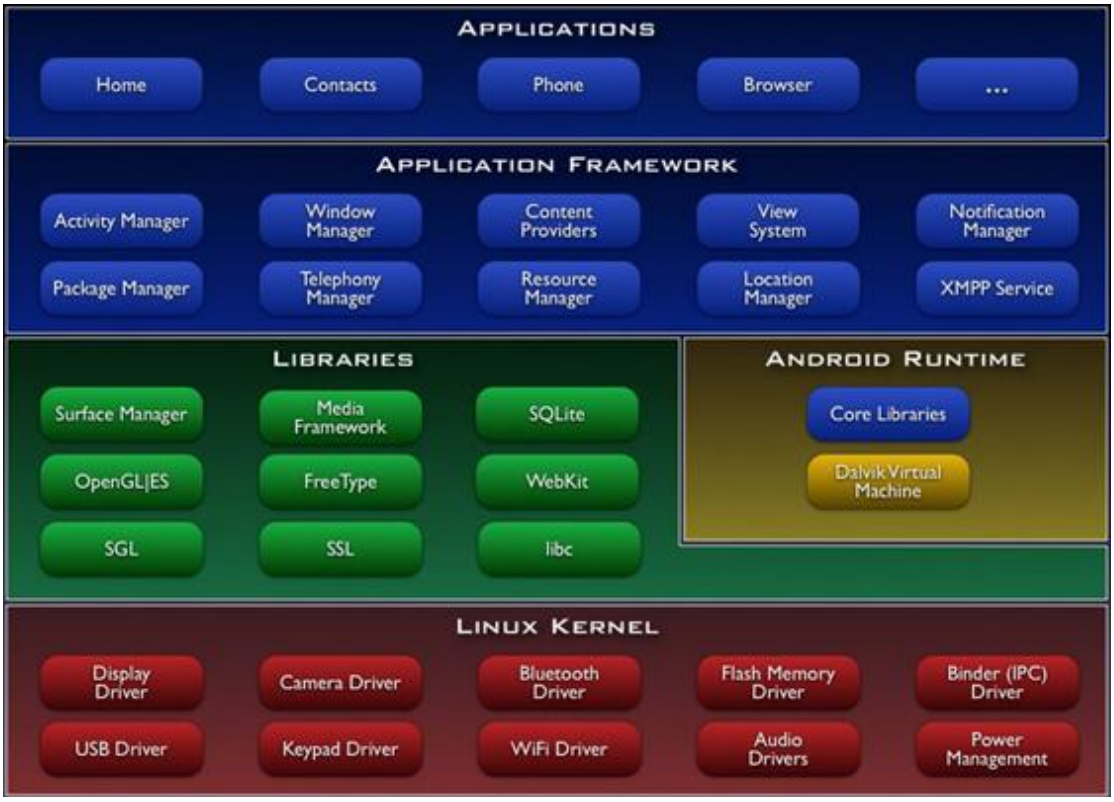


Figure 5-1

This chapter describes how to build and run Android 2.3.1 Gingerbread system in MYS-SAM9G45 platform, include the following:

(1) Quickly build Android system

(2) Quickly develop Android file system

(3) Product Android file system

(4) The use of Android System

5.2 Build Android System

This section describes how to use image to build Android system.

5.2.1 Install Download Tool

Refer to 03-Tools\SAM-BA\sam-ba install.

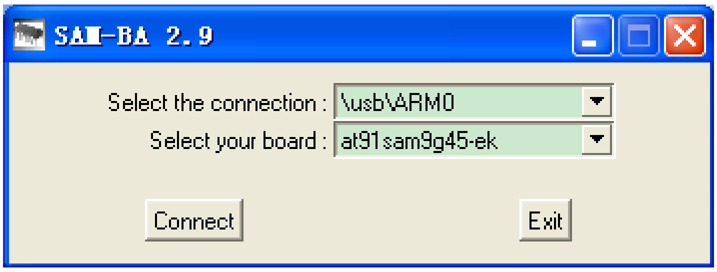
5.2.2 Connect Board and SAM-BA

(1) Install MYS-SAM9G45 USB driver

Refer to 03-Tools\SAM-BA\ The Board Driver Installation Guide.pdf.

(2) Connect board.

First disconnect JP1 and JP2, and then double-click sam-ba v2.9 on PC desktop. Interface is as shown in figure 5-2:



Select the connection : \usb\ARM0

Select your board : at91sam9g45-ek

Figure 5-2

Then connect JP2, click Connect to connect board.

5.2.3 Automatic Download

Complete chapter 5.2.1 and 5.2.2, and open 02-Images\Android_Image\ MYS-SAM9G45_Android_4.3. Double click download.bat, and wait for about 3 minutes, Android image will be downloaded automatically to board by SAM-BA. After the download is completed, press K1 button to reset board to start Android system.

5.2.4 Manual Download

All image files used in this section can be found in the directory: **\02-Images\Android_Image\MYS-SAM9G45_Android_4.3\**

The NandFlash content of Android system is divided as shown in figure 5-3:

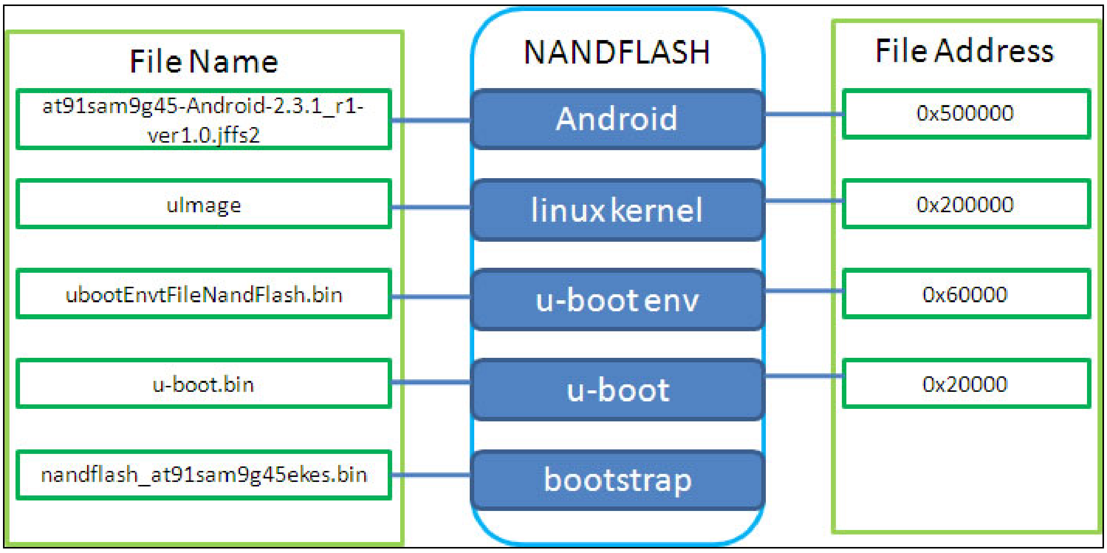
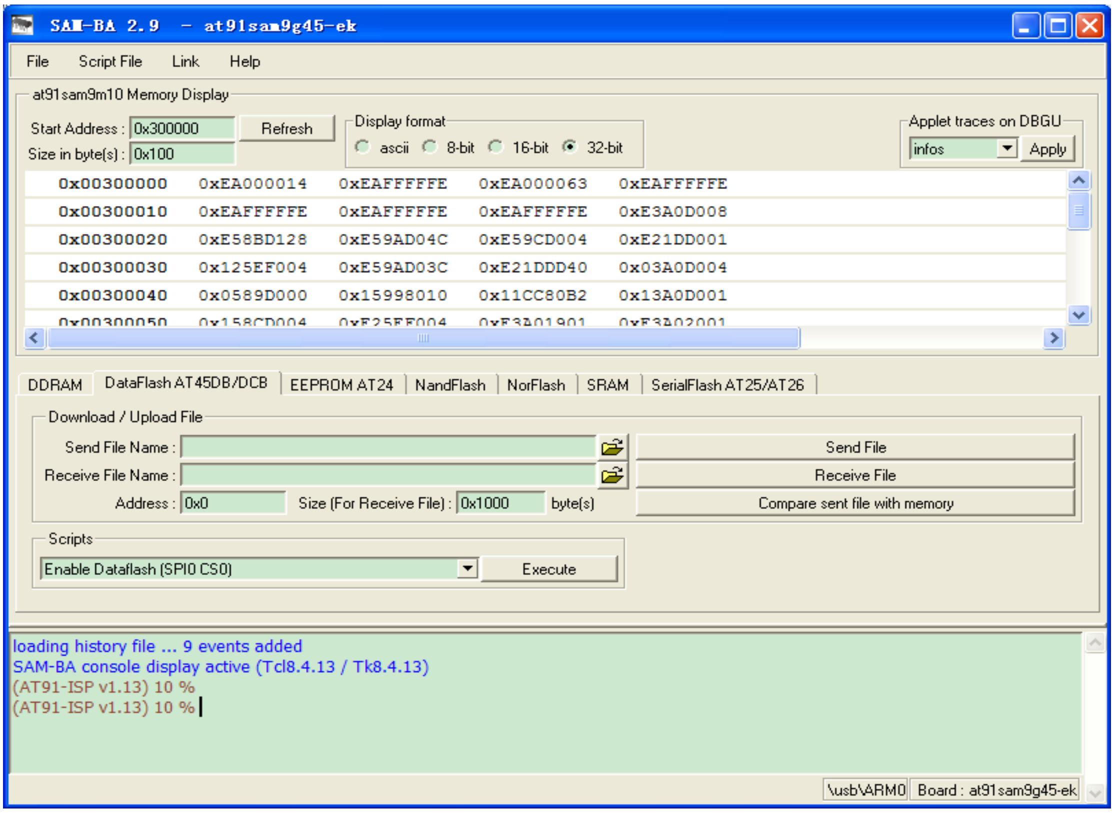


Figure 5-3

Manual download Linux by SAM-BA:

(1) Complete chapter 5.2.1, 5.2.2, and double-click sam-ba v2.11. Refer to figure 5-4:



at91sam9m10 Memory Display

Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

Send File Name :

Receive File Name :

Address : 0x0   Size (For Receive File) : 0x1000 byte(s)

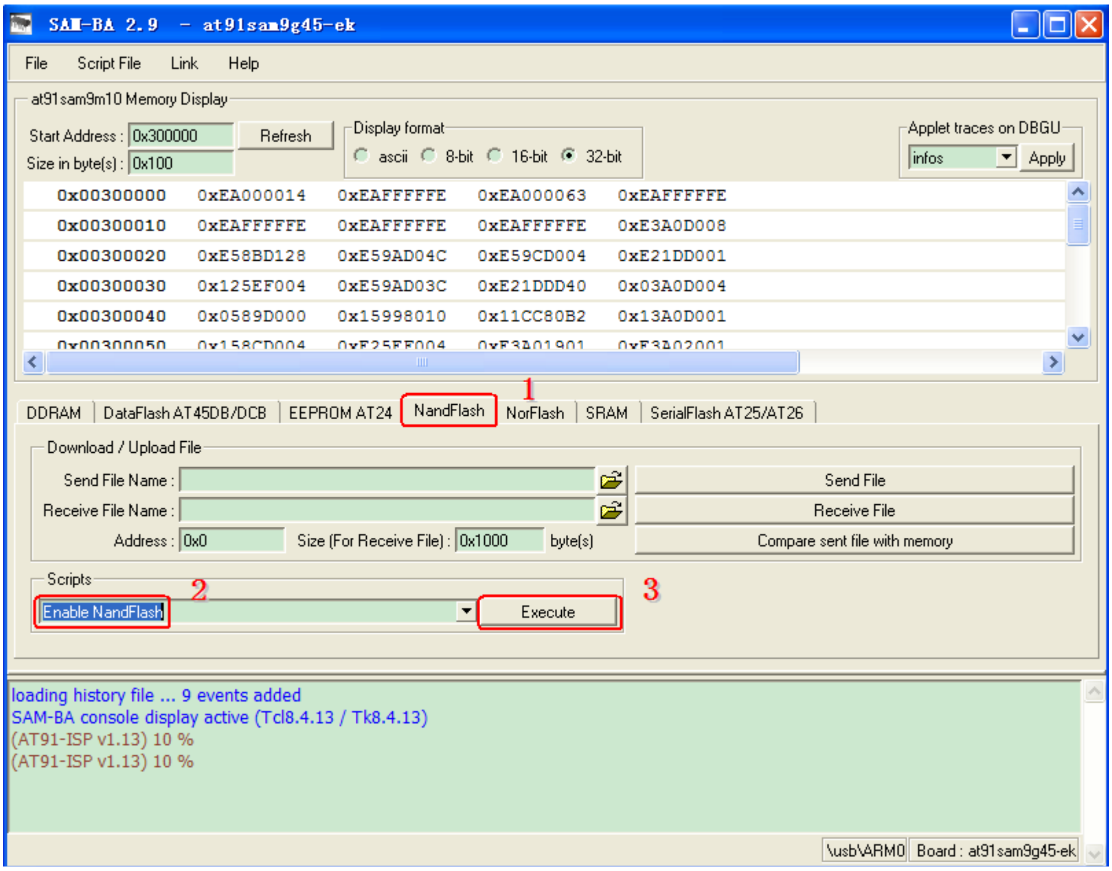
Scripts: Enable Dataflash (SPI0 CS0)

```
loading history file ... 9 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 10 %
(AT91-ISP v1.13) 10 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 5-4

(2) Select NandFlash tab. Select “Enable NandFlash” in Scripts tab and then click Execute. Refer to figure 5-5:



Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE3A01901   0xE3A02001
```

Send File Name :

Receive File Name :

Address : 0x0   Size (For Receive File) : 0x1000 byte(s)

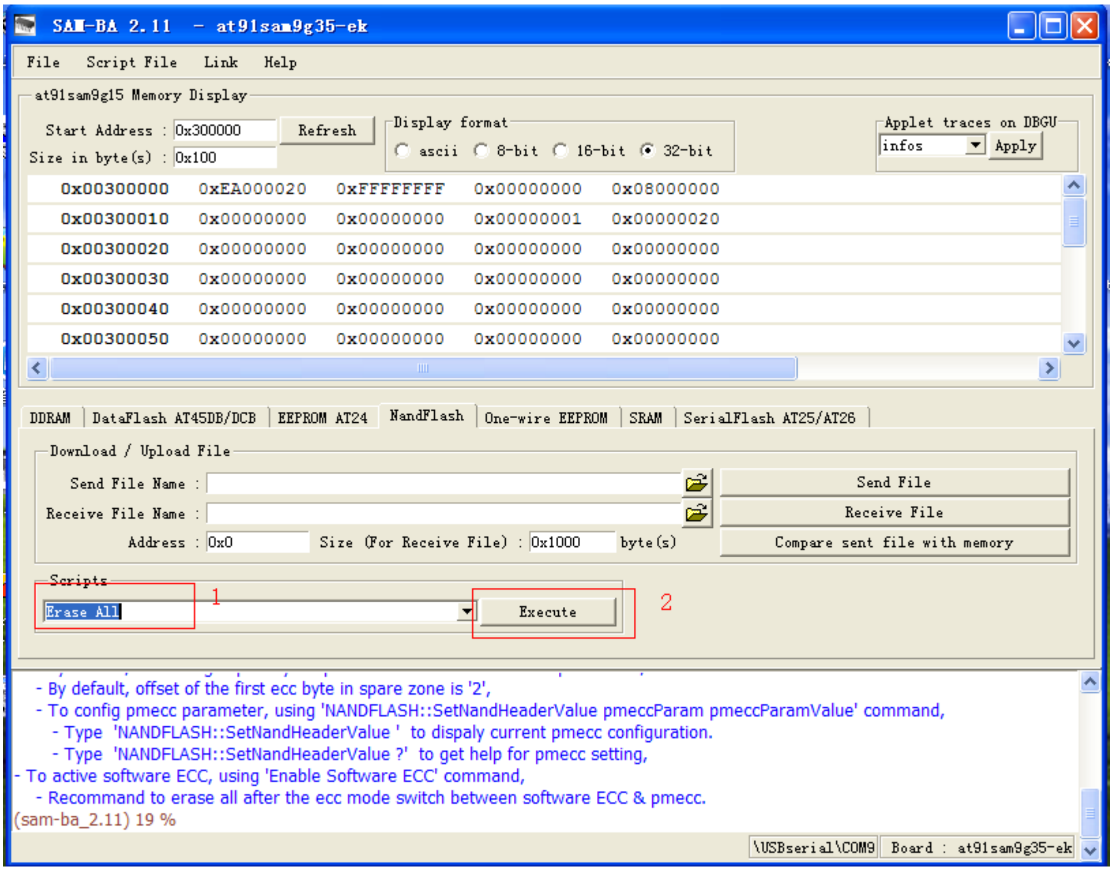
Scripts: Enable NandFlash

```
loading history file ... 9 events added
SAM-BA console display active (Tcl8.4.13 / Tk8.4.13)
(AT91-ISP v1.13) 10 %
(AT91-ISP v1.13) 10 %
```

\usb\ARM0   Board : at91sam9g45-ek

Figure 5-5

(3) Select Erase All in Scripts tab and then click Execute. Refer to figure 5-6:



Start Address : 0x300000
Size in byte(s) : 0x100
Display format ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU infos Apply

```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x00000000   0x00000000   0x00000001   0x00000020
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
0x00300040   0x00000000   0x00000000   0x00000000   0x00000000
0x00300050   0x00000000   0x00000000   0x00000000   0x00000000
```

Send File Name :
Receive File Name :
Address : 0x0 Size (For Receive File) : 0x1000 byte(s)

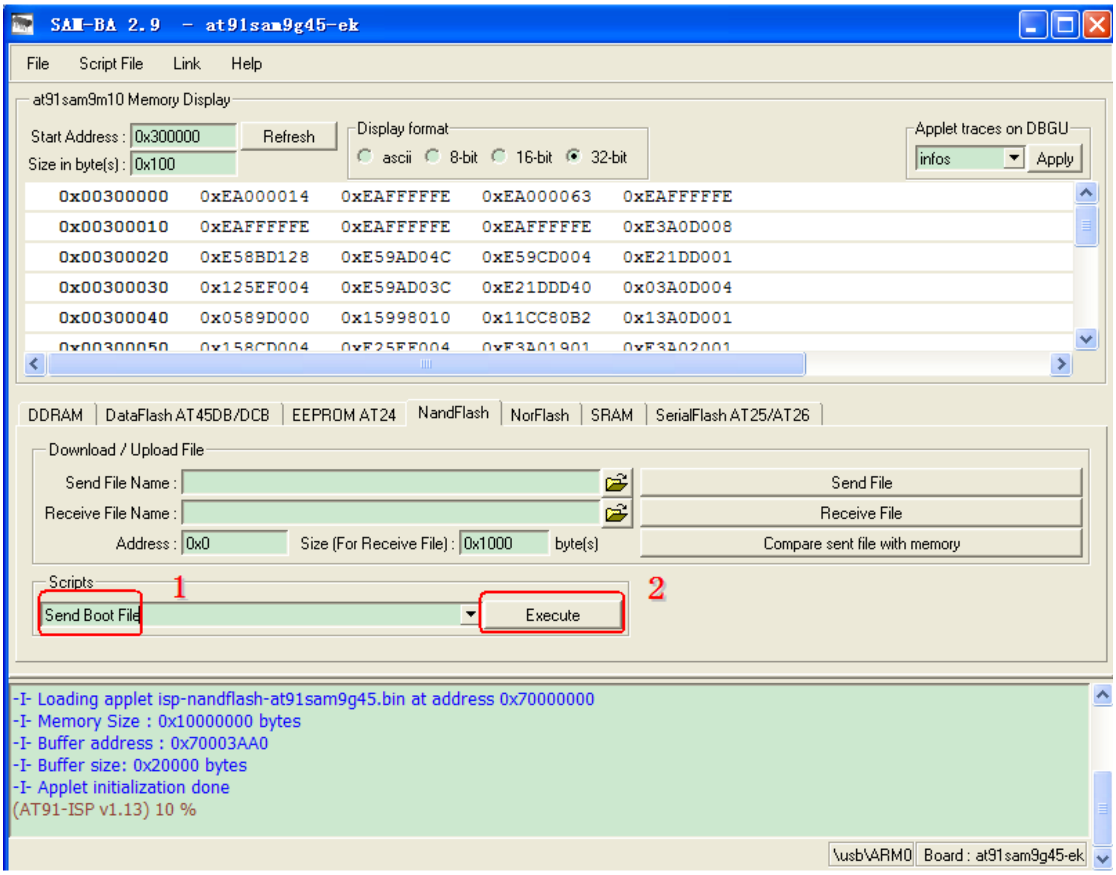
Scripts: Erase All

```
  - By default, offset of the first ecc byte in spare zone is '2',
  - To config pmecc parameter, using 'NANDFLASH::SetNandHeaderValue pmeccParam pmeccParamValue' command,
    - Type  'NANDFLASH::SetNandHeaderValue '  to dispaly current pmecc configuration.
    - Type  'NANDFLASH::SetNandHeaderValue ?'  to get help for pmecc setting,
- To active software ECC, using 'Enable Software ECC' command,
  - Recommand to erase all after the ecc mode switch between software ECC & pmecc.
(sam-ba_2.11) 19 %
```

\USBserial\COM9 Board : at91sam9g35-ek

Figure 5-6

(5) Program nandflash_at91sam9g45ekes.bin. Refer to figure 5-7:



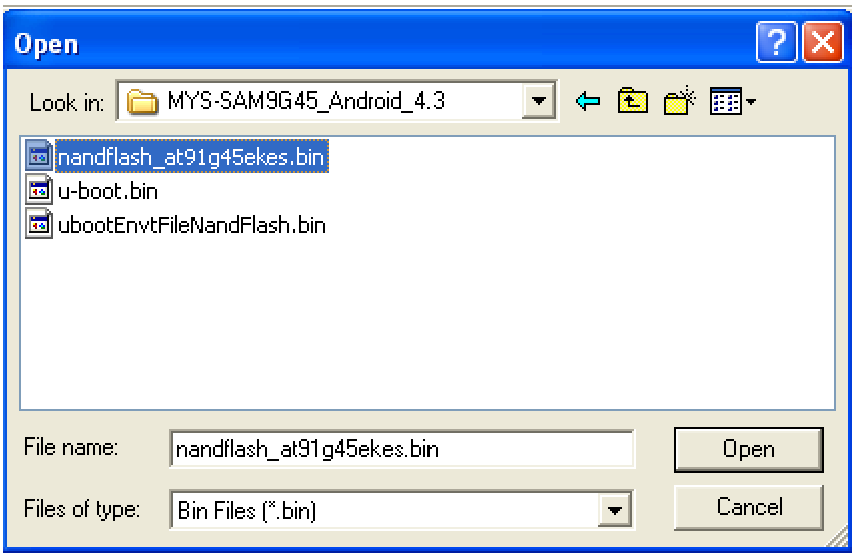
Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25FF004   0xE3A01901   0xE3A02001
```

Send File Name :
Receive File Name :
Address : 0x0
Size (For Receive File) : 0x1000 byte(s)
Scripts: Send Boot File

```
-I- Loading applet isp-nandflash-at91sam9g45.bin at address 0x70000000
-I- Memory Size : 0x10000000 bytes
-I- Buffer address : 0x70003AA0
-I- Buffer size: 0x20000 bytes
-I- Applet initialization done
(AT91-ISP v1.13) 10 %
```

\usb\ARM0 Board : at91sam9g45-ek



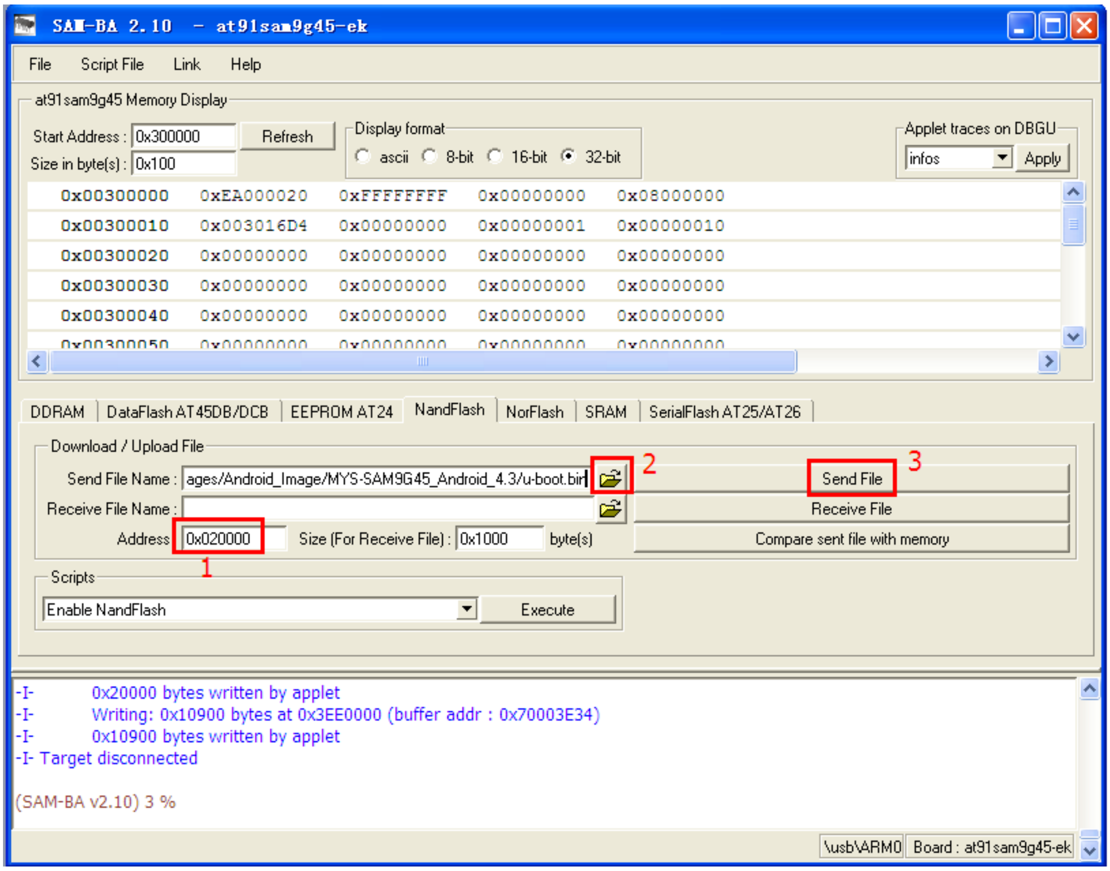
Look in: MYS-SAM9G45_Android_4.3

- nandflash_at91g45ekes.bin
- u-boot.bin
- ubootEnvtFileNandFlash.bin

File name: nandflash_at91g45ekes.bin
Files of type: Bin Files (*.bin)

Figure 5-7

(5) Download u-boot.bin file to 0x20000. Refer to figure 5-8:



at91sam9g45 Memory Display

Start Address : 0x300000
Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x003016D4   0x00000000   0x00000001   0x00000010
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
0x00300040   0x00000000   0x00000000   0x00000000   0x00000000
0x00300050   0x00000000   0x00000000   0x00000000   0x00000000
```

Download / Upload File

Send File Name : ages/Android_Image/MYS-SAM9G45_Android_4.3/u-boot.bin
Receive File Name :
Address : 0x020000
Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable NandFlash

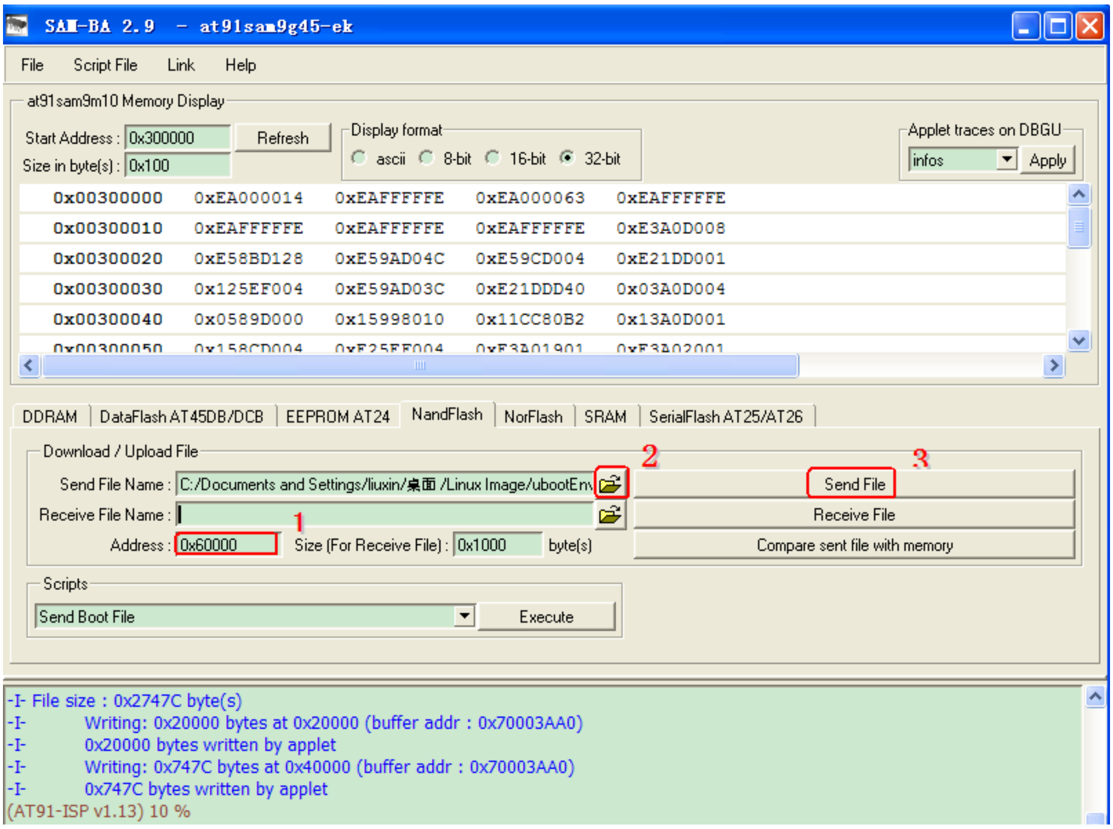
```
-I-        0x20000 bytes written by applet
-I-        Writing: 0x10900 bytes at 0x3EE0000 (buffer addr : 0x70003E34)
-I-        0x10900 bytes written by applet
-I- Target disconnected

(SAM-BA v2.10) 3 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 5-8

(6) Download u-boot.bin file to 0x60000. Refer to figure 5-9:



at91sam9m10 Memory Display

Start Address : 0x300000
Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

Download / Upload File

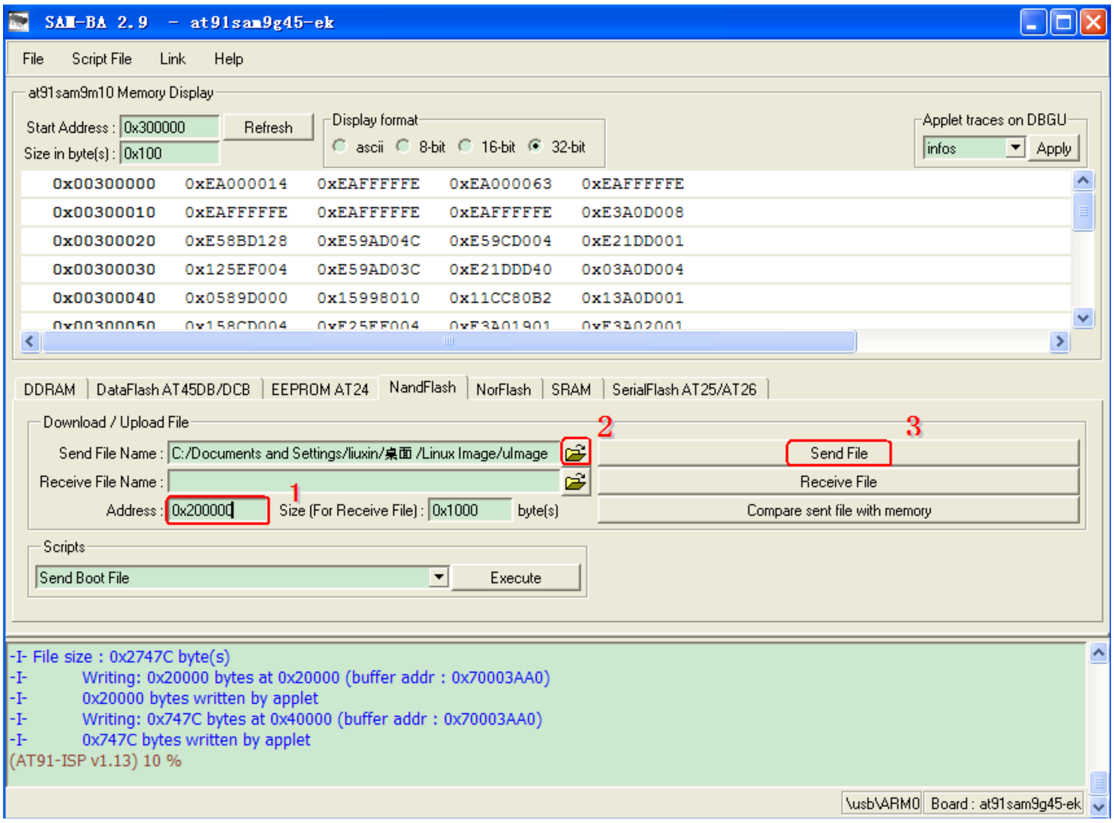
Send File Name : C:/Documents and Settings/liuxin/桌面 /Linux Image/ubootEn
Receive File Name :
Address : 0x60000
Size (For Receive File) : 0x1000 byte(s)

Scripts: Send Boot File

```
-I- File size : 0x2747C byte(s)
-I-        Writing: 0x20000 bytes at 0x20000 (buffer addr : 0x70003AA0)
-I-        0x20000 bytes written by applet
-I-        Writing: 0x747C bytes at 0x40000 (buffer addr : 0x70003AA0)
-I-        0x747C bytes written by applet
(AT91-ISP v1.13) 10 %
```

Figure 5-9

(7) Download Linux kernel uImage to 0x200000. Refer to figure 5-10:



Start Address : 0x300000
Size in byte(s) : 0x100
Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit
Applet traces on DBGU: infos

```
0x00300000   0xEA000014   0xEAFFFFFE   0xEA000063   0xEAFFFFFE
0x00300010   0xEAFFFFFE   0xEAFFFFFE   0xEAFFFFFE   0xE3A0D008
0x00300020   0xE58BD128   0xE59AD04C   0xE59CD004   0xE21DD001
0x00300030   0x125EF004   0xE59AD03C   0xE21DDD40   0x03A0D004
0x00300040   0x0589D000   0x15998010   0x11CC80B2   0x13A0D001
0x00300050   0x158CD004   0xE25EF004   0xE3A01901   0xE3A02001
```

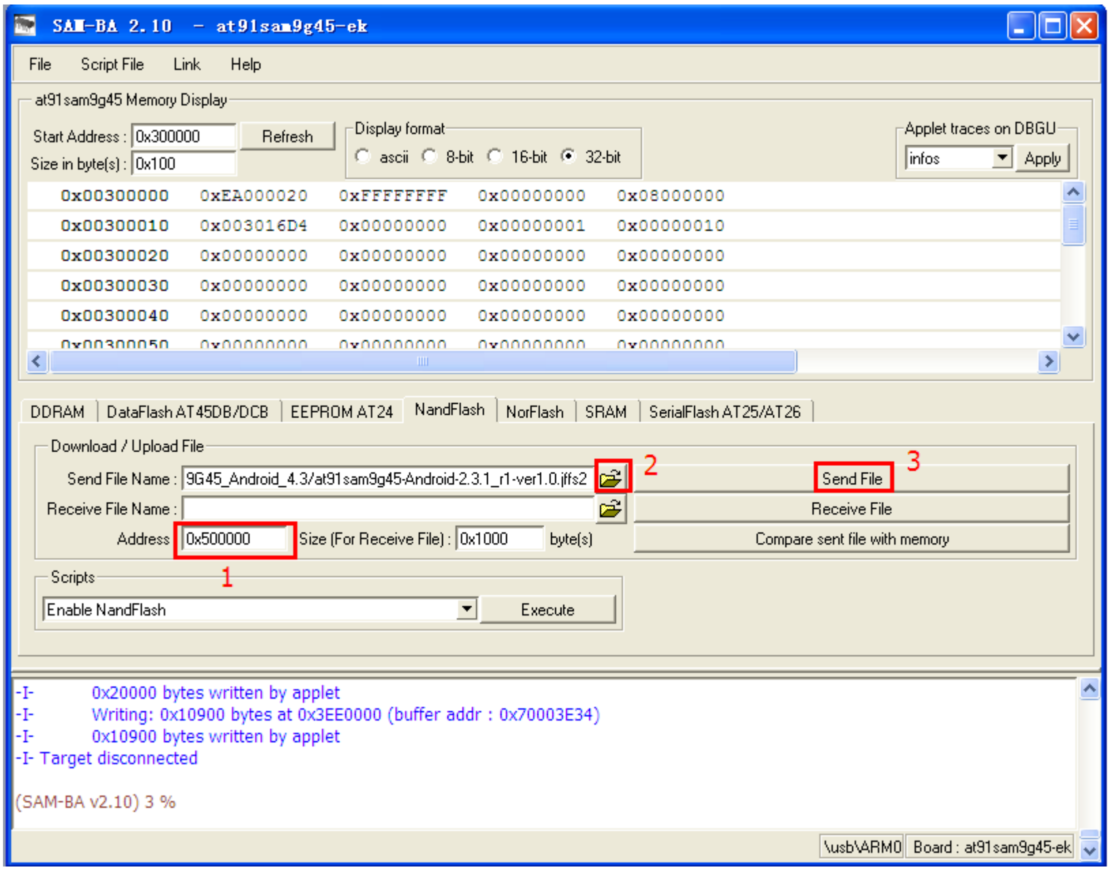
Send File Name : C:/Documents and Settings/liuxin/桌面 /Linux Image/uImage
Receive File Name :
Address : 0x200000
Size (For Receive File) : 0x1000 byte(s)
Scripts: Send Boot File

```
-I- File size : 0x2747C byte(s)
-I-        Writing: 0x20000 bytes at 0x20000 (buffer addr : 0x70003AA0)
-I-        0x20000 bytes written by applet
-I-        Writing: 0x747C bytes at 0x40000 (buffer addr : 0x70003AA0)
-I-        0x747C bytes written by applet
(AT91-ISP v1.13) 10 %
```

Board : at91sam9g45-ek

Figure 5-10

(8) Download at91sam9g45-Android-2.3.1_r1-ver1.0.jffs2 to 0x500000. Refer to figure 5-11:



Start Address : 0x300000

Size in byte(s) : 0x100

Display format: ○ ascii ○ 8-bit ○ 16-bit ● 32-bit

Applet traces on DBGU: infos

```
0x00300000   0xEA000020   0xFFFFFFFF   0x00000000   0x08000000
0x00300010   0x003016D4   0x00000000   0x00000001   0x00000010
0x00300020   0x00000000   0x00000000   0x00000000   0x00000000
0x00300030   0x00000000   0x00000000   0x00000000   0x00000000
0x00300040   0x00000000   0x00000000   0x00000000   0x00000000
0x00300050   0x00000000   0x00000000   0x00000000   0x00000000
```

Send File Name : 9G45_Android_4.3/at91sam9g45-Android-2.3.1_r1-ver1.0.jffs2

Receive File Name :

Address 0x500000

Size (For Receive File) : 0x1000 byte(s)

Scripts: Enable NandFlash

```
-I-        0x20000 bytes written by applet
-I-        Writing: 0x10900 bytes at 0x3EE0000 (buffer addr : 0x70003E34)
-I-        0x10900 bytes written by applet
-I- Target disconnected

(SAM-BA v2.10) 3 %
```

\usb\ARM0 Board : at91sam9g45-ek

Figure 5-11

At this point, Android system image file download is completed, disconnect USB and pressing K1 can restart Android system.

5.3 Develop Android file system

Full compile an Android file system takes a long time (the next section will expand description). For simple applications, such as add, modify files, modify the startup logo, add binding APK application system, we can bypass a long and relatively complex processs. Specific practice is to mount Android file system provided by the CD-ROM to the Linux system, and then modify Android file system, finally regenerate Android file system by mkfs.jffs2 tools. Here take Jewels_1.1.apk for an example which adds binding APK application.

5.3.1 Mount Android file system

Please refer to **“4.6.2 Mount jffs2 file system on a PC in Linux platform”.**

It needs to be noted file system loaded in step (2) is Android file system: at91sam9g45-Android-2.3.1_r1-ver1.0.jffs2

5.3.2 Modify Android file system

After mount Android file system, and then modify original file system contents. Here will add gem game Jewels_1.1.apk as application system, copy Jewels_1.1.apk to /system/app/ directory in Android file system directly:

```
# cd fsmount/
# sudo cp ../Jewels_1.1.apk ./system/app/
# sync
```

Note that here the modifications are carried out in simulation MTD partition, doesn't apply to at91sam9g45-Android-2.3.1_r1-ver1.0.jffs2 file in chapter 5.3.1. So after complete modification, it needs to regenerate a file system.

5.3.3 Regenerate Android file system

After complete modification, it needs to use mkfs.jffs2 tool to regenerate a file system. If there is no mkfs.jffs2, through the following command to get:

```
# sudo apt-get install mtd-utils
```

Enter following command to generate new jffs2 file system:

```
# sudo mkfs.jffs2 -n -l -s 0x800 -e 0x20000 -p 0x4000000 -d fsmount/ -o android-roo\
tfs.jffs2
# sync
```

mkfs.jffs2 Parameter description:
-n It is not writting CLEANMARKER at the beginning of each block, NANDFLASH itself have CLEANMAKER.
-l Little-endian format (little-ending)
-s Specify the page size, 0x800 (2KB)
-e Specify the erase block size, the block size of NANDFLASH in MYS - the SAM9G45 is 0x20000 (128KB)
-p Prefilled size is assigned to 0x4000000 (64MB) for the file system partition
-d Specify the enter directory of the generated file system
-o Specify the output file

After complete it, download generated android-rootfs.jffs2 file to board. The specific download, please refer to **5.2 quickly build the Android system.**

5.4 Make Android file system

This section describes methods and steps of Android file system.

5.4.1 Compilation and installation

We know that Android system is running on Linux-based system, so if build Android system, it must first set up a Linux- based platform.

(1) Install cross compiler tools, please refer to " 4.4.3 install the cross compiler tools " section.

(2) Compile AT91Bootstrap, please refer to“ **4.4.4 install AT91Bootstrap source and compile**”.

(3) Compile uboot, please refer to " **4.4.5 install uboot source code and compile**".

(4) compile Linux kernel

Unzip Linux kernel.

```
# tar xvjf \
06-Android_Source/linux_kernel_2.6.30_android \
/mys-sam9g45-linux-2.6.30-android.tar.bz2
# cd mys-sam9g45-linux-2.6.30-android
```

Enter the following command to compile linux kernel:

```
# make ARCH=arm menuconfig
# make uImage \
   ARCH=arm \
   CROSS_COMPILE=/usr/local/arm-2007q1/bin/arm-none-linux-gnueabi-
```

> Note: making uImage command requires your Ubuntu system has been installed uImage tool, otherwise, use the following command to install the tool:
> \# **apt-get install uboot-mkimage**

After compile kernel, uImage is Linux kernel file in arch/ ar /boot/directory.

5.4.2 Build compiler environment

In default, for Android 2.2.x version, it needs a 64-bit compiler environment. If compile Android 2.2.x or later in 32-bit compiler environments, it needs to make some changes described by Later chapters.

In addition, if compile Android system on a virtual machine, it needs at least 1G virtual memory, 30G hard disk space, otherwise the compilation cannot be completed. Compilation environment:

- Ubuntu 10.04 32-bit System
- Memory 1.5G
- Hard disk space 40G

(1) Install JDK 1.6

Click the following link to download JDK1.6 self-extracting installation package:

https://edelivery.oracle.com/otn-pub/java/jdk/6u32-b05/jdk-6u32-linux-i586.bin

Enter the following command to download.

```
# wget https://edelivery.oracle.com/otn-pub/java/jdk/6u32-b05/jdk-6u32-linux-i586.bin
```

Then run self-extracting packages to install directly:

```
# sudo chmod a+x jdk-6u32-linux-i586.bin
# ./jdk-6u32-linux-i586.bin
```

(2) Install necessary toolkit

Enter the following command to install

```
# sudo apt-get install git-core gnupg flex bison gperf build-essential \
   zip curl zlib1g-dev libc6-dev lib32ncurses5-dev ia32-libs \
   x11proto-core-dev libx11-dev lib32readline5-dev lib32z-dev \
   libgl1-mesa-dev g++-multilib mingw32 tofrodos python-markdown \
   libxml2-utils xsltproc
```

More details,refer to: http://source.android.com/source/initializing.html

5.4.3 Download Android Official Source

(1) Install Repo tool

Establish bin folder in user directory and add the path to the PATH

```
# mkdir ~/bin
# PATH=~/bin:$PATH
```

Download repo script, and give the executable permissions:

```
# curl https://dl-ssl.google.com/dl/googlesource/git-repo/repo > ~/bin/repo
# chmod a+x ~/bin/repo
```

(2) Initialize Repo Repo

Establish a working directory, here take myandroid for example:

```
# mkdir myandroid
# cd myandroid
```

Run repo init to specify Android source address, in order to ensure download success, if there is no Google account, it needs to register a Google account in following website:

https://accounts.google.com/SignUp

Open following pages to complete registration and using just registered account to login:

https://android.googlesource.com/new-password

Click on page "allow access", similar information is as follows:

```
    1 machine android.googlesource.com login git-<userName>.gmail.com password
<password>
    2 machine android-review.googlesource.com login git-<userName>.gmail.com
password <password>
```

<userName> and <password> is newly registered user name and password. The above information is appended to ~/.Netrc at the end of file (check the permissions of current user, if this file does not exist, create a new one).

Configure repo:

```
# repo init -u \
  https://android.googlesource.com/a/platform/manifest -b android-2.3.1_r1
```

(3) Download Android source code

Enter following command repo will download Android source code automatically from server to current directory:

```
# repo sync
```

This process will take longer, a few hours to a day range. If disconnected for some reason, and rerun repo sync.

Detailed process, refer to: http://source.android.com/source/downloading.html

5.4.4 Install patch code

In chapter 5.3.3, we create a working directory myandroid. Copy

06-Android_Source/Myir_Code/Android_Patch/atmel.tar.bz2 to /myandroid/device directory. And execute the following commands:

```
    # cd Android-2.3.1_r1/device
    # tar xvjf atmel.tar.bz2
```

Then copy 06-Android_Source/Myir_Code/Generate_jffs2_image/generate_jffs2_image.s to /myandroid directory.

5.4.5 Compile Android system

If use 64-bit compile platform, skip this chapter.

For Android 2.2.x versions need a 64 bits compiler environment, compile directly if in 32 bits platform will appear the following error:

```
build/core/main.mk:73: You are attempting to build on a 32-bit system.
build/core/main.mk:74: Only 64-bit build environments are supported beyond froyo/2.2.
```

The solution is as follows:

(1) Enter Android working directory, find four documents:

./external/clearsilver/cgi/Android.mk
./external/clearsilver/java-jni/Android.mk
./external/clearsilver/util/Android.mk
./external/clearsilver/cs/Android.mk

```
    LOCAL_CFLAGS += -m64
    LOCAL_LDFLAGS += -m64
```

Amended as follows:

```
    LOCAL_CFLAGS += -m32

    LOCAL_LDFLAGS += -m32
```

(2) open file . /Build/core/main.mk, and find:

```
    ifneq (64,$(findstring 64,$(build_arch)))
```

Amended as follows:

```
    ifneq (i686,$(findstring i686,$(build_arch)))
```

After completing the above modifications, compile android system normally in 32-bit compiler platform.

5.4.6 Configure and compile Android file system

Enter the following commands to configure and compile In turn.

```
# cd myandroid
# make clean
# source build/envsetup.sh
# partner_setup sam9g45
# choosecombo Device release sam9g45 eng
# make
```

5.4.7 Generat Android file system

Enter following command in turn to produce Android file system at91sam9g45-Android-2.3.1_r1-ver1.0.jffs2:

```
# cd myandroid
# cd Generate_jffs2_image
# ./generate_jffs2_image.sh -b sam9g45 -l 4.3
```

5.5 Download Android image

This section, please refer to chapter 5.2.

5.6 Android System Use

5.6.1 Mount SD card

(1) Begin to enter system interface. Refer to figure 5-11:

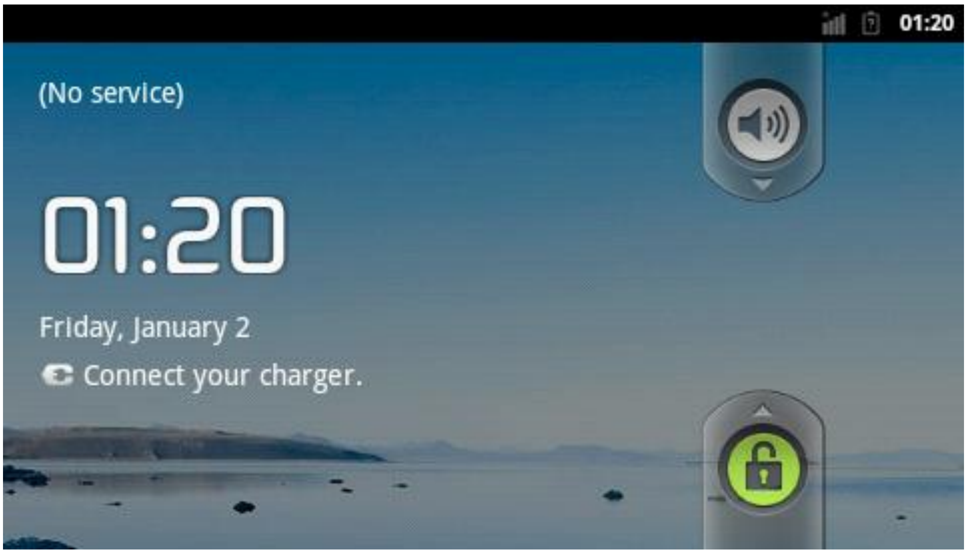


Figure 5-11

(2) When SD card is inserted into a slot inside, it will display a message in the status bar. Refer to figure 5-12:

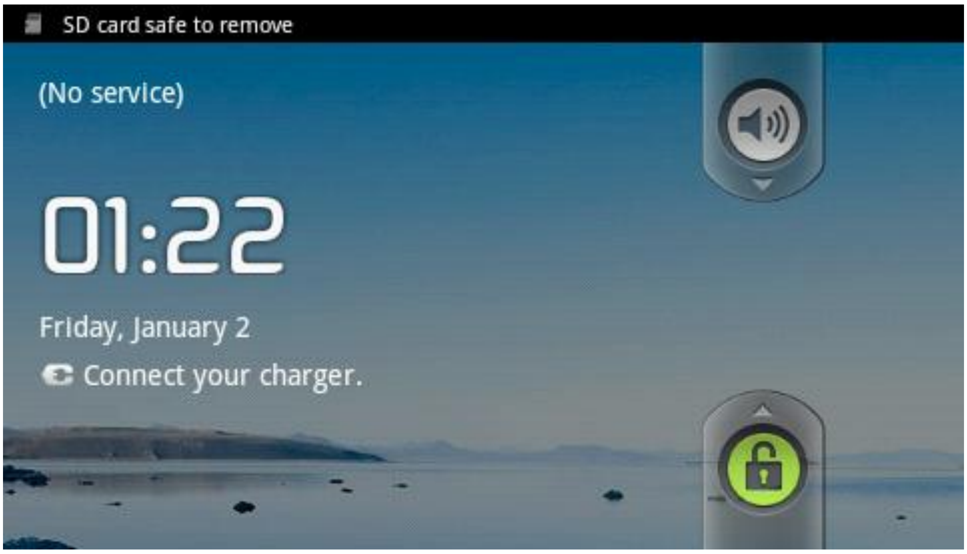
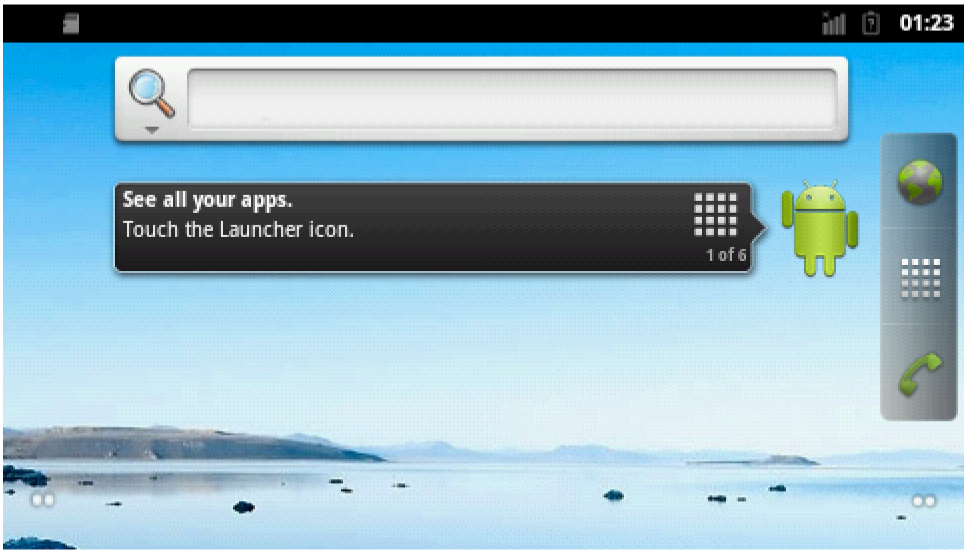


Figure 5-12

(3) Press User1 key or F1 key in usb keyboard can unlock system. After unlocking system, interface is shown in figure 5-13:



See all your apps.
Touch the Launcher icon.

Figure 5-13

(4) Then press ▦ flag into Select program interface. Refer to figure 5-14:

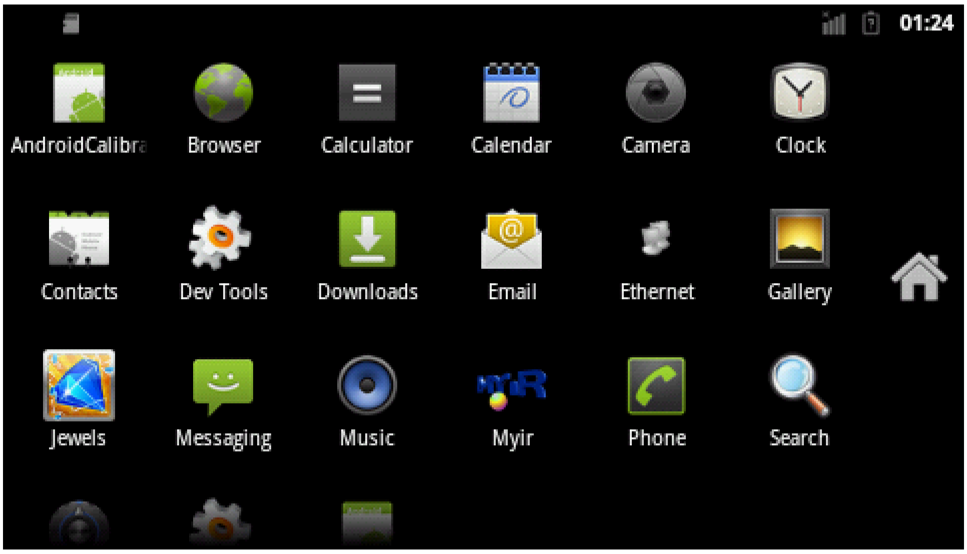


Figure 5-14

(5) Select "Settings" option. Refer to figure 5-15:

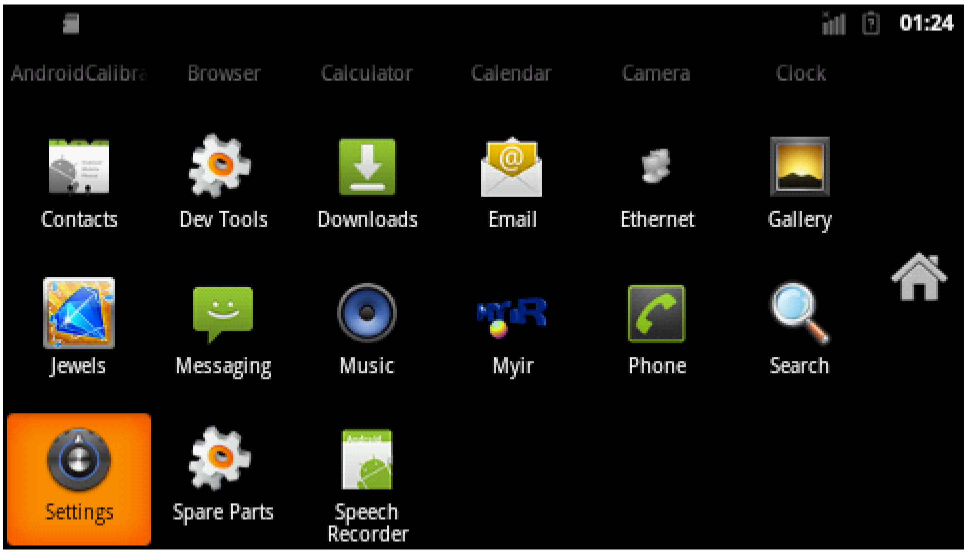


Figure 5-15

(6) After entering it, select "Storage" option. Refer to figure 5-16:

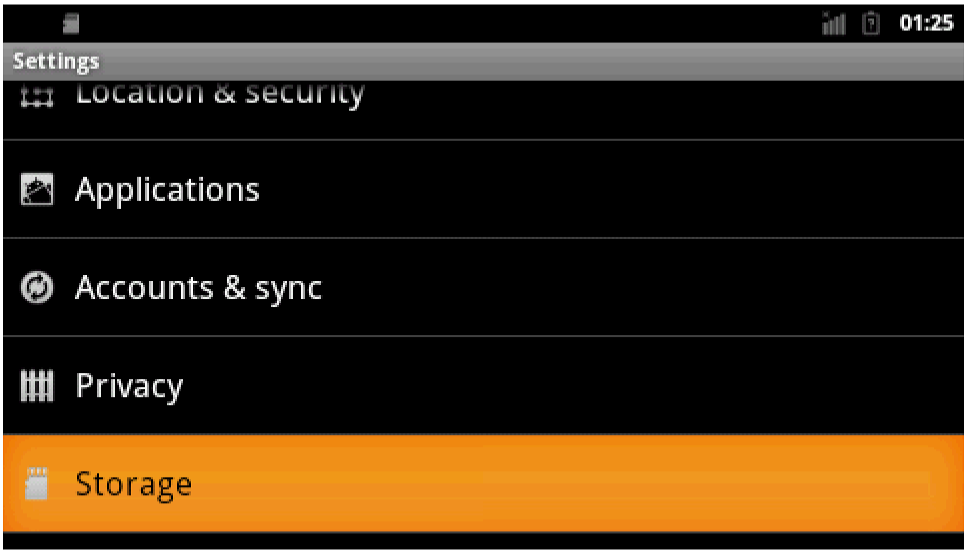
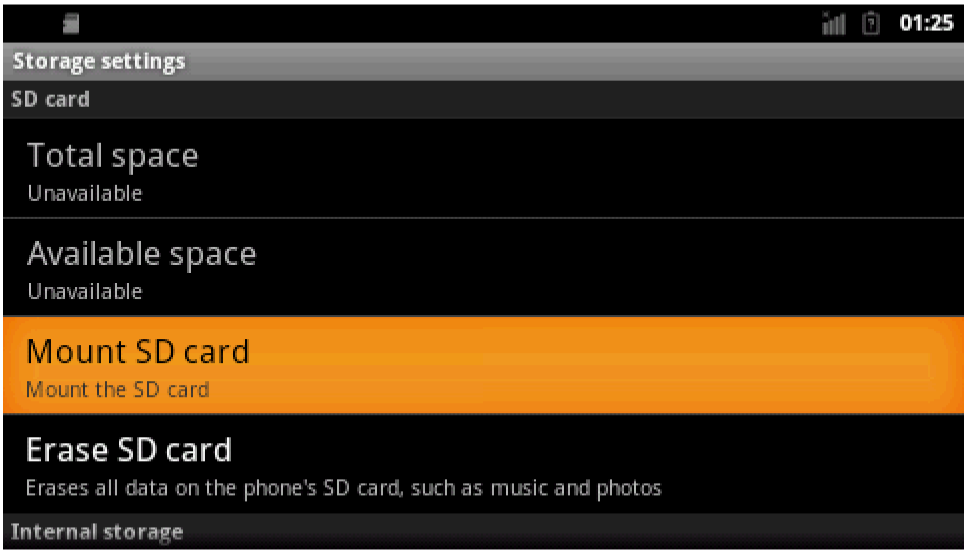


Figure 5-16

(7) Select "Mount SD card". Refer to figure 5-17:



01:25

Storage settings

SD card

Total space
Unavailable

Available space
Unavailable

Mount SD card
Mount the SD card

Erase SD card
Erases all data on the phone's SD card, such as music and photos

Internal storage

Figure 5-17

(8) After mount it successfully, SD card can be used.

5.6.2 USB keyboard use

Insert USB keyboard to J14, press Num Lock key, when light turns green, it shows usb keyboard can be used.

5.6.3 Audio test

(1) Find and click marked Music figure. Refer to figure 5-18:

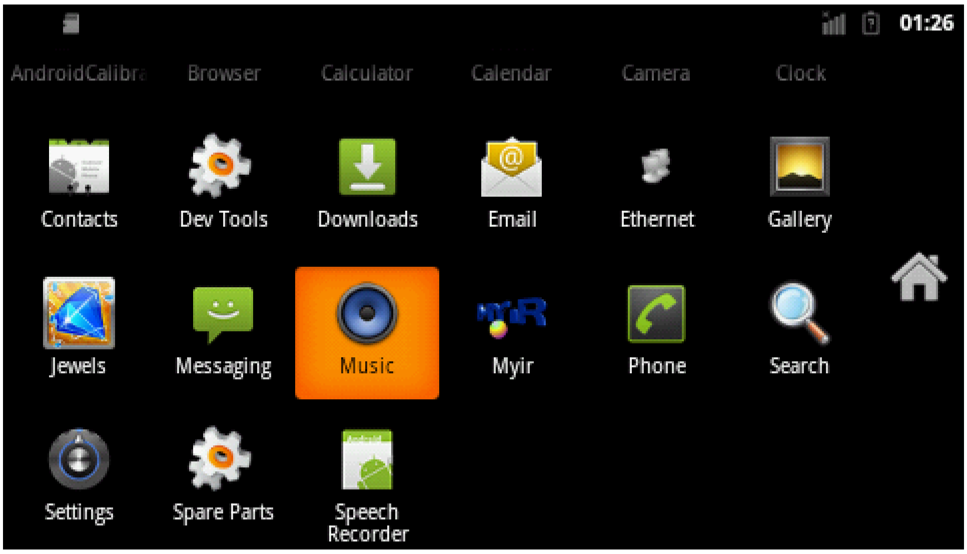


Figure 5-18

(2) Then there will be song attributes, select “Songs”. Refer to figure 5-19:

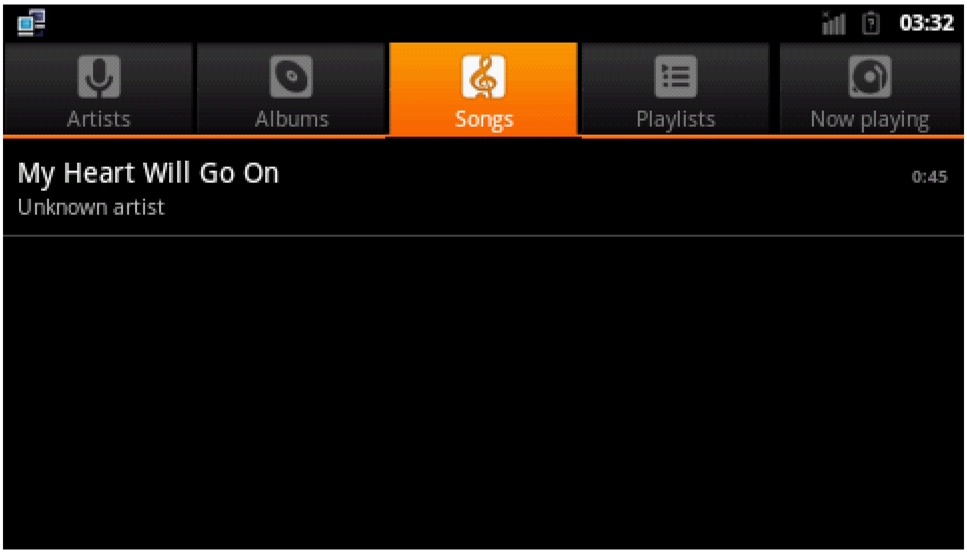


Figure 5-19

(3) Select songs and play. Refer to figure 5-20:

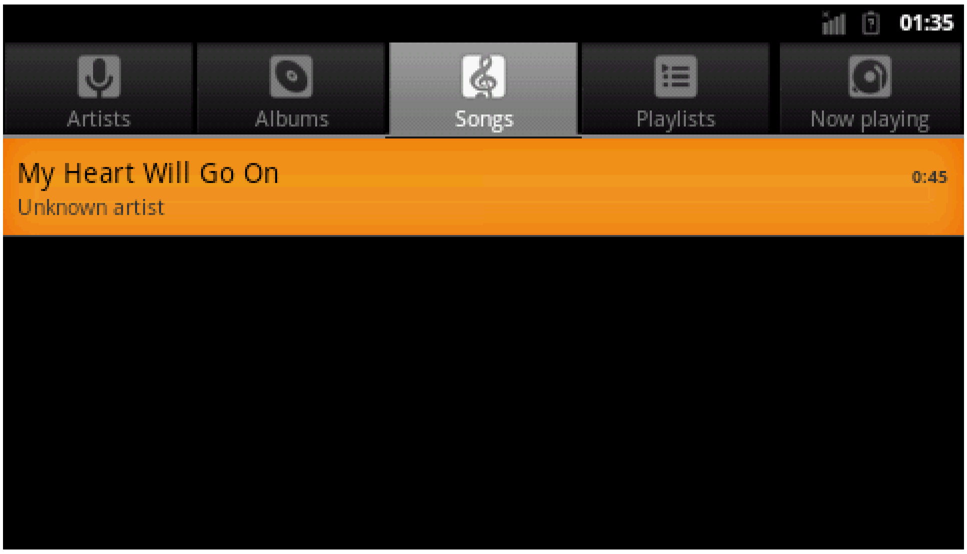


Figure 5-20

(4) The player interface is shown in figure 5-21:

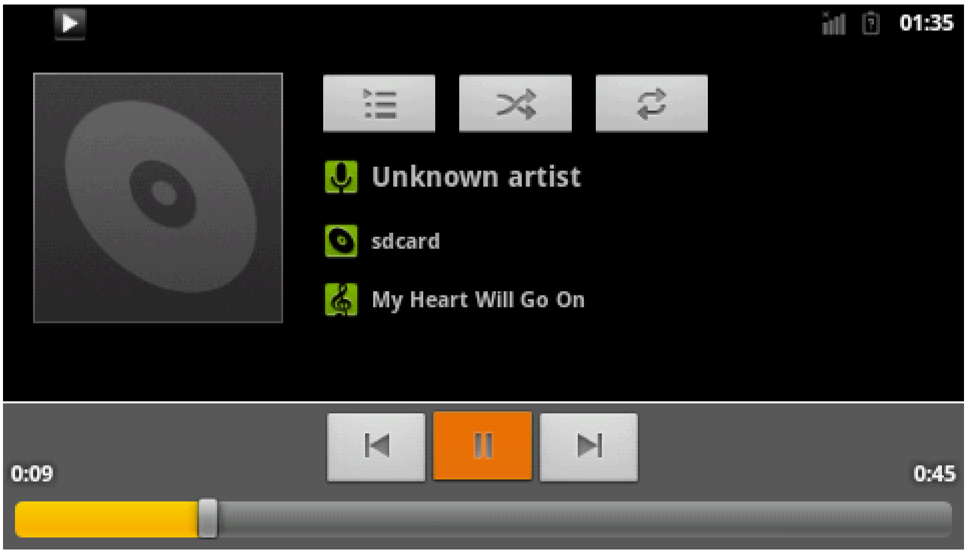


Figure 5-21

5.6.4 Ethernet test

Note: first ly to connect board to router by cable

(1) Locate and click Ethernet figure. Refer to figure 5-22:

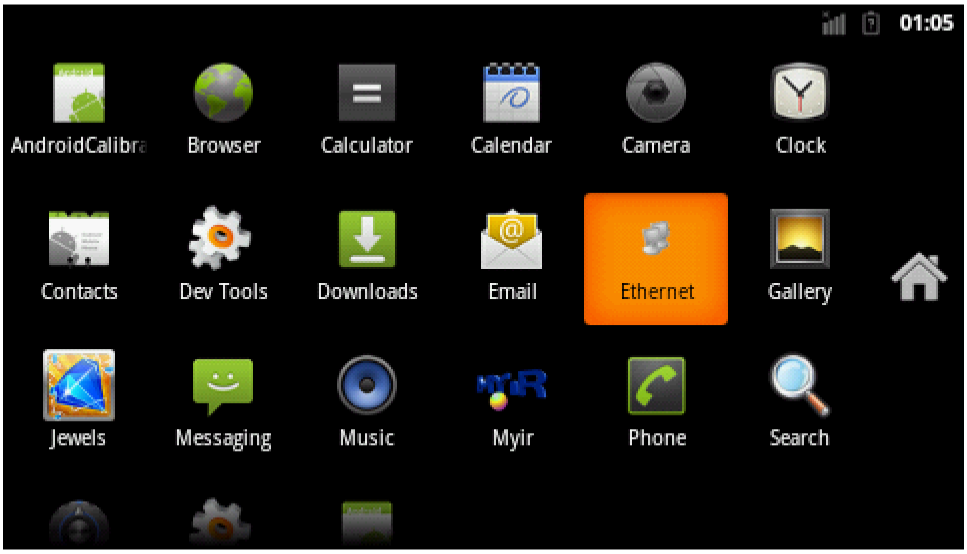
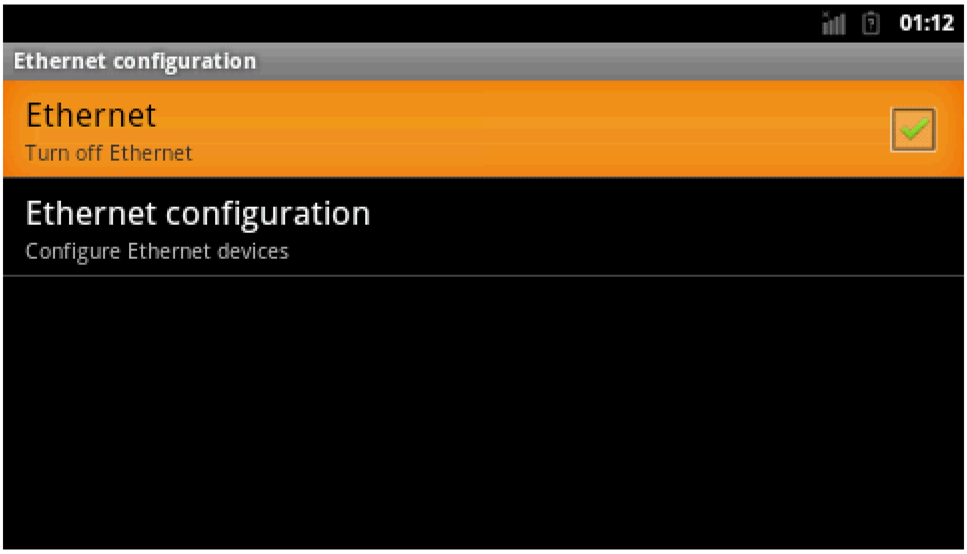


Figure 5-22

(2) Open Ethernet. Refer to figure 5-23:



01:12

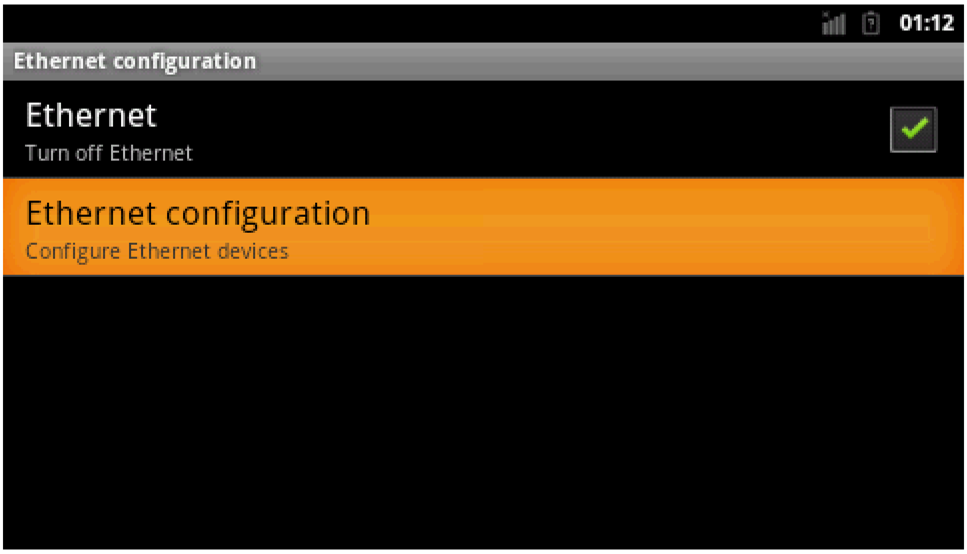
Ethernet configuration

Ethernet ☑
Turn off Ethernet

Ethernet configuration
Configure Ethernet devices

Figure 5-23

(3) Select "Ethernet configuration" to enter. Refer to figure 5-24:



01:12

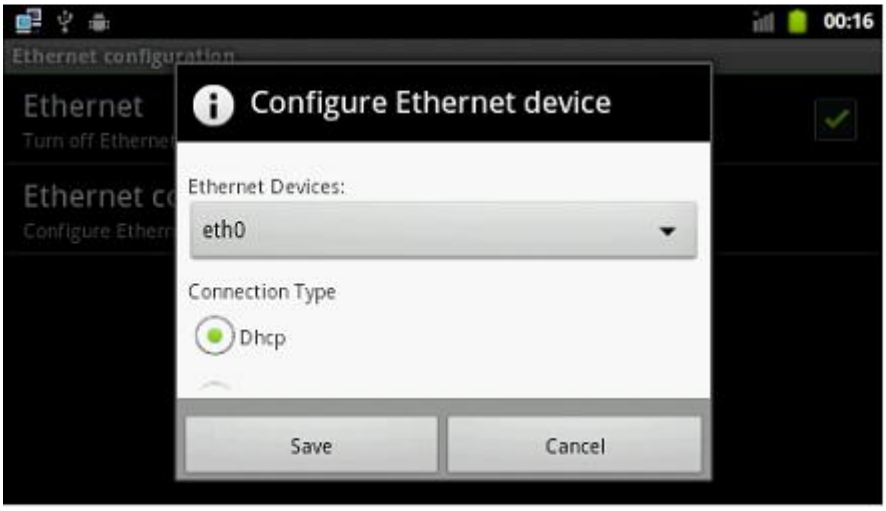
Ethernet configuration

Ethernet ☑
Turn off Ethernet

Ethernet configuration
Configure Ethernet devices

Figure 5-24

(4) Choose Dhcp to obtain dynamic IP. Otherwise, it should set IP address, subnet mask, DNS server, and default gateway manually. Set interface is as shown in figure 5-25:



Ethernet configuration

Ethernet
Turn off Etherne

Ethernet c
Configure Ethern

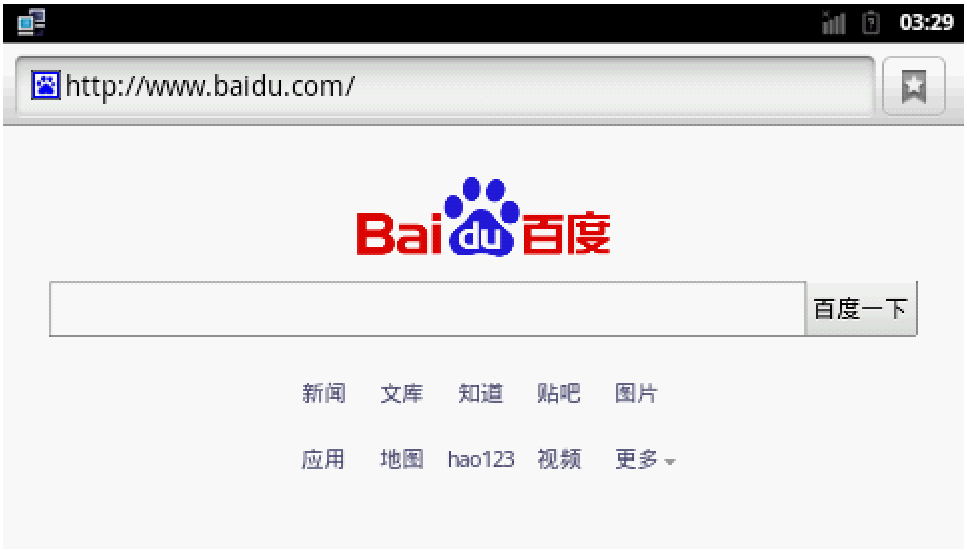
Configure Ethernet device

Ethernet Devices:
eth0

Connection Type
● Dhcp

Figure 5-25

(5) After configuration is successful, input: www.baidu.com, Refer to figure 5-26:



http://www.baidu.com/

Figure 5-26

5.6.5 WIFi Test

Note: firstly insert wifi (only support rt2070 and rt3070) into usb Host interface.

(1) Find and click "Settings". Refer to figure 5-27:

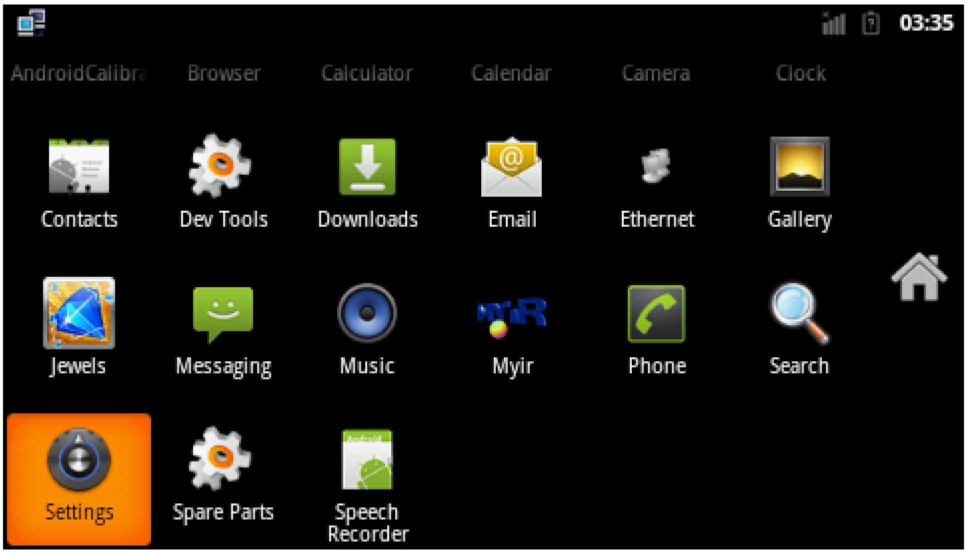


Figure 5-27

(2) Click "Wireless & networks" Refer to figure 5-28:

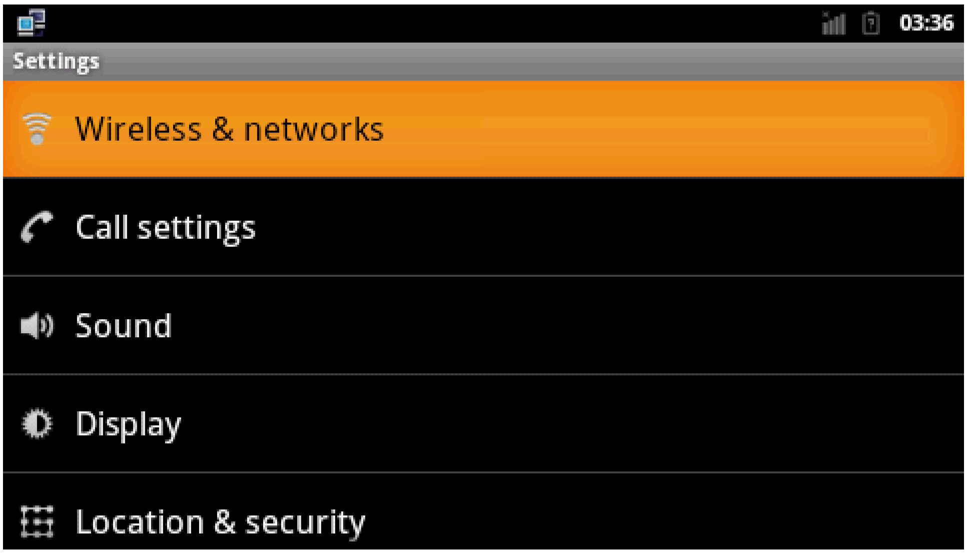
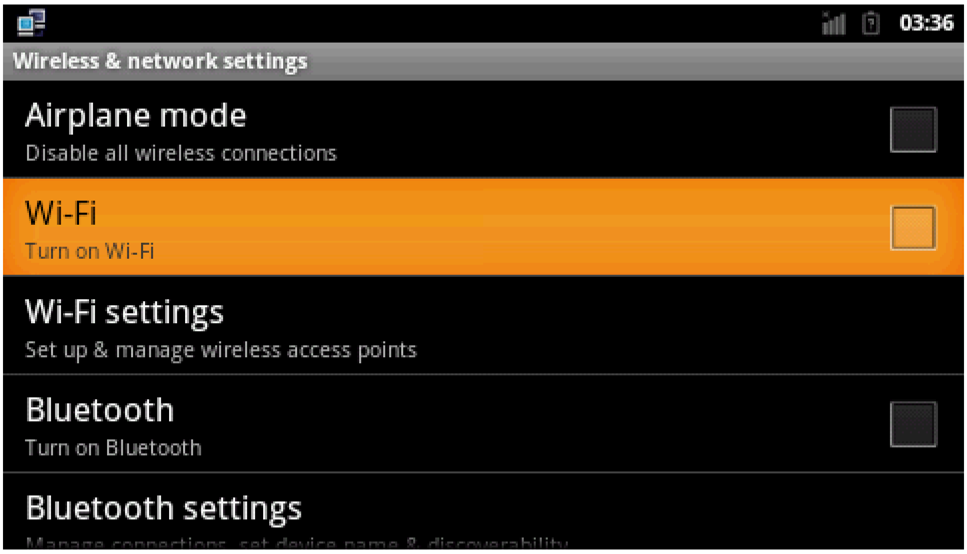


Figure 5-28

(3) Click "Wi-Fi" option to open Wi-Fi. Refer to figure 5-29:



03:36

Wireless & network settings

Airplane mode
Disable all wireless connections □

Wi-Fi
Turn on Wi-Fi □

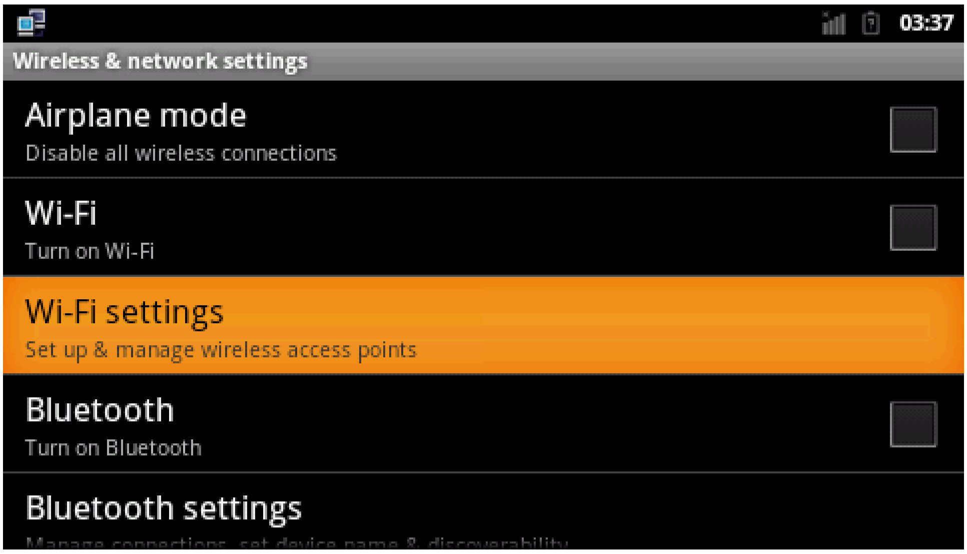
Wi-Fi settings
Set up & manage wireless access points

Bluetooth
Turn on Bluetooth □

Bluetooth settings
Manage connections, set device name & discoverability

Figure 5-29

(4) Click "Wi-Fi settings" option to configure wireless network. Refer to figure 5-30:



03:37

Wireless & network settings

Airplane mode
Disable all wireless connections □

Wi-Fi
Turn on Wi-Fi □

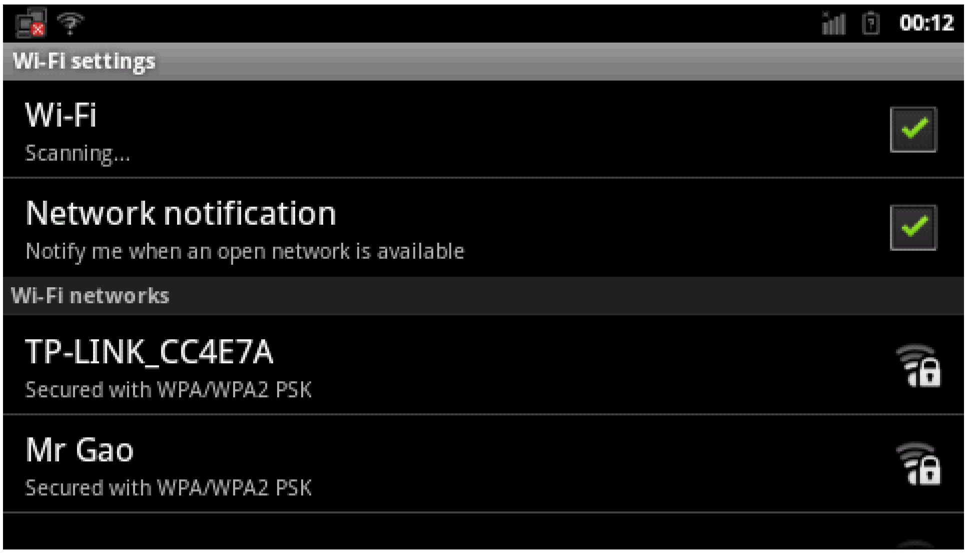
Wi-Fi settings
Set up & manage wireless access points

Bluetooth
Turn on Bluetooth □

Bluetooth settings
Manage connections, set device name & discoverability

Figure 5-30

(5) Select appropriate network and choose TP-LINK_CC4E7A. Refer to figure 5-31:



00:12

Wi-Fi settings

Wi-Fi ☑
Scanning...

Network notification ☑
Notify me when an open network is available

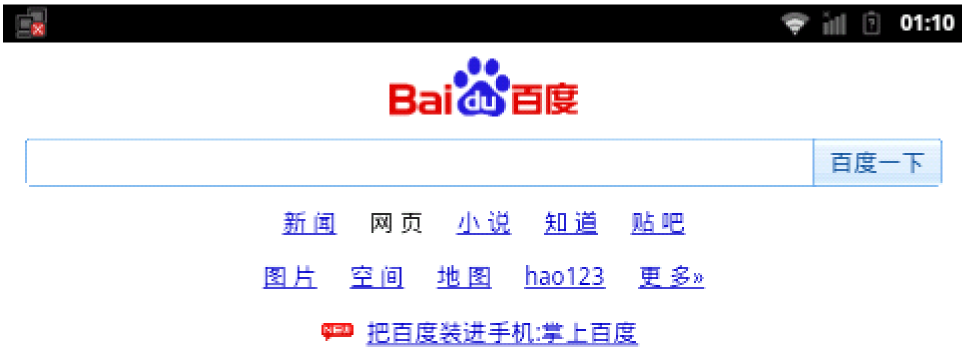
Wi-Fi networks

TP-LINK_CC4E7A
Secured with WPA/WPA2 PSK

Mr Gao
Secured with WPA/WPA2 PSK

Figure 5-31

(6) After network configuration, input: www.baidu.com. Refer to figure 5-32:



01:10

Bai du 百度

百度一下

新闻 网页 小说 知道 贴吧

图片 空间 地图 hao123 更多»

NEW 把百度装进手机:掌上百度

Figure 5-32

Wifi test successfully.

Appendix 1 sales FAQ and technical support

How to buy

We accept paypal payment and bank wire transfer

1.Paypal payment

Please select the products add into shopping cart, the checkout web page will redirect to paypal.com for you payment. Shipment fee will calculated automatically by your location region.

2.Bank wire transfer

Pls email or fax us with products list you want, we will send you a pro-invoice with order value total, shipping cost and bank information.

Shipping details

Pls select the shipping area catalogue for you location. If you have carrier account to pay the shipment fee, please select "Freight collect" and email us the carrier account.

Please visit http://www.myirtech.com/support.asp for more details

Noted

1.The shipment will start in 3 biz days by Fedex Express, it usually take 7 days to reach regular cities or regions.

2.We will use DHL Express for West asia or middle east countries, it usually take 7 days to reach regular cities or regions.

3.The remote regions defined by Fedex/DHL may cause delay, 14 days in generally.

4.Some countries have strict import policy, we will help to make shipping invoice with you requirement, like invoice value, trade term, custom statements and H.S code etc. Please contact us with these shipment requirements if your country has strict custom affairs.

Support and maintains

MYIR provides 12 months warranty for hardware products if the defects or failures were not caused by wrong use.

Return steps for defective products

1. Please email or call us get a Return Merchandise Authorization (RMA) by providing purchase details and reasons for return (defective, incorrect etc).
2. MYIR will make a shipping invoice (list value total, item description etc) for you return request. China have strict limit on return products, so please use MYIR's shipping invoice to return items to avoid custom delay.

Contact:

Tel:+86-0755-25622735 Fax: +86 755 2553 2724

Mail to: sales@myirtech.com support@myirtech.com

Website: www.myirtech.com