

CATALOG

1) Basic platform	.2.
A) Hardware platform	
B) Software platform	
2) How to build up the SMP8634 wince platform builder	.8.
3) How to run the CE system on SMP8634 board	.9.
A) Make the CE bootloader	
B) Make the SMP8634 specific CE Nk.bin	
C) Download the NK.BIN to SMP8634 board	
D) Make the SHIP mode SMP8634 CE Nk.bin and boot it up without platform builder	
4) How to make application for SMP8634 wince system	.46.
5) How to test the SMP8634 wince feature	.48.
A) SMP8634 ETH	
B) SMP8634 USB	
C) SMP8634 ATAPI	
D) SMP8634 Pflash Storage(disabled)	
E) SMP8634 IR remote control	
F) SMP8634 I2C	
G) SMP8634 Smart Card reader	
H) SMP8634 COM serial	
I) SMP8634 DDI	
J) SMP8634 HW cursor	
K) SMP8634 WAVE device	
L) SMP8634 DDR RAM	
M) SMP8634 RTC(RealTime Clock)	
N) SMP8634 GPIO	
O) SMP8634 PWM	
P) SMP8634 Hardware Demux	
Q) SMP8634 Hardware Jpeg decode	
R) SMP8634 LVDS output	
S) SMP8634 Multi-Decoder	
T) SMP8634 VGA output	
U) SMP8634 Audio and Video Captures filter	
6) How to upgrade/modify the CE boot loader	.73.
A) Update boot.bin	
B) Update XENV	
C) Set the default value for boot loader	
7) How to modify wince feature on SMP8634	.75.
A) Web server	
B) FTP server	
C) Telnet server	
D) SMB server	
E) Hive-based Registry	
F) IE 6.0 for Windows CE	
G) Media Player	
H) Microsoft Network Media Device	
I) Standard Shell	
J) DirectDVD	
k) Wire less Ethernet	
8) How to upgrade XOS of SMP8634 chip	.80.
9) How to get XOS information	.80.
10) How to make the boot Screen	.80.
11) How to show a bitmap logo in boot screen	.81.
12) How to show a progress in boot screen	.81.
13) How to dynamic change video output	.81.
14) Current Driver escape codes	.83.
15) Extended Project	.87.
16) About Windows CE Test kits(CETK)	.89.
17) BUG list	.89.

SIGMA DESIGNS®

1) Basic platform

A) Hardware platform

1. SMP8634 envision board(ES6 late version) or production board

Pflash: recommend 64M bytes(NOR flash)

- If loading NK.bin from a IDE device, it needs a 1M bytes NOR flash only
- If your boot screen is a big 24bit mode bitmap, it needs a 4M bytes NOR flash

DDR RAM: recommend two 128M bytes

Ethernet card port: RTL8100CL(RTL8139 compatibility) Enable (This device is for debug only)

SMP8634 MAC Enable

- SMP8634 vantage board(ES6 late version)

Pflash: recommend 64M bytes(NOR flash)

- If loading NK.bin from a IDE device, it needs a 1M bytes NOR flash only
- If your boot screen is a big 24bit mode bitmap, it needs a 4M bytes NOR flash

DDR RAM: recommend two 128M bytes

Ethernet card port: SMP8634 MAC Enable

RTL8100CL(RTL8139 compatibility) MINI PCI card(This device is for debug only)

2. SMP8634 peripheral:

USB Mouse

USB keyboard

Serial debug card

Serial port link cable

IR remote control

CRT or LCD TV monitor

Two Ethernet link cables

One MINI PCI RTL8100(RTL8139 compatibility) Ethernet card(For Vantage board only)

3. X86 PC

2.4G CPU

512M bytes Memory

60G bytes Hard disk

4. Ethernet switcher

4 ports 10/100M bps

B) Software platform

1. Microsoft Windows XP SP2 system

2. Microsoft windows CE platform builder 5.0 with MIPSII BSP

Recommend install follow updates.

a. "Windows CE 5.0 Platform Builder Yearly Update (2007)" for MIPSII

File name: WinCEPB50-071231-Product-Update-Rollup-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=A54779D5-F4A5-49F0-9E36-979D461F536C&displaylang=en>

b: "Windows CE 5.0 Platform Builder Monthly Update (January 2008)" for MIPSII

File name: WinCEPB50-080131-2008M01-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=98F0906A-45C6-4792-91FE-47A0E073A998&displaylang=en>

c: "Windows CE 5.0 Platform Builder Monthly Update (February 2008)" for MIPSII

File name: WinCEPB50-080229-2008M02-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=9AC7441F-3ADC-4E35-ABD1-FA4FBB90A132&displaylang=en>

d: "Windows CE 5.0 Platform Builder Monthly Update (March 2008)" for MIPSII

File name: WinCEPB50-080331-2008M03-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=AACEA4AE-E8C4-40C3-941C-D2A549A55DEB&displaylang=en>

e: "Windows CE 5.0 Platform Builder Monthly Update (April 2008)" for MIPSII

File name: WinCEPB50-080430-2008M04-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=4A3D4537-AB1B-4E8F-B735-348A64A9D8D6&displaylang=en>

f. "Windows CE 5.0 Networked Media Device Feature Pack" for MIPSII

File name: WinCEPB50-Product-Update-Rollup-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?familyid=BF17D6B0-4716-494B-9018-7DEEE9B91832&displaylang=en>

g. "WinCE 5.0 Networked Media Device Feature Pack - Cumulative Product Update Rollup Package (through 12/31/2007)" for MIPSII

File name: WinCEPB50_NMDFP-071231-Product-Update-Rollup-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=81C5D9F3-A1D1-4D0E-BF6A-D579D1C41077&displaylang=en>

h. "Windows CE 5.0 Networked Media Device Feature Pack Monthly Update (January 2008)" for MIPSII

File name: WinCEPB50_NMDFP-080131-2008M01-MIPSII.msi

Link:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=D64F66D4-9A4B-443F-8118-8D4CCA95FF68&displaylang=en>

3. SMP8634 BSP

a. Current version:2.0.0.0

b. Name: WindowsCE.5.0.BSP.2.0.0.0.zip

c. Directory Structure:

+2.0.0.0.bsp

+CEC

-smp863x.cec

-smp863x_atapi.cec

-smp863x_ehci.cec

-smp863x_eth.cec

-smp863x_i2c.cec

-smp863x_ohci.cec

-smp863x_pflash.cec

-smp863x_serial.CEC

-smp863x_smartcard.cec

+pflash_tools

-BOOT_STB.nb0

-dram_64.cmd

-dram_128.cmd

-genxenv.config

-genxenv.exe

-pflash_combo.config

-pflash_combo.exe

-pflash_CS2_64.mem

-readme.txt

-xenv_64.mem

-xenv_128_64.mem

-xreset.cmd

-xrpc_xload_bootstrap_cs2_b1000000_ES4_dev.bin

-xrpc_xload_bootstrap_cs2_b1000000_ES4_prod.bin

+Setxenv

+Setxenv

-main.cpp

-Makefile

-rmbasic.h

-sources

+xrpc

-xrpc.exe

-changes.txt

-readme.txt

-smp8634-envision.2.0.0.0.msi

d. BSP update

1) I2C update:

This update is about setting speed for I2C device

```
Name: Bsp_i2c.zip
Directory Structure:
+Bsp_i2c
+bsp_i2c_lib
  -bsp_i2c_lib.c
  -makefile
  -sources
+dll
  -i2c_iisr.c
  -i2c_iisr.def
  -i2c_iisr.h
  -makefile
  -sources
+sys
  -bsp_i2c.c
  -bsp_i2c.def
  -makefile
  -sources
+test
  -main.cpp
  -makefile
  -sources
-dirs
2) USB update:
  This update is about USB2.0 and it is for SMP8634 RevC/ES9 version
  chip only
  Name: SMP8634_USB_RevC_Optimization.zip
  Directory Structure:
+SMP8634_USB_RevC_Optimization
+ehci86xx
  -ehci86xx.def
  -ehci86xx.reg
  -makefile
  -resource.h
  -rmver.rc
  -sources
  -system.c
+ohci86xx
  -makefile
  -ohcd86xx.def
  -ohci86xx.reg
  -resource.h
  -rmver.rc
  -sources
  -system.c
+usb11
+common
  -cdevice.cpp
  -cdevice.hpp
  -cphysmem.cpp
  -cphysmem.hpp
  -globals.hpp
  -hcd.cpp
  -hcd.hpp
  -hcdrv.cpp
  -makefile
  -pipeabs.hpp
  -sources
  -sync.cpp
  -sync.hpp
+ohcd2
  -chw.cpp
  -chw.hpp
  -cohcd.cpp
  -cohcd.hpp
  -cpipe.cpp
  -cpipe.hpp
  -makefile
```

```
-sources
-transfer.cpp
-transfer.hpp
-dirs
+usb20
+EHCI
-cehcd.cpp
-cehcd.h
-chw.cpp
-chw.h
-cpipe.cpp
-cpipe.h
-ctd.cpp
-ctd.h
-makefile
-sources
-td.h
-trans.cpp
-trans.h
-usb2lib.cpp
-usb2lib.h
+USB2COM
-cdevice.cpp
-cdevice.hpp
-cphysmem.cpp
-cphysmem.hpp
-globals.hpp
-hcd.cpp
-hcd.hpp
-hcdrv.cpp
-makefile
-pipeabs.hpp
-sources
-sync.cpp
-sync.hpp
-dirs
+version
-rmver
-readme.txt
3) RTC update
Your board MUST have the RTC(RealTime Clock) hardware support
for this update to work.
Name: bsp_rtc-10-16-07.zip
Directory Structure:
+ bsp_rtc-10-16-07
+ WINCE500
+ PLATFORM
+ SMP863X
+Src
+Libs
+ bsp_rtc
- makefile
- readme.txt
- rmbasic.h
- rtc.c
- sources
- xos_xrpc.h
- xpurtc_xtask_load_bin_dev.h
- xpurtc_xtask_load_bin_prod.h
- xpurtc_xtask_unload_bin_dev.h
- xpurtc_xtask_unload_bin_prod.h
4. SMP8634 multimedia package
a. Current version: 2.0.0.10
b. Name: Multimedia.2.0.0.10.zip
c. Directory Structure:
+Multimedia.2.0.0.10
+include
- bitblt.h
```

- ddi86xxesc.h
- icapturesrc863x.h
- ids863x.h
- ihwdemux863x.h
- itsdemux.h
- jpeg_api.h
- smp863x_formats.h
- smp863x_ioctl.h
- SMP863x_RestoreFB.h
- +restorefb
 - + dev
 - SMP863x_RestoreFB.lib
 - + prod
 - SMP863x_RestoreFB.lib
- +retail
 - capsrc863x.map
 - ddi_86xx.map
 - ds863x.map
 - dswmapro.map
 - hdmi863x.map
 - hwdemux863x.map
 - smp863x.map
 - swjpeglib.map
 - tsdemux.map
 - wave863x.map
 - jpeg_api.lib
 - capsrc863x.pdb
 - ddi_86xx.pdb
 - ds863x.pdb
 - dswmapro.pdb
 - hdmi863x.pdb
 - hwdemux863x.pdb
 - jpeg_api.pdb
 - smp863x.pdb
 - swjpeglib.pdb
 - tsdemux.pdb
 - wave863x.pdb
 - capsrc863x.rel
 - ddi_86xx.rel
 - ds863x.rel
 - dswmapro.rel
 - hdmi863x.rel
 - hwdemux863x.rel
 - smp863x.rel
 - swjpeglib.rel
 - tsdemux.rel
 - wave863x.rel
 - capsrc863x.dll
 - ddi_86xx.dll
 - ds863x.dll
 - dswmapro.dll
 - hdmi863x.dll
 - hwdemux863x.dll
 - smp863x.dll
 - swjpeglib.dll
 - tsdemux.dll
 - wave863x.dll
- +samples
 - +BootFB
 - main.cpp
 - Makefile
 - rmbasic.h
 - sources
 - +HwPlayJpeg
 - HwPlayJpeg.cpp
 - sources
 - HwPlayJpeg.h
 - HwPlayJpeg.rc

- makefile
- resource.h
- readme_capturesrc863x.txt
- readme_ddi_86xx.txt
- readme_ds863x.txt
- readme_dswmapro.txt
- readme_hdmi863x.txt
- readme_hwdemux863x.txt
- readme_hwjpeg.txt
- readme_smp863x.txt
- readme_tsdemux.txt
- readme_wave863x.txt
- Windows CE Boot Screen.txt
- whatsnew.2.0.0.10.txt
- d. MultiMedia update

5. SMP8634 XOS version

Version: E0

FileName: xrpc_xload_xosu-xosMe0-8634_ES4_dev.bin

6. The different files for development chip(Es6/Es7/Es9) and production chip(Rev A/B/C) in BSP

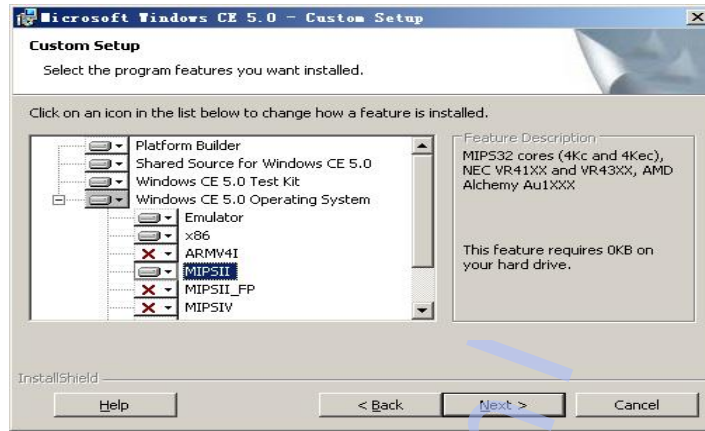
For follow files , there are two version in the BSP, one for development chip, one for production chip, if you use wrong version for your system, it can make the board fault, so please take careful when you use this mode file in your system.

- a. Signed file
 - xrpc_xload_bootstrap_cs2_b1000000_ES4_dev.bin
 - xrpc_xload_bootstrap_cs2_b1000000_ES4_prod.bin
- b. Framebuffer library
 - SMP863x_RestoreFB.lib

SIGMA DESIGNS®

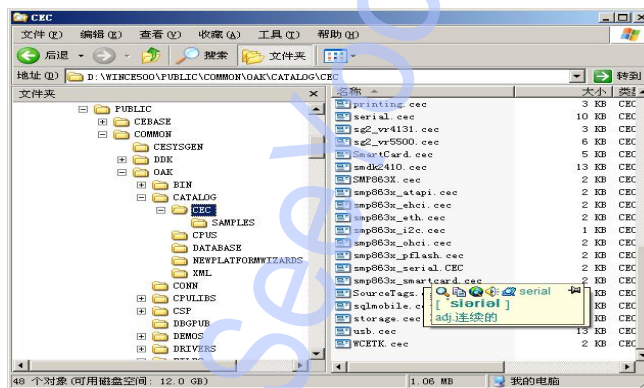
2) How to build up the SMP8634 wince platform builder

- A) Build up a Microsoft windows XP SP2 system on a X86 PC
- B) Install Microsoft windows CE platform builder 5.0 with MIPSII BSP to the system as pic1 shows.



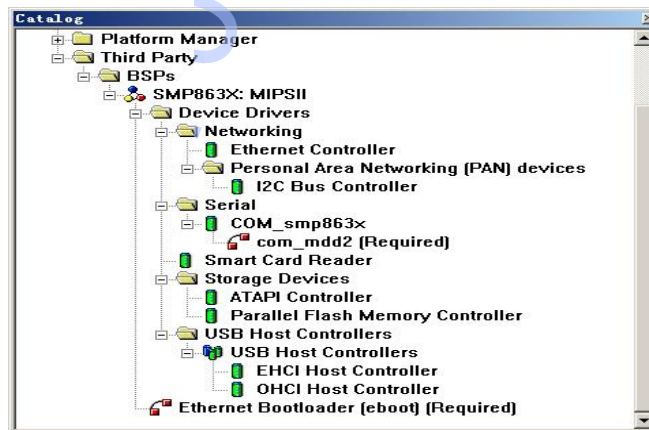
<pic1>

- C) Install all Update packages for Microsoft windows CE 5.0 platform builder
- D) Install SMP8634 wince BSP(smp8634-envision.2.0.0.0.msi), upgrade BSP
- E) Copy all nine CEC files (smp863x.cec,smp863x_atapi.cec,smp863x_eth.cec,smp863x_i2c.cec,smp863x_ohci.cec,mp863x_serial.CEC, smp863x_pflash.cec,smp863x_smartcard.cec, smp863x_ehci.cec) to "\\WINCE500\\PUBLIC\\COMMON\\OAK\\CATALOG\\CEC" directory



<pic2>

- F) Open Microsoft windows CE 5.0 platform builder, use "File---Manage Catalog Items..." menu to open "Manage Catalog Items" dialog, use "Import" button to import all eight CEC files to platform builder, and make sure all SMP8634 driver features are in Catalog windows as pic3 shows



<pic3>

3) How to run the CE system on SMP8634 board

A) Make the CE bootloader

There is not a default combo CE bootloader in BSP, but all of the making bootloader tools are in pflash_tools directory. You can use them to make a CE bootloader.

All EXE files are DOS mode command.

1. Make the XENV file(xenv_128_128.mem)

Set genxenv.config as follow sample, and use this new genxenv.config to make a new xenv_128_128.mem file.

The command line is "**genxenv e xenv_128_128.mem genxenv.config**".

```
#
# Hardware config file for SMP863x platform (xenv)
# Default for SMP8634 STB board (a.k.a Envision)
#
```

```
# -----
# xos section
# -----
# boot xrpc/xload offset:
x.boot = 0x8000
# dram stuffing
# [29:20] - size of DRAM2, in MB, if present
# [19:10] - size of DRAM1, in MB, if present
# [ 9: 0] - size of DRAM0, in MB.
#x.ds = 0x10040 # DRAM0 64M + DRAM1 64M
x.ds = 0x20080 # DRAM0 128M + DRAM1 128M
# dram timings/delays
x.d0.cfg = 0xf34111ba # DRAM0 128M
#x.d0.cfg = 0xe34111ba # DRAM0 64M
#x.d0.dl0 = 0x000a4444
x.d1.cfg = 0xf34111ba # DRAM1 128M
#x.d1.cfg = 0xe34111ba # DRAM1 64M
#x.d1.dl0 = 0x000a4444
# dram test
x.dt = 1
# pll setting
#x.pll3 = 0x01020057
#x.mux = 0x201
# frequency setting
x.csf = 0x2
```

2. Make the pflash combo file(SMP8634CE128M.mem)

Set pflash_combo.config as follow sample,use this new config file to make the CE loader file(SMP8634CE128M.mem).

The command is "**pflash_combo SMP8634CE128M.mem pflash_combo.config**".

```
# parallel flash memory configuration for XOS versions > a0
# encrypted boot stub located at offset 0x800 and pointed by x.boot=0x800 in xenv.mem
#
```

```
0x00000000 xenv_128_128.mem
0x00008000 xrpc_xload_bootstrap_cs2_b1000000_ES4_dev.bin
0x00010000 boot.nb0
```

The SMP8634CE128M.mem must be 576K bytes big size.

The boot.nb0 file of each version BSP is different, so if BSP is upgraded, we must update the boot.nb0 too

Build a new CE project with new version BSP, get the boot.nb0 file, use it to replace the default boot.nb0 file or download boot.bin to SMP8634 board with CE platform builder.

If the CE bootloader is made for a production SMP8634 board(Rev A, Rev B or Rev C version), we must use [xrpc_xload_bootstrap_cs2_b1000000_ES4_prod.bin](#) file to replace the [xrpc_xload_bootstrap_cs2_b1000000_ES4_dev.bin](#) file.

3. Write the CE bootloader(SMP8634CE128M.mem) to Pflash0(BOOT FLASH) of SMP8634 board from beginning address with Yamon command or other tools.

There are three ways to do that.

- a. Use a external flash programmer to program the flash image directly to the flash.

Of course you have to remove the flash chip from the board first.

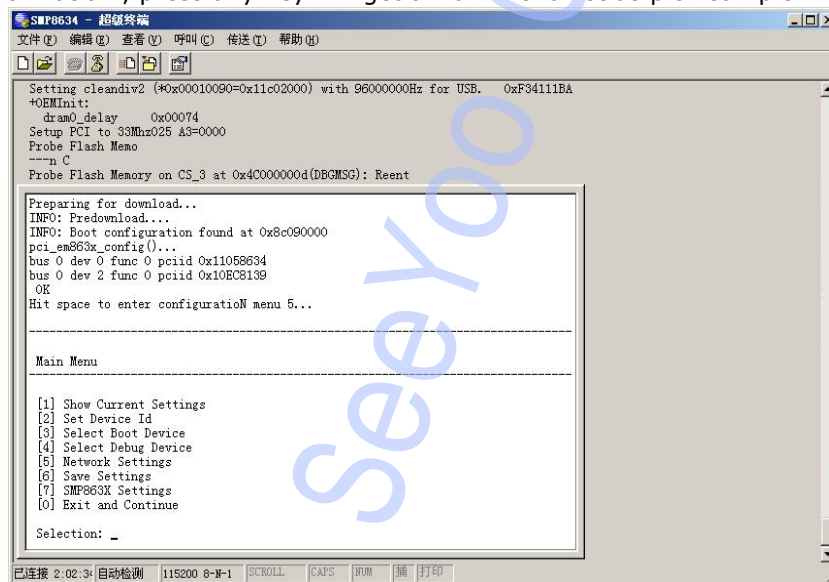
- b. Use a jtag probe to do that, but it involves a US\$2500 jtag hardware to do that. If you don't have it, you can ignore this option.

c. Use Yamon prompt to program WindowsCE boot loader. I'll show you the step below. Please be VERY careful when you do this. If the process fail in the middle, it will crash Yamon and make your board not bootable, and you have to go back to option 1.

Here is how you program the file to on-board flash by Yamon.
Assume I want to program file "file0.bin" to address 0 of flash.

1. UUENCODE the file by uuencode command in x86 linux.
>**uuencode file0.bin x > file0.bin.uuencode**
2. In Yamon prompt, run
YAMON> load uu 0xb0100000
3. Now the Yamon is waiting for the file from serial port. Send the file0.bin.uuencode file to SMP8634 board through serial terminal. If you are using TeraTerm, you can do it by select "File->Send File".(Must with text mode)
4. When downloading is done, the file size of the received file is given in YAMON.
5. Compare the reported size to the filesize of file0.bin (NOT file0.bin.uuencode) and make sure they are exactly the same. No more, no less.
6. In Yamon prompt, run
YAMON> pflash write 0x0 0xb0100000 <reported file size>
This command tell yamon to program the <reported file size> of data in 0xb0100000 to flash address 0x0.

4. Reboot the board, the Serial Output window of PC will shows the CE loader's bootup information , press any key will get a Main Menu list as pic4 sample.



```
SMP8634 - 超級終端
文件(F) 編輯(E) 查看(V) 呼叫(C) 传送(T) 帮助(H)
Setting cleandiv2 (*0x00010090=0x11c02000) with 96000000Hz for USB. 0xF34111BA
+EMInit:
 dram0_delay 0x00074
 Setup PCI to 33Mhz025 A3=0000
 Probe Flash Memo
 --n C
 Probe Flash Memory on CS_3 at 0x4C000000d(DBGMSG): Reent

Preparing for download...
INFO: Preadownload...
INFO: Boot configuration found at 0x8c090000
pci_em863x_config()...
bus 0 dev 0 func 0 pciid 0x11088634
bus 0 dev 2 func 0 pciid 0x10EC8139
OK
Hit space to enter configuration menu 5...

-----
Main Menu
-----

[1] Show Current Settings
[2] Set Device Id
[3] Select Boot Device
[4] Select Debug Device
[5] Network Settings
[6] Save Settings
[7] SMP863X Settings
[0] Exit and Continue

Selection: _
```

<pic4>

B) Make the SMP8634 specific CE Nk.bin

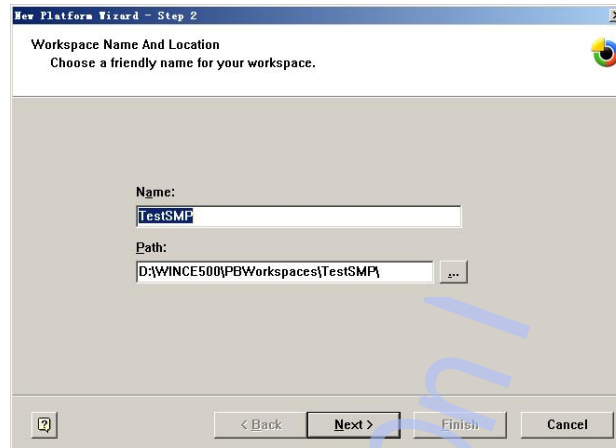
The SMP8634 CE system will support follow features,

- GDI output from HDMI, S-Video, AV, YCbCr, YPbPr
- DirectDraw DDI support, Alpha blending support
- HD GDI and Video output(480P,720P,1080I,1080P), LVDS output(24bit RGB)
- USB Mouse/Keyboard input(USB1.1/2.0), IR remote control, COM serial port input, I2C control, mini-PCI
- Hard Disk, USB storage device(USB2.0), FAT/FAT32 partition, Pflash Memory storage device, Smart card reader
- Ethernet Network, RTL8139 and SMP8634 MAC ETH two devices, TCP/IP
- Media Decode for
MPEG1,
Program Stream(MPEG2),
Transport Stream(h.264, mpeg-1, vc-1, and mpeg-2 codec video, ac3, mpeg audio, and wma-ts codec audio), HD Transport Stream(720P,1080I/P),
WMV(WMV9 video, WMA/WMA Pro audio), HDWMV(720P,1080I/P),
AVI(VCM/WMV9/DIVX/XVID/H264 + MPA/MP3/AC3), HD AVI(720P,1080I/P),
WMA, Mp3,
Hardware TS demux (From SPI/SSI port or local file)
- System WAV audio output
- Image decode for JPEG, PNG, BMP, GIF(static), Hardware JPEG decode

SIGMA DESIGNS®

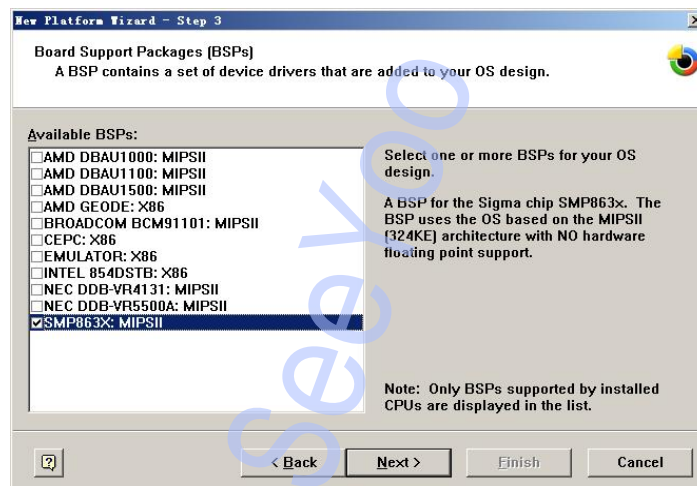
- Network multimedia support, IP Media(WMT) Via MMS protocol(From Windows Media Server), IP Media(WMT, Mpeg) Via Http protocol, Media On SMB server
- Web server, FTP server, Telnet Server, SMB sever

1. Run "New platform.." menu command to wizard build CE project
 - a. Enter the name you would like to name the platform.



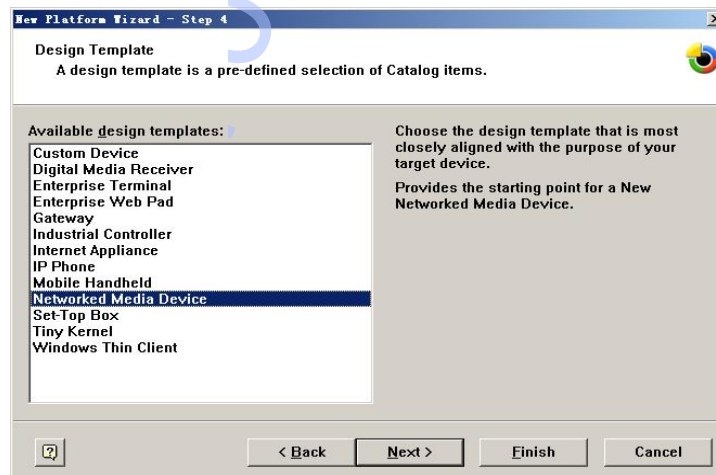
<pic5>

- b. Select "SMP863X: MIPSII" BSP.



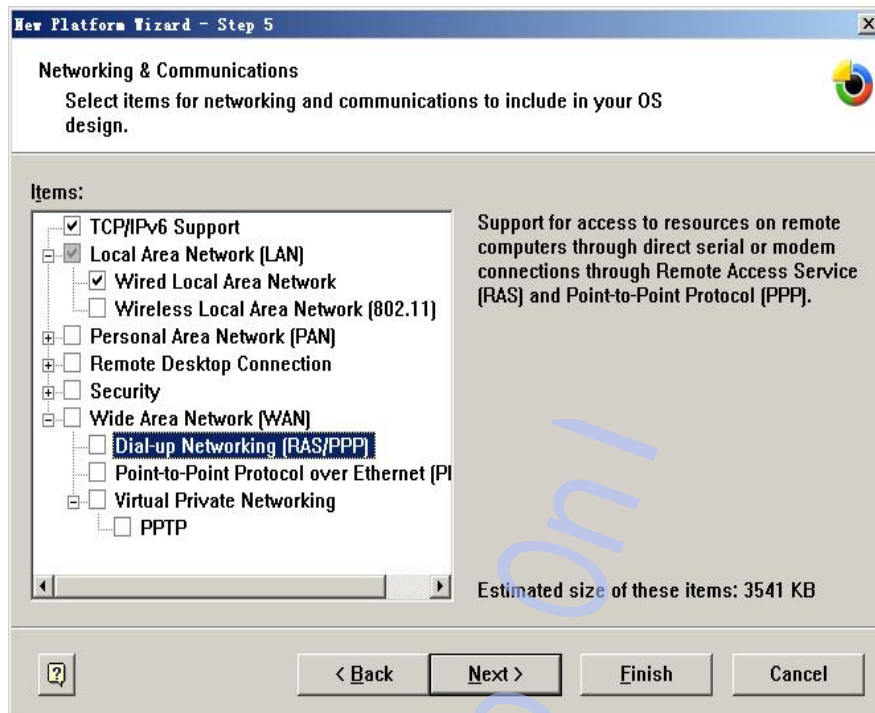
<pic6>

- c. Select the "Networked Media Device" design. (To enable this design, it needs copy the Networked_Media_Device.xml file to "X:\WINCE500\public\common\OAK\CATALOG\Newplatformwizards" directory, the xml file is downloaded from MSDN, it's a NMD template design wizard from Microsoft.)



<pic7>

d. Add/Remove the networked drivers, finish the new platform wizard.

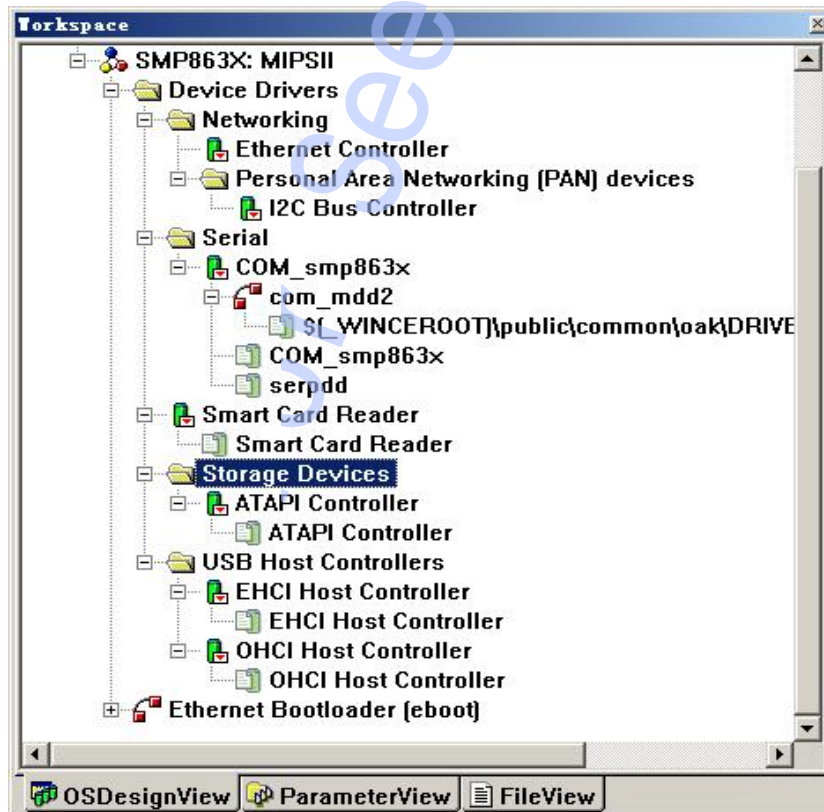


<pic8>

2. Add/Remove all specific driver from Catalog window

a. Add follow SMP8634 drivers to current CE project

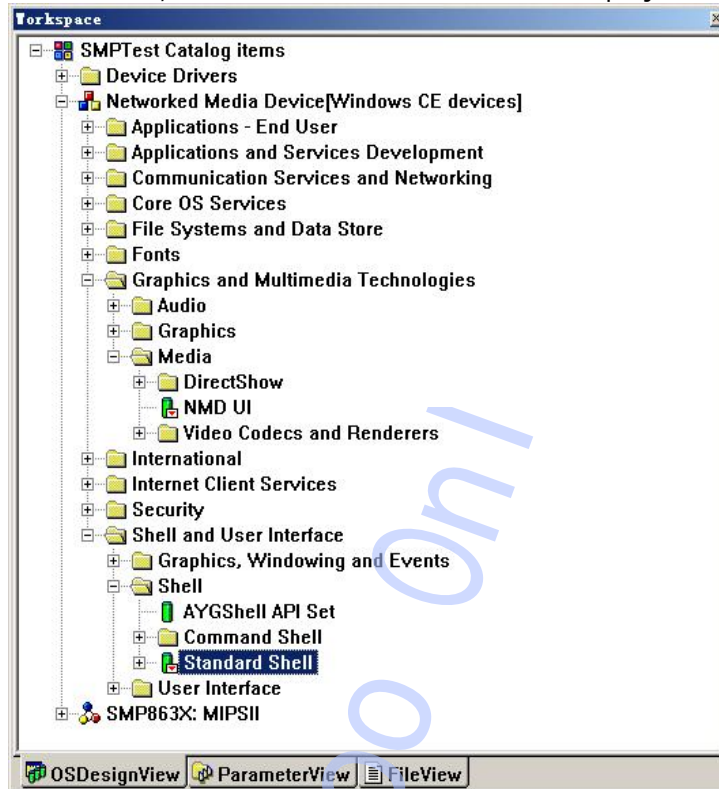
Ethernet Controller	I2C Bus Controller	Ethernet Bootloader (eboot)
Smart Card Reader	ATAPI Controller	COM_smp863x
OHCI Host Controller	EHCI Host Controller	



<pic9>

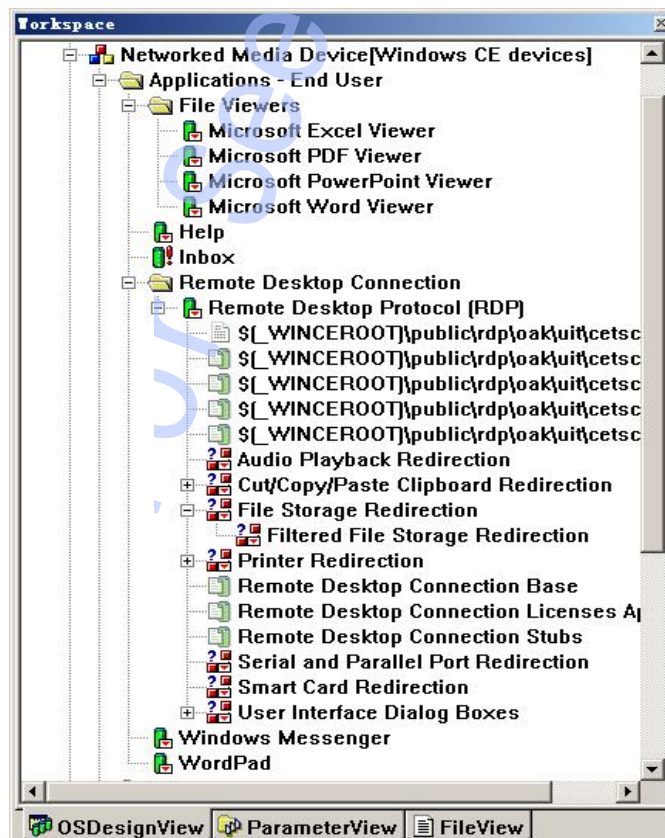
SIGMA DESIGNS®

- c. Edit all device module as sample shows
- Delete "NMD UI" feature, add "Standard Shell" feature to CE project



<pic10>

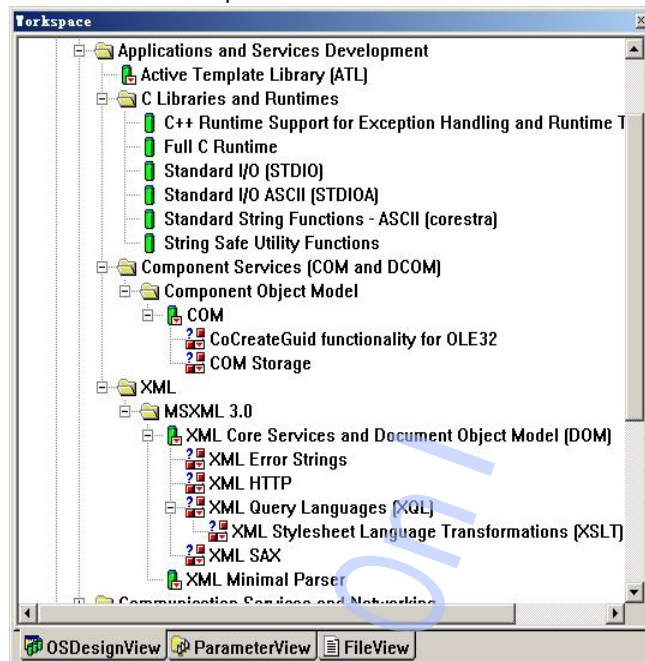
- Applications-End User, if you don't like Microsoft application, You can delete them.



<pic11>

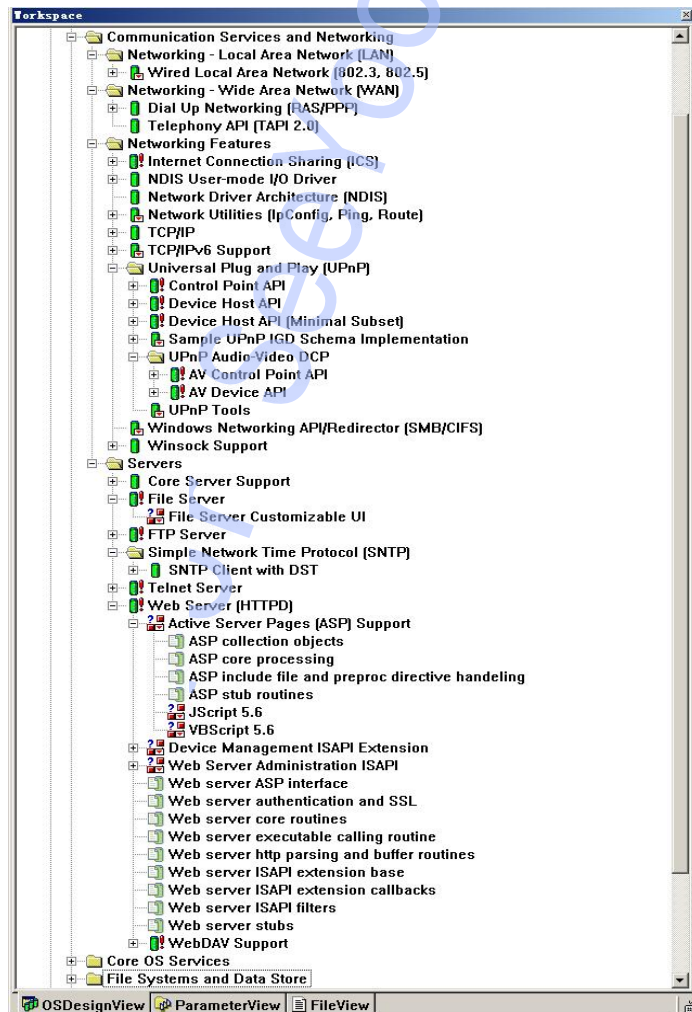
SIGMA DESIGNS®

- Applications and Services Development



<pic12>

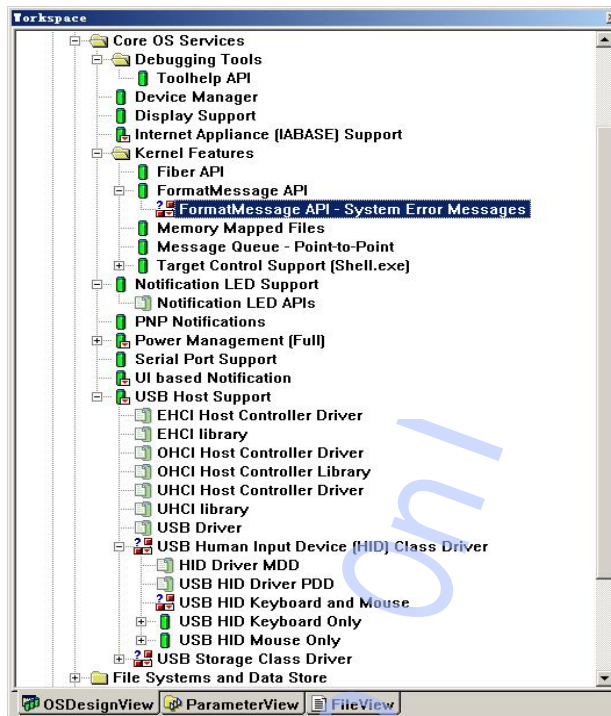
- Communication Services and Networking



<pic13>

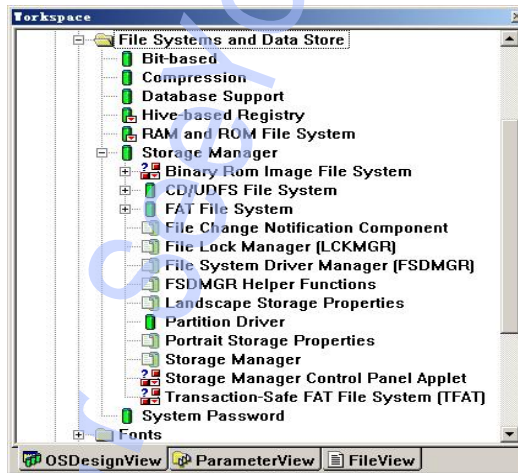
SIGMA DESIGNS®

- Core OS Services



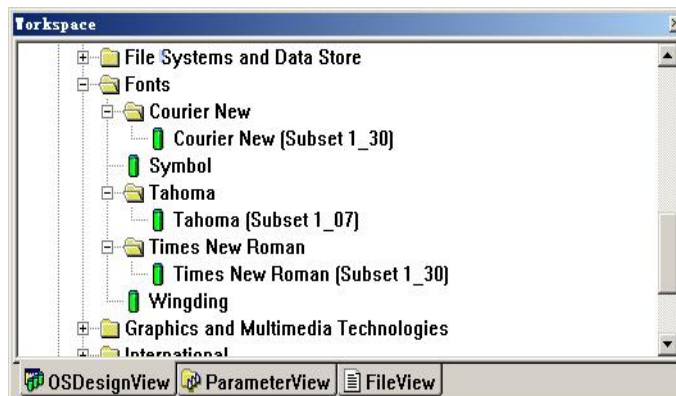
<pic14>

- File Systems and Data Store



<pic15>

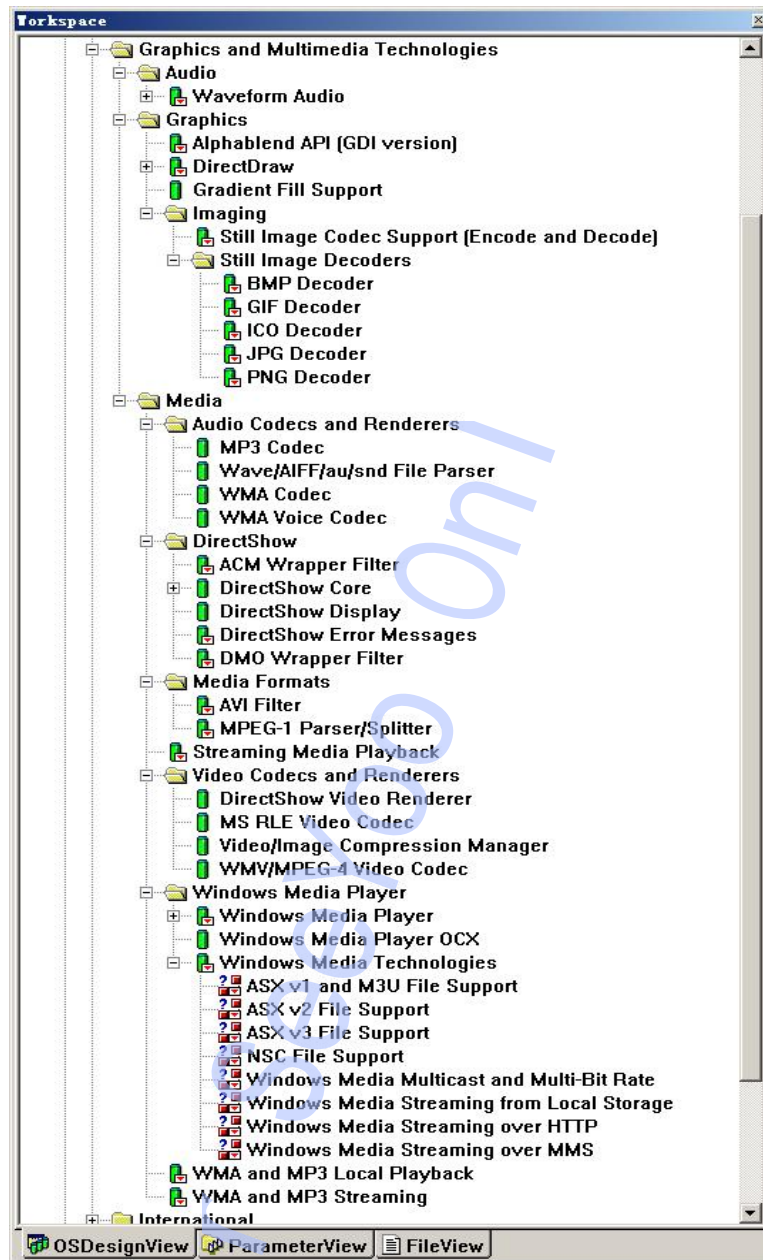
- Fonts



<pic16>

SIGMA DESIGNS®

- Graphics and Multimedia Technologies



<pic17>

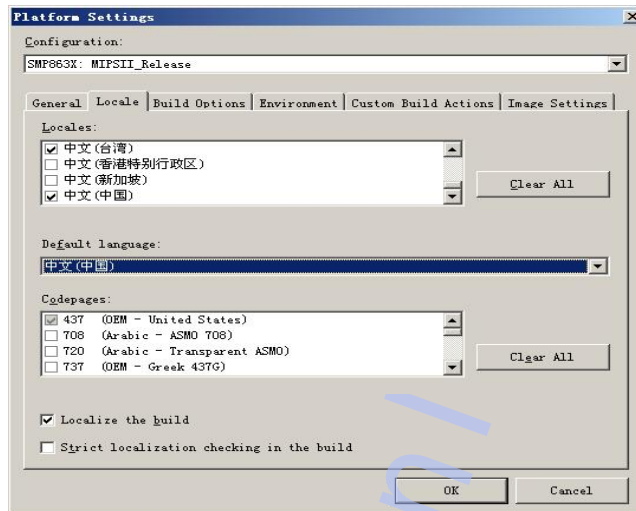
- International



<pic18>

SIGMA DESIGNS®

Set the locale language support, Use "Platform----Settings...." Menu command to open "Platform Settings" dialog, turn the "Locale" page,

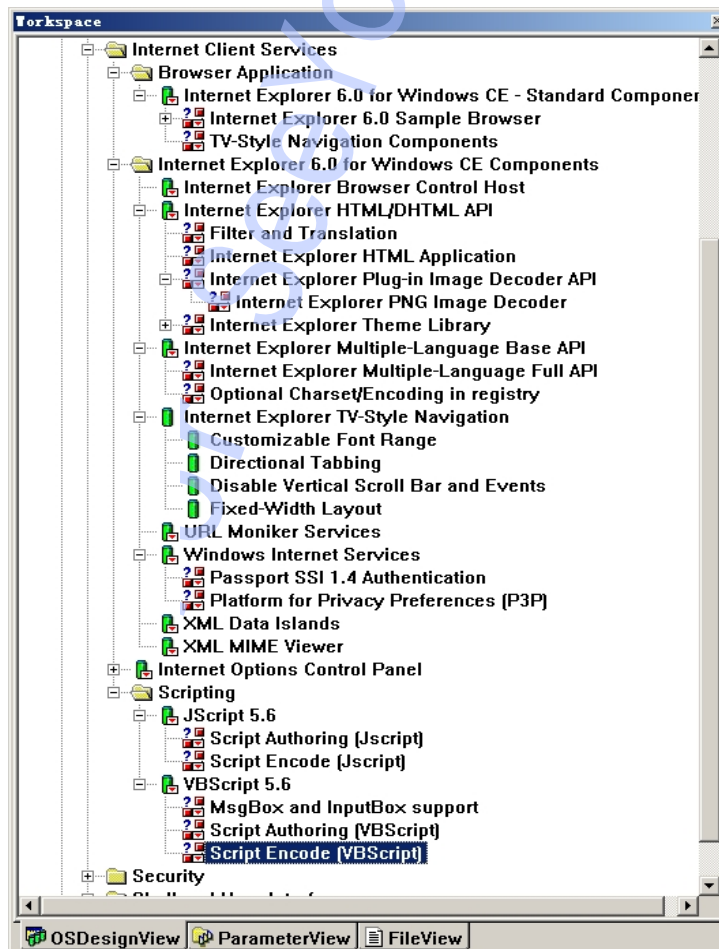


<pic19>

Use first "Clear All" button to clear system default Locales settings, select the languages that you want to support, set the Default language that you want to show with CE GUI. Click the "OK" button to close the dialog and enable the Locale Specific setting.

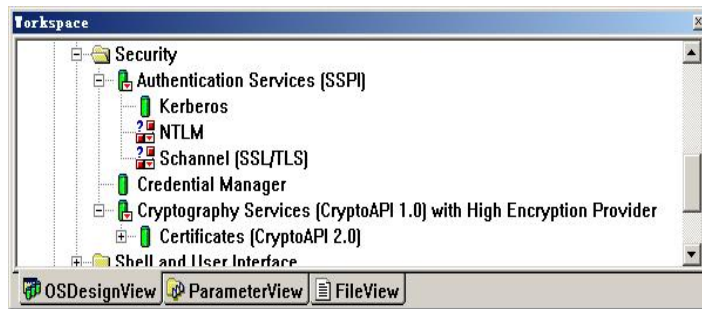
Note: Adding "Agfa AC3 Font Compression" to CE project can make the Chinese Fonts library to be small, and in fact, the catalog is included in "Chinese(Simplified)" item group, not included in "Korean" group.

- Internet Client Services



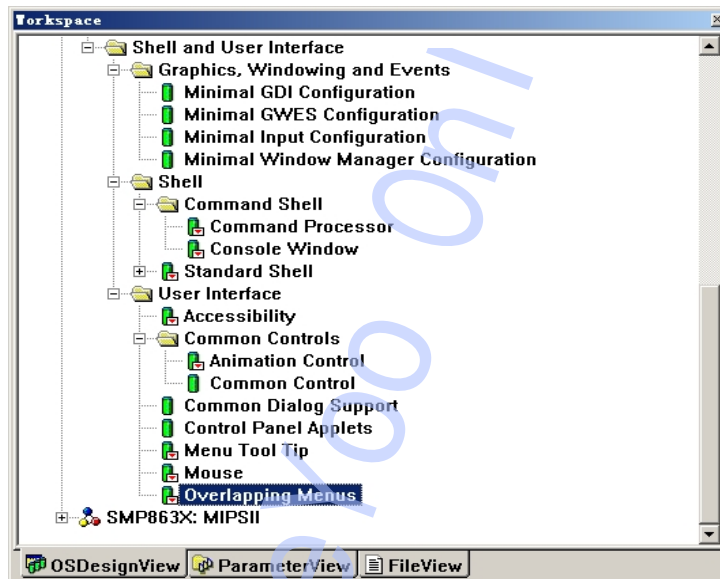
<pic20>

- Security



<pic21>

- Shell and User Interface

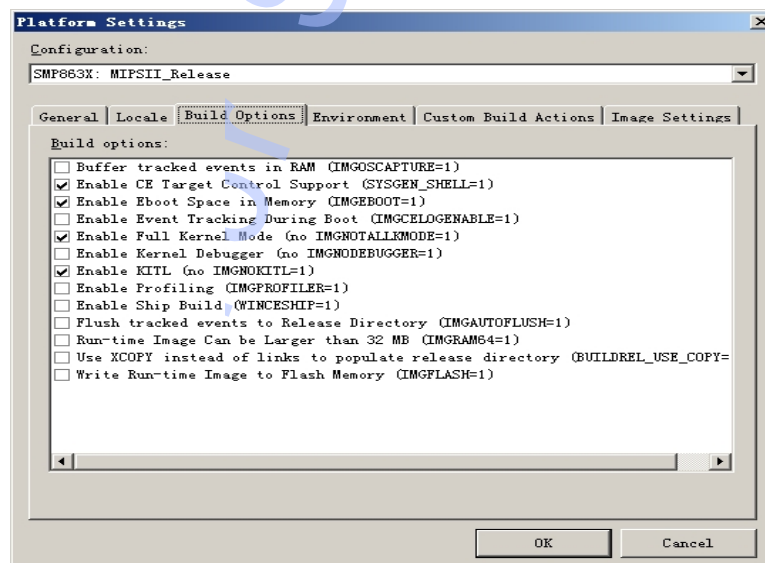


<pic22>

3. Set the build options and Environment

a. Build options

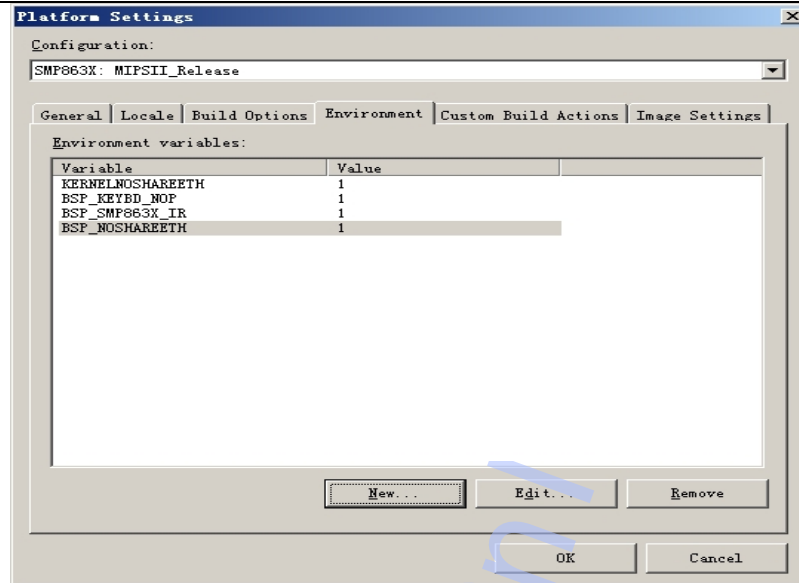
Use "Platform----Settings..." Menu command to open the "Platform Settings" dialog, turn to "Build Options" page, set it as pic24 sample



<pic23>

b. Build Environment

On "Platform Settings" dialog, turn to "Environment" page,



<pic24>

Add follow Environment Variable Value to project.

Variable	Value	Comment
KERNELNOSHAREETH	1	Disable VIMINI1 (ignore for Vantage)
BSP_NOSHAREETH	1	Disable VIMINI1 (ignore for Vantage)
BSP_KEYBD_NOP	1	Enable USB keyboard
BSP_SMP863X_IR	1	Enable SMP8634 IR remote control
PRJ_ENABLE_FSMOUNTASROOT	1	Enable HIVE-Based Registry(HDD)
PRJ_ENABLE_REGFLUSH_THREAD	1	Enable HIVE-Based Registry(HDD)
PRJ_BOOTDEVICE_ATAPI	1	Enable HIVE-Based Registry(HDD)

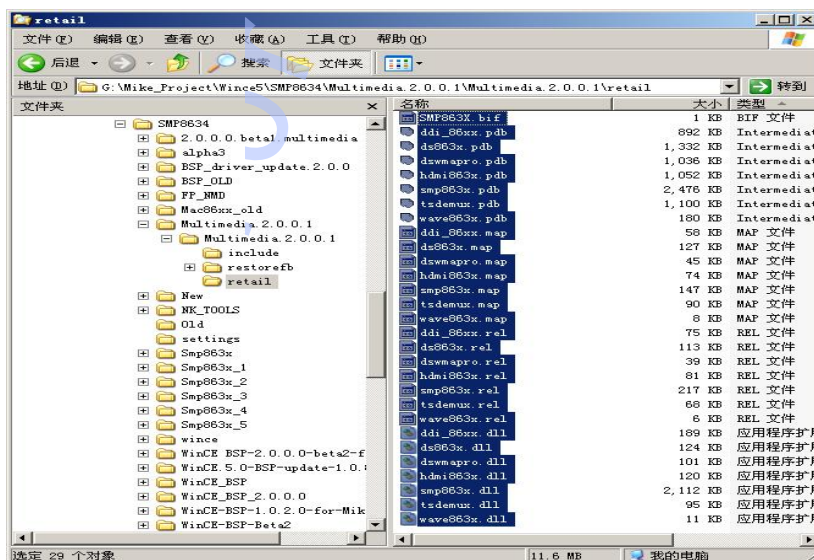
If you use Vantage Platform, you don't need add KERNELNOSHAREETH and BSP_NOSHAREETH.

c. Click "OK" button to close dialog and enable all settings

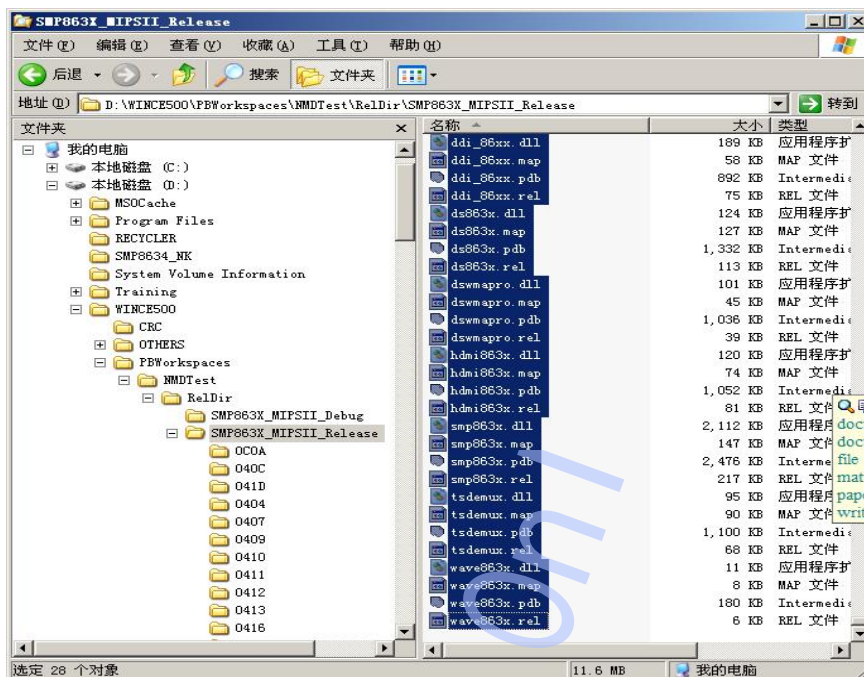
4. Run "Build OS----Build and Sysgen" menu command to build the CE project. Wait for the building is successful complete, it needs more time, maybe 1-3 hours. Then we can have a cup of tee or coffee.

If the SMP8634 BSP2.0.0.0 is updated, please run "Build OS----Build and Sysgen Current BSP" and "Build OS---- Sysgen" again. This step needs about 1 hour.

5. Copy all files in retail directory of multimedia package(pic26) to "\WINCE500\PBWorkspace\your CE project name)\RelDir\SMP863X_MIPSII_Release"(pic27) CE project directory



<pic25>



<pic26>

6. Open the **platform.bib** file in "\\WINCE500\PBWorkspace\your CE project name)\RelDir\SMP863X_MIPSI_Release" directory, add follow settings to it under ";@CESYSGEN IF CE_MODULE_DEVICE" line.

;USB keyboard

KbdNopUS.dll \$(_FLATRELEASEDIR)\KbdNopUS.dll NK SH

;SMP8634

capsrc863x.dll \$(_FLATRELEASEDIR)\capsrc863x.dll NK SH
 ddi_86xx.dll \$(_FLATRELEASEDIR)\ddi_86xx.dll NK SH
 ds863x.dll \$(_FLATRELEASEDIR)\ds863x.dll NK SH
 dswmapro.dll \$(_FLATRELEASEDIR)\dswmapro.dll NK SH
 hdmi863x.dll \$(_FLATRELEASEDIR)\hdmi863x.dll NK SH
 hwdemux863x.dll \$(_FLATRELEASEDIR)\hwdemux863x.dll NK SH
 smp863x.dll \$(_FLATRELEASEDIR)\smp863x.dll NK SH
 swjpeglib.dll \$(_FLATRELEASEDIR)\swjpeglib.dll NK SH
 tsdemux.dll \$(_FLATRELEASEDIR)\tsdemux.dll NK SH
 wave863x.dll \$(_FLATRELEASEDIR)\wave863x.dll NK SH

Find the pflash device dll file line.

pflash.dll \$(_FLATRELEASEDIR)\pflash.dll NK SH
 pflash2.dll \$(_FLATRELEASEDIR)\pflash2.dll NK SH

Disable it as follow sample.

; pflash.dll \$(_FLATRELEASEDIR)\pflash.dll NK SH
 ; pflash2.dll \$(_FLATRELEASEDIR)\pflash2.dll NK SH

Close and save the platform.bib file.

7. Open the **config.bib** file in "\\WINCE500\PBWorkspace\your CE project name)\RelDir\SMP863X_MIPSI_Release" directory, add follow settings to the end of file.

;system memory setting 25%::75%
 FSRAMPERCENT=0x202020

Find follow four line settings.

GLOBAL_MEM 90202800 00100800 RESERVED ;1MB for EMHWLIB REQUIRED
 CONTIGUOUS MEMORY
 EXTENSION_DRAM 90303000 00E1D000 RESERVED ;15MB extension dram
 NK 92600000 02000000 RAMIMAGE ;32MB for NK.BIN image
 RAM 94600000 03000000 RAM ;48MB for CE applications

Modify them as follow sample

SMP863X 90202800 01B00800 RESERVED ;27MB for SMP863X driver
 NK 91D10000 02A00000 RAMIMAGE ;42MB for NK.BIN image
 RAM 94710000 032F0000 RAM ;50MB for CE applications

Note: the detail information about the config.bib setting

SIGMA DESIGNS®

The sample config.bib file is:

MEMORY

IRQHANDLER	90000000	00200000	RESERVED	;2MB for IRQ handler
ARGS	90200000	00000800	RESERVED	;2KB info from bootloader to system
IDE_DMA	90200800	00002000	RESERVED	;8KB IDE DMA side buffer
SMP863X	90202800	01B00800	RESERVED	;27MB for SMP863X driver
NK	91D10000	02A00000	RAMIMAGE	;42MB for NK.BIN image
RAM	94710000	032F0000	RAM	;50MB for CE applications
XOS_PRIVATE	97A00000	00600000	RESERVED	;6MB for xos >= a6

IRQHANDLER

The first 2MB of DRAM0 is reserved for usage by Sigma Designs. This is a requirement and must be respected.

ARGS

This is a 2Kb region used by the bootloader to send arguments to the kernel. This is a hard coded location coded in the bootloader and the kernel. This should NOT be changed.

IDE_DMA

This is a reserved 8Kb memory location to be used by the SMP8634 atapi driver. This location is hard coded in the driver, and should not be changed unless the driver is changed accordingly.

SMP863x

This is a reserved portion of memory to be used by the SMP863X.DLL driver. The location and size of this memory can be changed. See the SMP863X.DLL documentation for more details.

In this sample, it is set as 9 + 8 + 9 + 1 model, it's all 27M bytes

9M ---- For DDI setting, 1920X1080X4+2048, less if your desk size is small

8M ---- For DirectDraw setting, less if you disable Directdraw

9M ---- For Second decoder, less if you disable second decoder

1M ---- For Basic setting, it is must reserved

EXTENSION_DRAM0

This part is disable in sample config.bib, if you don't want use memory as this mode, please ignore this part

```
EXTENSION_DRAM0 90303000 00E1D000 RESERVED ; 15MB extension dram
```

This is a portion of memory that is used as an "extension" to the "RAM" area. This area is seen by the kernel as available for use. The location and size of this memory can be changed, but it is hard coded in the

/SMP863X/Src/Kernel/Oal/init.c file:

```
#define EXTENSION_DRAM_START_ADDRESS (0x10303000)
```

```
#define EXTENSION_DRAM_LENGTH (0x00E1D000)
```

If you change the location/size of this memory area, you must change init.c accordingly and re-compile the kernel.

NK

This is where the kernel gets loaded. The location is hard coded in the files:

```
SMP863X\Src\Inc\image_cfg.h(32):
```

```
#define IMAGE_WINCE_CODE_PA_START 0x1D10000
```

```
SMP863X\Src\Kernel\Kern\sources(14):
```

```
EXEBASE=0x91D10000
```

```
SMP863X\Src\Kernel\Kernkitl\sources(14):
```

```
EXEBASE=0x91D10000
```

```
SMP863X\Src\Kernel\Kernkitlprof\sources(13):
```

```
EXEBASE=0x91D10000
```

Changing this location requires you to change the appropriate files, and rebuilding the bootloader and kernel.

RAM

This is the normal RAM area the kernel sees.

XOS_PRIVATE

This is a special area reserved for use by XOS. The size and location can only be changed via xenv variables. It is advised to leave this default. The default location is the last 6MB of DRAM0. So, if DRAM0 had 128MB, the physical address start would be 0x17A00000. If DRAM0 had 64MB, the physical address start would be 0x13A00000.

8. Open the **platform.reg** file in "\WINCE500\PBWorkspace\\RelDir\SMP863X_MIPSII_Release" directory, add follow registry settings to the end of file.

```
;GDI 2.0.0.10 verion
[HKEY_LOCAL_MACHINE\System\GDI\Drivers]
"Display"="ddi_86xx.dll"
;
; See smp863x_formats.h for the proper hex values for each video mode
;
"DigitalOutput"=dword:21
"DigitalColorSpace"=dword:3
"MainAnalogOutput"=dword:6f
"MainAnalogColorSpace"=dword:4
"ComponentAnalogOutput"=dword:65
"ComponentMode"=dword:6
"ComponentOrder"=dword:0
"ComponentColorSpace"=dword:4
"VGAOutput"=dword:0 ; 0 = disable VGA output, 1 = enable VGA output.
; Also, when it's 1, need to set ComponentAnalogOutput to CVT_xxx or VESA_xxx,
; ComponentMode to RGB_SCART, ComponentColorSpace to RGB_0_255
"ScreenWidth"=dword:00000500 ;1280
"ScreenHeight"=dword:000002D0 ;720
"OutputPosX"=dword:100 ; output window position x
"OutputPosY"=dword:100 ; output window position y
"OutputPosWidth"=dword:e00 ; output window position width
"OutputPosHeight"=dword:e00 ; output window position height
"MemorySize"=dword:00800000
; extra memory size for h/w acceleration, see note in the smp863x.reg
; regarding the PRIMARY_DISPLAY_SURFACE_DRAMBANK
"EnableHwCursor"=dword:1 ; 0 = disable, 1 = enable
"LiveDetectHdmiConnection"=dword:1 ; 0 = driver won't do live hdmi detection, 1 =
check live hdmi connection
"EnableHDCP"=dword:0 ; 0 = disable the HDCP, 1 = enable the HDCP
"DefaultKeyColor"=dword:00010101 ; In RGB format: R = 3rd byte, G = 2nd byte,
B = 1st byte
"EnableEDIDDetection"=dword:0 ; 0 = disable, 1 = enable. Disable/enable the
HDMI preferred mode.
"HDMI2CModule"=dword:2 ; 0 = software, 1 = hardware, 2 = built-in hdmi
"InvertClock"=dword:1 ; 0 = do not invert digital video clock, 1 = invert
(default is inverted)
"LVDS_Enable"=dword:0
"LVDS_GPIOFieldIDOutput"=dword:B ; board dependent
"LVDS_GPIOPanelOn"=dword:E ; board dependent

; Default custom digital output video mode - 720p59
[HKEY_LOCAL_MACHINE\System\GDI\Drivers\CustomDigitalVideoMode]
"PixelClock"=dword:46BD550
"HActive"=dword:500
"HFrontPorch"=dword:6E
"HSyncWidth"=dword:28
"HBackPorch"=dword:DC
"HSyncPolarity"=dword:1 ; TRUE: positive, FALSE: negative
"VActive"=dword:2D0
"VFrontPorch"=dword:5
"VSyncWidth"=dword:5
"VBackPorch"=dword:14
"VSyncPolarity"=dword:1 ; TRUE: positive, FALSE: negative
"Interlaced"=dword:0
```

```

; Sigma Designs built-in driver for low-level access
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\SMP86xx]
  "Dll"="smp863x.dll"
  "Prefix"="SDH"
  "Index"=dword:1
  "Order"=dword:1
  "SMP863X_RESERVED_START_DRAM0"=dword:10202800 ; starting address of
reserved memory in dram0
  "SMP863X_RESERVED_DRAM0_SIZE" =dword:01B00000 ; dram0 needs at 1MB
(more if primary surface is allocated in dram0)
  "SMP863X_RESERVED_START_DRAM1"=dword:0C000000 ; starting address of
reserved memory in dram1
  "SMP863X_RESERVED_DRAM1_SIZE" =dword:00000000 ; dram1 needs at 40MB
(less if primary surface is allocated in dram0)
  "PRIMARY_DISPLAY_SURFACE_DRAMBANK"=dword:0 ; dram bank to
allocate primary display surface (0 or 1)
  "DEFAULT_ASPECT_RATIO_X"=dword:10 ; default aspect ratio for the
output, horizontal direction
  "DEFAULT_ASPECT_RATIO_Y"=dword:9 ; default aspect ratio for the
output, vertical direction

; ds863x.dll
; @CESYSGEN IF QUARTZ_IMAGE
; Sigma Designs Renderer Filter
[HKEY_CLASSES_ROOT\Filter\{caa414f0-c7c3-11d4-a914-0080ad91bd94}]
IF INITDSHOWFILTERNAMES
@="Sigma Designs Filter"
ENDIF
;"DelaySettingCodecTypes"=dword:1 ; delay setting of the codec types - ONLY enable
this if you are using the .asx playlist
"FlushDisplayOnStop"=dword:0
"FlushDisplayOnBeginFlush"=dword:0
"FlushDisplayOnDiscontinuity"=dword:0

[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}]
IF INITNODSHOWFILTERCLASSNAMES !
@="Sigma Designs Filter"
ENDIF
"Merit"=dword:00880001

[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\InprocServ
er32]
@="ds863x.dll"

[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t Video]
;"Direction"=dword:00000000
;"IsRendered"=dword:00000001
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t Audio]
;"Direction"=dword:00000000
;"IsRendered"=dword:00000001
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

; MEDIATYPE_Video\MEDIASUBTYPE_MPEG1Payload
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{e436eb81-524f-11ce-9f53-0
020af0ba770}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_MPEG2_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{e06d8026-db46-11cf-b4d1-0  
0805f6cbbea}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_SDMPEG4_Video  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{8AB4D3D1-C812-11d5-BE17  
-00A0C90AA8A1}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_U_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{58564944-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_50_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{30355844-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_XVID_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{44495658-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_DIV3_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{33564944-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_MP43_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{3334504D-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_DivX_H264_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{34363248-0000-0010-8000-  
00aa00389b71}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_SMP863x_Capture_Video  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{c6271d47-809e-4b79-a2a9-f  
787e8266760}]
```

```
; Windows Media Video 9  
; FOURCC: 'WMV3' Subtype: 33564D57-0000-0010-8000-00AA00389B71  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{33564D57-0000-0010-8000-  
00AA00389B71}]
```

```
; Windows Media Video 9 Advanced Profile  
; FOURCC: 'WMVA' Subtype: 41564D57-0000-0010-8000-00AA00389B71  
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu  
t  
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{41564D57-0000-0010-8000-  
00AA00389B71}]
```



```
; Windows Media Video Advanced Profile with no Start Codes
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Video\Types\{73646976-0000-0010-8000-00AA00389B71}\{31435657-0000-0010-8000-
00AA00389B71}]

;MEDIATYPE_Audio\MEDIASUBTYPE_MPEG1Payload
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{e436eb81-524f-11ce-9f53-0
020af0ba770}]

;MEDIATYPE_Audio\MEDIASUBTYPE_DOLBY_AC3
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{e06d802c-db46-11cf-b4d1-0
0805f6cbbea}]

;MEDIATYPE_Audio\MEDIASUBTYPE_AudioAC3
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00002000-0000-0010-8000-
00AA00389B71}]

;MEDIATYPE_Audio\MEDIASUBTYPE_MPEG2_AUDIO
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{e06d802b-db46-11cf-b4d1-0
0805f6cbbea}]

;MEDIATYPE_Audio\MEDIASUBTYPE_MPEG1AudioPayload
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000050-0000-0010-8000-
00AA00389B71}]

;MEDIATYPE_Audio\MEDIASUBTYPE_SDMPEG4_PCMAudio
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{8AB4D3D2-C812-11d5-BE17
-00A0C90AA8A1}]

;MEDIATYPE_Audio\MEDIASUBTYPE_PCM
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000001-0000-0010-8000-
00AA00389B71}]

;MEDIATYPE_Audio\MEDIASUBTYPE_DivX_MP3
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000055-0000-0010-8000-
00AA00389B71}]

;MEDIATYPE_Audio\MEDIASUBTYPE_ADTS_ACAUDIO
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{341F3A28-4476-4277-ADFF-
ADEADE3D1716}]

; Windows Media Audio 9 and previous compatible versions
; Format tag: 0x161 Subtype: 00000161-0000-0010-8000-00AA00389B71
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Inpu
t
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000161-0000-0010-8000-
00AA00389B71}]
```

SIGMA DESIGNS®

```
; Windows Media Audio 9 Professional
; Format tag: 0x162 Subtype: 00000162-0000-0010-8000-00AA00389B71
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Input
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000162-0000-0010-8000-
00AA00389B71}]

; Windows Media Audio 9 Lossless
; Format tag: 0x163 Subtype: 00000163-0000-0010-8000-00AA00389B71
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Input
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000163-0000-0010-8000-
00AA00389B71}]

; Windows Media Audio 9 Voice
; Format tag: 0x00A Subtype: 0000000A-0000-0010-8000-00AA00389B71
[HKEY_CLASSES_ROOT\CLSID\{caa414f0-c7c3-11d4-a914-0080ad91bd94}\Pins\Input
Audio\Types\{73647561-0000-0010-8000-00AA00389B71}\{0000000A-0000-0010-8000-
00AA00389B71}]

;; WMA Pro
[HKEY_CLASSES_ROOT\Filter\{04a43571-24c5-474c-9504-7a315db89850}]
IF INITDSHOWFILTERNAMES
@="Sigma Designs WMAPro Decoder"
ENDIF

[HKEY_CLASSES_ROOT\CLSID\{04a43571-24c5-474c-9504-7a315db89850}]
IF INITNODSHOWFILTERCLASSNAMES !
@="Sigma Designs WMAPro Decoder"
ENDIF
"Merit"=dword:00400000

[HKEY_CLASSES_ROOT\CLSID\{04a43571-24c5-474c-9504-7a315db89850}\InprocServer32]
@="dswmapro.dll"

; Sigma Designs 8634 wave built-in driver
;
; Supports only wave out
[HKEY_LOCAL_MACHINE\Drivers\Builtin\Audio]
"Prefix"="WAV"
"Dll"="wave863x.dll"
"Index"=dword:1
"Order"=dword:1

////////////////////////////////////
; @CESYSGEN IF QUARTZ_IMAGE
; SMP863x Capture Source Filter
////////////////////////////////////
[HKEY_CLASSES_ROOT\Filter\{9a036243-33fa-45c5-9d9d-a7d07c2fc670}]
IF INITDSHOWFILTERNAMES
@="SMP86xx Capture Source Filter"
ENDIF
[HKEY_CLASSES_ROOT\CLSID\{9a036243-33fa-45c5-9d9d-a7d07c2fc670}]
IF INITNODSHOWFILTERCLASSNAMES !
@="SMP86xx Capture Source Filter"
ENDIF
"Merit"=dword:00600000
[HKEY_CLASSES_ROOT\CLSID\{9a036243-33fa-45c5-9d9d-a7d07c2fc670}\InprocServer32]
@="capsrc863x.dll"

[HKEY_CLASSES_ROOT\CLSID\{9a036243-33fa-45c5-9d9d-a7d07c2fc670}\Pins\Video
Output]
"Direction"=dword:00000001
;"IsRendered"=dword:00000000
"AllowedZero"=dword:00000001
```



```
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{674A6F9F-E8D4-436f-9498-462CBC7F0DDD}\Pins\Video
Output\Types\{73646976-0000-0010-8000-00AA00389B71}\{e06d8026-db46-11cf-b4d1-
00805f6cbbca}]

[HKEY_CLASSES_ROOT\CLSID\{674A6F9F-E8D4-436f-9498-462CBC7F0DDD}\Pins\Audio
Output]
"Direction"=dword:00000001
;"IsRendered"=dword:00000000
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{674A6F9F-E8D4-436f-9498-462CBC7F0DDD}\Pins\Audio
Output\Types\{73647561-0000-0010-8000-00AA00389B71}\{00000050-0000-0010-8000
-00AA00389B71}]

;;;;;;;;; hwdemux863x.dll ;;;;;;;;;;
; @CESYSGEN IF QUARTZ_IMAGE
; Sigma Designs HW Demux Filter
;;;;;;;;;
[HKEY_CLASSES_ROOT\Filter\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}]
IF INITDSHOWFILTERNAMES
@="Sigma Designs HW TS Demux"
ENDIF

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}]
IF INITNODSHOWFILTERCLASSNAMES !
@="Sigma Designs HW TS Demux"
ENDIF
"Merit"=dword:00200000 ; Merit is set to DO_NOT_USE - your application must
manually insert this filter

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\InprocServer32]
@="hwdemux863x.dll"

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\Input
]
"Direction"=dword:00000000
;"IsRendered"=dword:00000000
"AllowedZero"=dword:00000000
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\OutputVPayload]
"Direction"=dword:00000001
;"IsRendered"=dword:00000000
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\OutputAPayload]
"Direction"=dword:00000001
;"IsRendered"=dword:00000000
"AllowedZero"=dword:00000001
;"AllowedMany"=dword:00000000

[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\Input\Types\{e436eb83-524f-11ce-9f53-0020af0ba770}\{00000000-0000-0000-0000-00000000
0000}]

;MEDIATYPE_Audio\MEDIASUBTYPE_DOLBY_AC3
```

SIGMA DESIGNS®

```
[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\OutputAPayload\Types\{73647561-0000-0010-8000-00AA00389B71}\{e06d802c-db46-11cf-b4d1-00805f6cbbea}]
```

```
;MEDIATYPE_Video\MEDIASUBTYPE_MPEG2_VIDEO  
[HKEY_CLASSES_ROOT\CLSID\{bfc6c826-4b93-4a66-8f58-ed0b7047311c}\Pins\OutputVPayload\Types\{73646976-0000-0010-8000-00AA00389B71}\{e06d8026-db46-11cf-b4d1-00805f6cbbea}]
```

Find the pflash device registry line.

```
#include "$(_TARGETPLATROOT)\src\drivers\pflash\pflash\pflash.reg"  
#include "$(_TARGETPLATROOT)\src\drivers\pflash\pflash2\pflash2.reg"
```

Disable it as follow sample.

```
; #include "$(_TARGETPLATROOT)\src\drivers\pflash\pflash\pflash.reg"  
; #include "$(_TARGETPLATROOT)\src\drivers\pflash\pflash2\pflash2.reg"
```

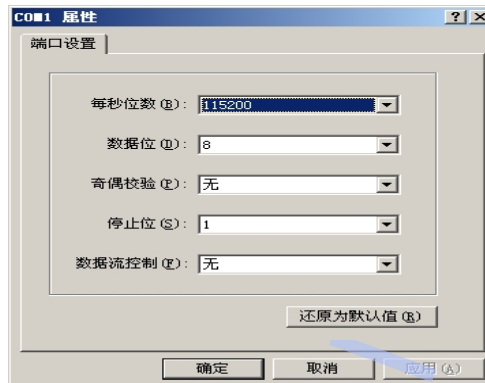
Close and save the platform.reg file.

9. Run "Build OS----Make Run-Time Image" menu command to make a new NK.BIN file.

SIGMA DESIGNS®

C) Download the NK.BIN to SMP8634 board

Link all SMP8634 Ethernet ports and platform builder PC to a network with Ethernet cable, link the SMP8634 Serial debug card to PC serial COM1 port with a serial cable, set the COM1 port parameter as follow sample(pic28).

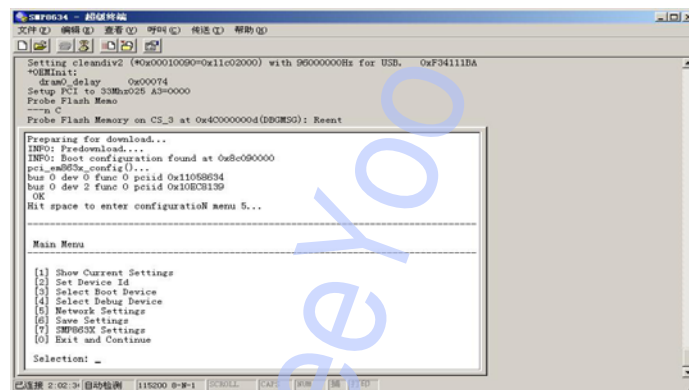


<pic27>

boot up the SMP8634 board.

1. Set the CE bootloader

When the Serial Output window (terminal application) of PC will show the CE loader's bootup information, press any key and get the Main Menu list.



<pic28>

a. Use "[5] Network Settings" to set the bootloader IP address



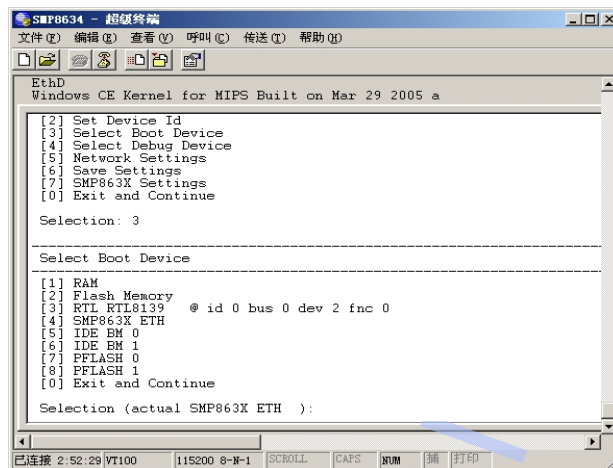
<pic29>

The IP address must be same network IP mask with your PC IP. Example,

PC IP setting,
IP: 192.168.1.51
MASK: 255.255.255.0

Bootload IP setting,
IP: 192.168.1.54
MASK: 255.255.255.0

b. Use "[3] Select Boot Device" to set "RTL RTL8139 ..." to be boot device.



<pic30>

c. Use "[4] Select Debug Device" to set "RTL RTL8139 ..." to be debug device.

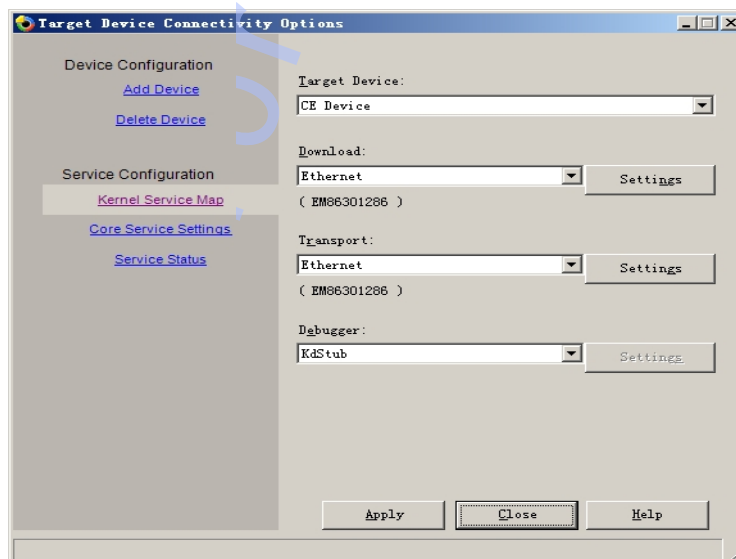


<pic31>

d. Use "[6] Save Settings" to save all settings, Use "[0] Exit and Continue" to continue boot up SMP8634 board

2. Set the platform builder

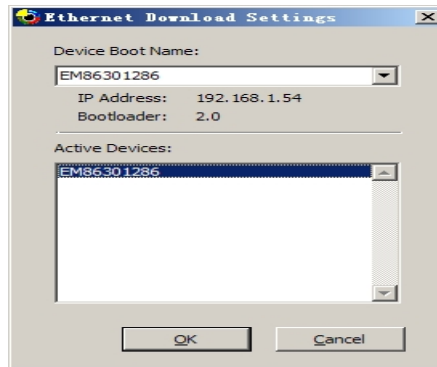
Use "Target---Connectivity Options..." menu command open the "Target Device Connectivity Options" dialog,



<pic32>

SIGMA DESIGNS®

Select "Download" and "Transport" combobox to be "Ethernet"
Select "Debugger" combobox to "KdStub"
Click first "Setting" button to open "Ethernet Download Settings" dialog,



<pic33>

Select the active device (You can see a name as "EM8630*****" and the IP address that you set for SMP8634) to be boot device.

Apply and close all setting dialog,

3. Use "Target----Attach Device" menu command to download your NK.bin to the SMP8634 board.

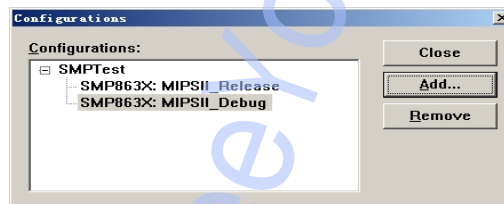
D) Make the SHIP mode SMP8634 CE NK.bin and boot it up without platform builder

If you want to run the CE system specialty, please use the ship mode to build the NK.bin.

1. Make a SHIP mode building

a. Use "Build OS----Configurations..." Menu command to open the

"Configurations" dialog



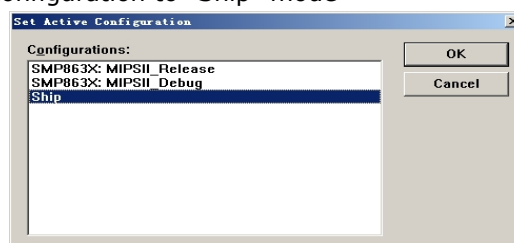
<pic34>

b. Click "Add..." button to open "Add platform Configuration" dialog, select "Copy settings from:" combobox to "SMP863X:MIPSII_Release", type "Ship" in "Configuration:" edit box, click "OK" button to close "Add platform Configuration" dialog, click "Close" button to close "Configurations" dialog.



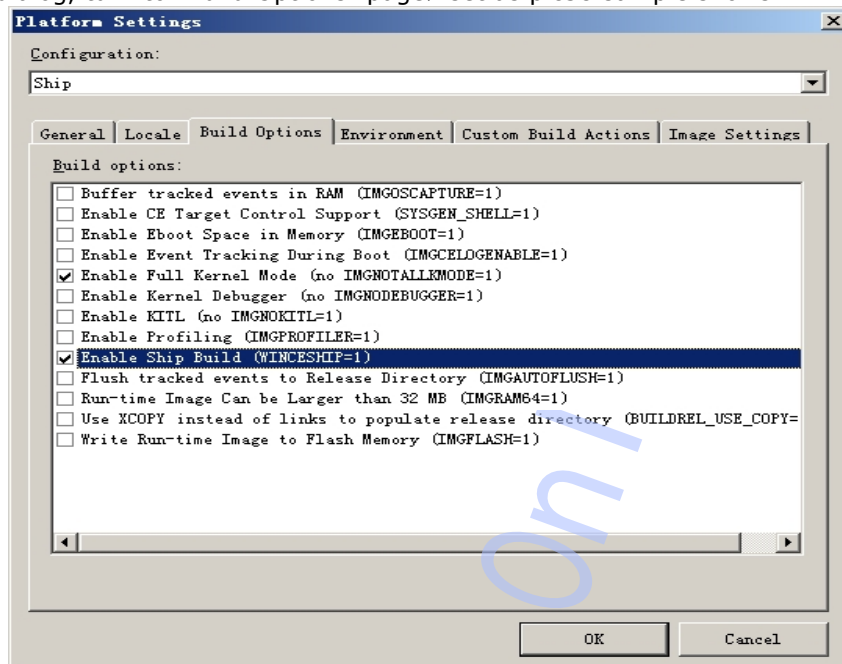
<pic35>

c. Use "Build OS---- Set Active Configuration..." menu command to open "Set Active Configuration" dialog, select "Ship" in "Configurations:" list, click "OK" button to close dialog and set current configuration to "Ship" mode



<pic36>

d. Use "Platform----Settings...." Menu command to open the "Platform Settings" dialog, turn to "Build Options" page, set as pic38 sample shows.



<pic37>

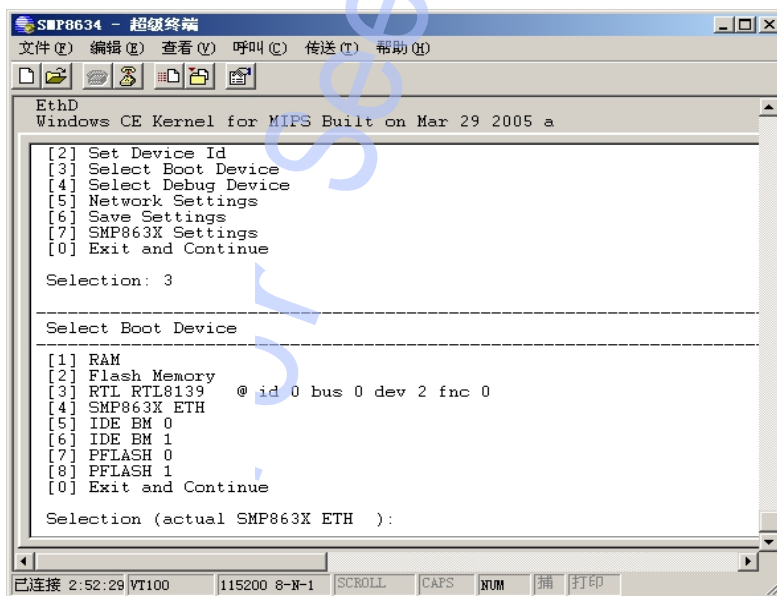
e. Copy all multimedia dll files to "\\WINCE500\\PBWorkspace\\(your CE project name)\\RelDir\\Ship" CE project directory

g. Modify config.bib, platform.bib, platform.reg file and make the nk.bin file

h. Copy the nk.bin to the first FAT32 partition of HDD device or pflash2 storage device

2. Boot up Nk.bin from primary master HDD

In Main Menu of CE boot loader in Serial Output window(terminal application) of PC, Use "[3] Select Boot Device" to set "[5]IDE BM 0" to be boot device.



<pic38>

CE boot loader can loader NK.bin from the first FAT32 partition of HardDisk only.

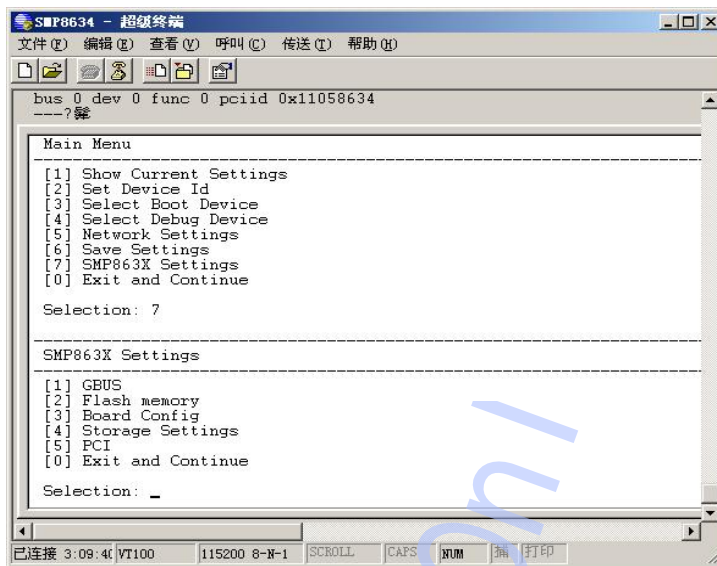
3. Boot up Nk.bin from Pflash

a. If you have copy the nk.bin file to pflash2 storage device

In Main Menu of CE boot loader, use "[3] Select Boot Device" to set "[5]PFLASH 1" to be boot device.

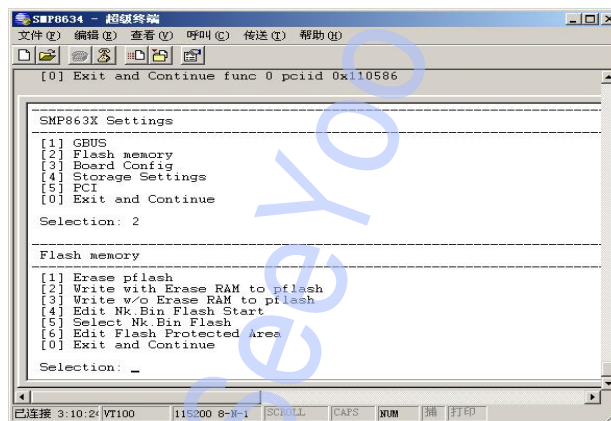
b. If you have write the nk.bin file to the address of pflash, for example, I write the nk.bin to boot flash(PFLASH 0) from 0x400000 address

In Main Menu of CE boot loader, Select "[7] SMP863X Setting" get the SMP8634 Settings menu



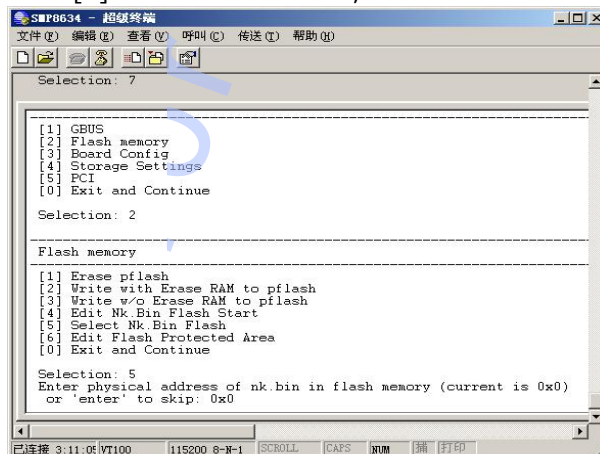
<pic39>

Select "[2]Flash memory", get the Flash memory menu



<pic40>

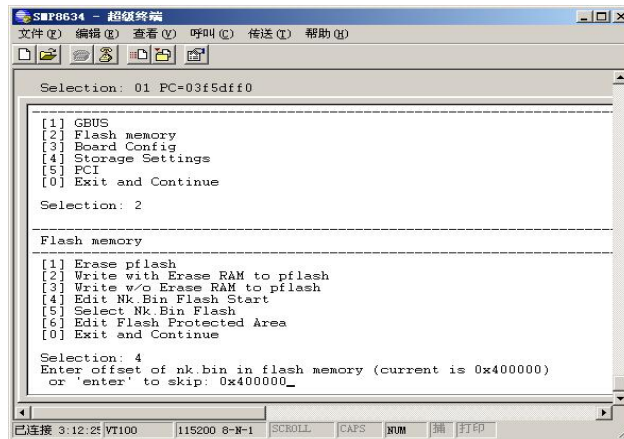
Select "[5]Select NK.Bin Flash", set flash index to 0x0



<pic41>

Select "[4]Edit NK.Bin Flash Start", set NK.bin start flash address to

0x400000



<pic42>

In Main Menu of CE boot loader, use "[3] Select Boot Device" to set "[2]Flash Memory" to be boot device.

c. Use "[6] Save Settings" to save all settings, Use "[0] Exit and Continue" to continue boot up SMP8634 board

4. Write the NK.bin file to Boot flash(PFLASH 0)

Follow sample code shows how to write a file to PFLASH, **it's so dangerous for your boot flash, so please use it carefully.**

The sample function is based on Setxenv demo, all cfi command is defined in Setxenv demo.

```
#define BOOTADDRESS           0x0
#define LOGOADDRESS          0x200000
#define CENKADDRESS          0x400000

static int g_bWriteOK=1;
static RUint8 p_data[128*1024];
static RUint8 cfi_data[0x100];
static int Flash8bit = 0;

////////////////////////////////////
///
RMint32 cfi_query_first_sector_length (void)
{
    RMuint32 n1, l1;

    n1 = cfi_data[0x5a]+1;
    l1 =
        ((RMuint16)cfi_data[0x5c] << 0) |
        ((RMuint16)cfi_data[0x5e] << 8) |
        ((RMuint16)cfi_data[0x60] << 16);
    l1 &= ~1;

    return (RMint32)l1;
}

RMint32 cfi_query_parameters (RMuint32 FlashBaseAddress)
{
    RETAILMSG (1, (TEXT("cfi_query_parameters Begin !\n")));
    memset (cfi_data, 0, sizeof (cfi_data));
    // wait a small bit
    Sleep (100);

    // query command
    *((volatile RUint8 *) (0xaa + FlashBaseAddress)) = 0x98;
    cfi_data[0x20] = *((volatile RUint8 *) (0x20 + FlashBaseAddress));
    cfi_data[0x22] = *((volatile RUint8 *) (0x22 + FlashBaseAddress));
    cfi_data[0x24] = *((volatile RUint8 *) (0x24 + FlashBaseAddress));

    if (cfi_data[0x20] == 'Q' && cfi_data[0x22] == 'R' && cfi_data[0x24] == 'Y')
```

```
{
    RETAILMSG (1, (TEXT("It is a 8 bit flash\n")));
    Flash8bit = 1;
}
else
{
    // try 16 bit access
    // query command
    *((volatile RMuint16 *) (0xaa + FlashBaseAddress)) = 0x98;
    cfi_data[0x20] = *((volatile RMuint8 *) (0x20 + FlashBaseAddress));
    cfi_data[0x22] = *((volatile RMuint8 *) (0x22 + FlashBaseAddress));
    cfi_data[0x24] = *((volatile RMuint8 *) (0x24 + FlashBaseAddress));
    Flash8bit = 0;
    RETAILMSG (1, (TEXT("It is a 16 bit flash\n")));
}
if ((cfi_data[0x20] != 'Q') || (cfi_data[0x22] != 'R') || (cfi_data[0x24] != 'Y'))
{
    // no flash found?
    return 0;
}

cfi_data[0x4e] = *((volatile RMuint8 *) (0x4e + FlashBaseAddress));
RETAILMSG(1, (TEXT("device size: %d\n"), 1 << cfi_data[0x4e]));
cfi_data[0x58] = *((volatile RMuint8 *) (0x58 + FlashBaseAddress));
RETAILMSG(1, (TEXT("number of regions: %d\n"), cfi_data[0x58]));

cfi_data[0x5a] = *((volatile RMuint8 *) (0x5a + FlashBaseAddress));
cfi_data[0x5c] = *((volatile RMuint8 *) (0x5c + FlashBaseAddress));
cfi_data[0x5e] = *((volatile RMuint8 *) (0x5e + FlashBaseAddress));
cfi_data[0x60] = *((volatile RMuint8 *) (0x60 + FlashBaseAddress));
if((RMuint16)cfi_data[0x5c] == 0x1)
{
    RETAILMSG(1, (TEXT("region 1: %d * 0x%04x\n"), (cfi_data[0x5a]+1) *
((RMuint16)cfi_data[0x5c] + 1), ((RMuint16)cfi_data[0x5e] << 8) |
((RMuint16)cfi_data[0x60] << 16)));
}
else
{
    RETAILMSG(1, (TEXT("region 1: %d *
0x%04x\n"), cfi_data[0x5a]+1, ((RMuint16)cfi_data[0x5c] << 0) |
((RMuint16)cfi_data[0x5e] << 8) | ((RMuint16)cfi_data[0x60] << 16)));
}

cfi_data[0x62] = *((volatile RMuint8 *) (0x62 + FlashBaseAddress));
cfi_data[0x64] = *((volatile RMuint8 *) (0x64 + FlashBaseAddress));
cfi_data[0x66] = *((volatile RMuint8 *) (0x66 + FlashBaseAddress));
cfi_data[0x68] = *((volatile RMuint8 *) (0x68 + FlashBaseAddress));
RETAILMSG(1, (TEXT("region 2: %d *
0x%04x\n"), cfi_data[0x62]+1, ((RMuint16)cfi_data[0x64] << 0) |
((RMuint16)cfi_data[0x66] << 8) | ((RMuint16)cfi_data[0x68] << 16)));

cfi_data[0x6a] = *((volatile RMuint8 *) (0x6a + FlashBaseAddress));
cfi_data[0x6c] = *((volatile RMuint8 *) (0x6c + FlashBaseAddress));
cfi_data[0x6e] = *((volatile RMuint8 *) (0x6e + FlashBaseAddress));
cfi_data[0x70] = *((volatile RMuint8 *) (0x70 + FlashBaseAddress));
RETAILMSG(1, (TEXT("region 3: %d *
0x%04x\n"), cfi_data[0x6a]+1, ((RMuint16)cfi_data[0x6c] << 16) |
((RMuint16)cfi_data[0x6e] << 8) | ((RMuint16)cfi_data[0x70] << 0)));

cfi_data[0x72] = *((volatile RMuint8 *) (0x72 + FlashBaseAddress));
cfi_data[0x74] = *((volatile RMuint8 *) (0x74 + FlashBaseAddress));
cfi_data[0x76] = *((volatile RMuint8 *) (0x76 + FlashBaseAddress));
cfi_data[0x78] = *((volatile RMuint8 *) (0x78 + FlashBaseAddress));
RETAILMSG(1, (TEXT("region 4: %d * 0x%04x\n"),
cfi_data[0x72]+1, ((RMuint16)cfi_data[0x74] << 0) | ((RMuint16)cfi_data[0x76] << 8) |
((RMuint16)cfi_data[0x78] << 16)));

// reset command
```

```
if (Flash8bit)
{
    *((volatile RUint8 *)FlashBaseAddress) = 0xf0;
}
else
{
    *((volatile RUint16 *)FlashBaseAddress) = 0xf0;
}

Sleep (1000);
RETAILMSG (1, (TEXT("cfi_query_parameters End !\n")));
return ((RMint32)1 << cfi_data[0x4e]);
}

void cfi_erase (RUint32 FlashBaseAddress, RUint32 addr, RMint32 len)
{
    RETAILMSG(1, (TEXT("cfi_erase Begin !\n")));
    RUint8 tmp0, tmp1;
    RUint32 i, s;
    RUint32 n1, n2, n3, n4;
    RUint32 l1, l2, l3, l4;
    RUint32 start1, start2, start3, start4;

    g_bWriteOK=0;

    if((RUint16)cfi_data[0x5c] == 0x1)
    {
        n1 = (cfi_data[0x5a]+1) * ((RUint16)cfi_data[0x5c] + 1);
    }
    else
    {
        n1 = cfi_data[0x5a]+1;
    }
    l1 =
        ((RUint16)cfi_data[0x5c] << 0) |
        ((RUint16)cfi_data[0x5e] << 8) |
        ((RUint16)cfi_data[0x60] << 16);
    l1 &= ~1;
    start1 = 0;

    n2 = cfi_data[0x62]+1;
    l2 =
        ((RUint16)cfi_data[0x64] << 0) |
        ((RUint16)cfi_data[0x66] << 8) |
        ((RUint16)cfi_data[0x68] << 16);
    l2 &= ~1;
    start2 = start1 + n1 * l1;

    n3 = cfi_data[0x6a]+1;
    l3 =
        ((RUint16)cfi_data[0x6c] << 16) |
        ((RUint16)cfi_data[0x6e] << 8) |
        ((RUint16)cfi_data[0x70] << 0);
    l3 &= ~1;
    start3 = start2 + n2 * l2;

    n4 = cfi_data[0x72]+1;
    l4 =
        ((RUint16)cfi_data[0x74] << 0) |
        ((RUint16)cfi_data[0x76] << 8) |
        ((RUint16)cfi_data[0x78] << 16);
    l4 &= ~1;
    start4 = start3 + n3 * l3;

    if ((addr >= start1) && (addr < start2) && l1)
    {
        s = start1;
        if(addr > s)
```

```
{
    s=addr;
}
// region 1
for (i=0; i<n1; i++)
{
    if (Flash8bit)
    {
        *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
        *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
        *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0x80;
        *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
        *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
        *((volatile RUint8 *) (s + FlashBaseAddress)) = 0x30;
    }
    else
    {
        *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
        *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
        *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0x80;
        *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
        *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
        *((volatile RUint16 *) (s + FlashBaseAddress)) = 0x30;
    }
    // wait for the erase to complete
    while (1)
    {
        tmp0 = *((volatile RUint8 *) (s + FlashBaseAddress));
        tmp1 = *((volatile RUint8 *) (s + FlashBaseAddress));
        if ((tmp0 & 0x40) == (tmp1 & 0x40))
        {
            if (tmp0 & 0x80)
                break;
        }
    }
    s += l1;
    addr = s;
    len -= l1;
    if (len <= 0)
        break;
}
}

RETAILMSG(1, (TEXT("region1 done.\n")));
if (len <= 0)
    return;

if ((addr >= start2) && (addr < start3) && l2)
{
    s = start2;
    // region 2
    for (i=0; i<n2; i++)
    {
        if (Flash8bit)
        {
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (s + FlashBaseAddress)) = 0x30;
        }
        else
        {
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
        }
    }
}
```

```
        *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
        *((volatile RUint16 *) (s + FlashBaseAddress)) = 0x30;
    }
    // wait for the erase to complete
    while (1)
    {
        tmp0 = *((volatile RUint8 *) (s + FlashBaseAddress));
        tmp1 = *((volatile RUint8 *) (s + FlashBaseAddress));
        if ((tmp0 & 0x40) == (tmp1 & 0x40))
        {
            if (tmp0 & 0x80)
                break;
        }
    }
    s += 12;
    addr = s;
    len -= 12;
    if (len <= 0)
        break;
}

RETAILMSG(1, (TEXT("region2 done.\n")));
if (len <= 0)
    return;

if ((addr >= start3) && (addr < start4) && !3)
{
    s = start3;
    // region 3
    for (i=0; i<n3; i++)
    {
        if (Flash8bit)
        {
            *((volatile RUint8 *) (0xaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (0xaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint8 *) (0xaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (s + FlashBaseAddress)) = 0x30;
        }
        else
        {
            *((volatile RUint16 *) (0xaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
            *((volatile RUint16 *) (0xaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint16 *) (0xaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
            *((volatile RUint16 *) (s + FlashBaseAddress)) = 0x30;
        }
        // wait for the erase to complete
        while (1)
        {
            tmp0 = *((volatile RUint8 *) (s + FlashBaseAddress));
            tmp1 = *((volatile RUint8 *) (s + FlashBaseAddress));
            if ((tmp0 & 0x40) == (tmp1 & 0x40))
            {
                if (tmp0 & 0x80)
                    break;
            }
        }
        s += 13;
        addr = s;
        len -= 13;
        if (len <= 0)
            break;
    }
}
```

```
RETAILMSG(1, (TEXT("region3 done.\n")));
if (len <= 0)
    return;

if ((addr >= start4) && (l4))
{
    s = start4;
    // region 4
    for (i=0; i<n4; i++)
    {
        if (Flash8bit)
        {
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint8 *) (0x555 + FlashBaseAddress)) = 0x55;
            *((volatile RUint8 *) (s + FlashBaseAddress)) = 0x30;
        }
        else
        {
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0x80;
            *((volatile RUint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
            *((volatile RUint16 *) (0x554 + FlashBaseAddress)) = 0x55;
            *((volatile RUint16 *) (s + FlashBaseAddress)) = 0x30;
        }
        // wait for the erase to complete
        while (1)
        {
            tmp0 = *((volatile RUint8 *) (s + FlashBaseAddress));
            tmp1 = *((volatile RUint8 *) (s + FlashBaseAddress));
            if ((tmp0 & 0x40) == (tmp1 & 0x40))
            {
                if (tmp0 & 0x80)
                    break;
            }
        }
        s += l4;
        len -= l4;
        if (len <= 0)
            break;
    }
}
RETAILMSG(1, (TEXT("region4 done.\n")));

// reset command
if (Flash8bit)
{
    *((volatile RUint8 *) (FlashBaseAddress)) = 0xf0;
}
else
{
    *((volatile RUint16 *) (FlashBaseAddress)) = 0xf0;
}
RETAILMSG(1, (TEXT("cfi_erase End !\n")));
}

void cfi_write (RUint32 FlashBaseAddress, RUint32 addr, RUint32 length, RUint8
*flashbuf)
{
    RETAILMSG(1, (TEXT("cfi_write Begin !\n")));
    RUint32 i;
    RUint8 *p = flashbuf;

    if (Flash8bit)
    {
```



```
// unlock bypass
*((volatile RMuint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
Sleep (100);
*((volatile RMuint8 *) (0x555 + FlashBaseAddress)) = 0x55;
Sleep (100);
*((volatile RMuint8 *) (0xaaa + FlashBaseAddress)) = 0x20;
Sleep (100);

// unlock bypass
*((volatile RMuint8 *) (0xaaa + FlashBaseAddress)) = 0xaa;
Sleep (100);
*((volatile RMuint8 *) (0x555 + FlashBaseAddress)) = 0x55;
Sleep (100);
*((volatile RMuint8 *) (0xaaa + FlashBaseAddress)) = 0x20;
Sleep (100);
}
else
{
// unlock bypass
*((volatile RMuint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
Sleep (100);
*((volatile RMuint16 *) (0x554 + FlashBaseAddress)) = 0x55;
Sleep (100);
*((volatile RMuint16 *) (0xaaa + FlashBaseAddress)) = 0x20;
Sleep (100);

// unlock bypass
*((volatile RMuint16 *) (0xaaa + FlashBaseAddress)) = 0xaa;
Sleep (100);
*((volatile RMuint16 *) (0x554 + FlashBaseAddress)) = 0x55;
Sleep (100);
*((volatile RMuint16 *) (0xaaa + FlashBaseAddress)) = 0x20;
Sleep (100);
}
for (i=addr; i<(addr + length); i++)
{
if ((i % 0x10000) == 0)
{
RETAILMSG (1, (TEXT("writing address 0x%08lx\r\n"), i));
fflush (stdout);
}
// unlock bypass program
if (Flash8bit)
{
*((volatile RMuint8 *) (FlashBaseAddress)) = 0xa0;
*((volatile RMuint8 *) (i + FlashBaseAddress)) = *p;
while (*((volatile RMuint8 *) (i + FlashBaseAddress)) != *p);
p++;
}
else
{
RMuint16 data = ((RMuint16)p[0] | (((RMuint16)p[1] << 8);
*((volatile RMuint16 *) (FlashBaseAddress)) = 0xa0;
*((volatile RMuint16 *) (i + FlashBaseAddress)) = data;
while (*((volatile RMuint16 *) (i + FlashBaseAddress)) != data);
p++;
p++;
i++;
}
}
// reset command
if (Flash8bit)
{
*((volatile RMuint8 *) (FlashBaseAddress)) = 0xf0;
}
else
```

```
{
    *((volatile RUint16 *)FlashBaseAddress) = 0xf0;
}

Sleep (1000);
g_bWriteOK=1;
RETAILMSG(1, (TEXT("cfi_write End !\n")));
}

////////////////////////////////////
#define HOST_INTERFACE_BASE_ADDRESS ((RUint32)OALPtoUA(0x20000))
#define PBI_BASE_ADDRESS (HOST_INTERFACE_BASE_ADDRESS + 0x800)
#define PBI_TIMING0 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x00)
#define PBI_TIMING1 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x04)
#define PBI_TIMING2 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x08)
#define PBI_TIMING3 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x0C)
#define PBI_TIMING4 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x10)
#define PBI_TIMING5 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x14)
#define PBI_DEFAULT_TIMING ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x18)
#define PBI_USE_TIMING0 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x1C)
#define PBI_USE_TIMING1 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x20)
#define PBI_USE_TIMING2 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x24)
#define PBI_USE_TIMING3 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x28)
#define PBI_USE_TIMING4 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x2C)
#define PBI_USE_TIMING5 ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x30)
#define PBI_CS_CONFIG ((volatile RUint32 *)PBI_BASE_ADDRESS + 0x34)

void setup_pbi (void)
{
    RUint32 pbi_timing0 = *PBI_TIMING0;
    RUint32 pbi_timing1 = *PBI_TIMING1;
    RUint32 pbi_timing2 = *PBI_TIMING2;
    RUint32 pbi_timing3 = *PBI_TIMING3;
    RUint32 pbi_timing4 = *PBI_TIMING4;
    RUint32 pbi_timing5 = *PBI_TIMING5;
    RUint32 pbi_use_timing0 = *PBI_USE_TIMING0;
    RUint32 pbi_use_timing1 = *PBI_USE_TIMING1;
    RUint32 pbi_use_timing2 = *PBI_USE_TIMING2;
    RUint32 pbi_use_timing3 = *PBI_USE_TIMING3;
    RUint32 pbi_use_timing4 = *PBI_USE_TIMING4;
    RUint32 pbi_use_timing5 = *PBI_USE_TIMING5;

    RUint32 pbi_default_timing = *PBI_DEFAULT_TIMING;
    RUint32 pbi_cs_config = *PBI_CS_CONFIG;

    // configure CS2, CS3 to be 16bit, packed, non-ide, Address/Data lines not muxed, use
    IOrdy
    pbi_cs_config &= ~0xCCCCC;
    pbi_cs_config |= 0x000C0;
    *PBI_CS_CONFIG = pbi_cs_config;

    // what is the proper value for flash memory?
    RUint32 Ta, Tb, Tc, Td, timing, sysclk;
    sysclk = 3003; // XXX hard coded to 300MHz

    Ta = sysclk * 0 / 10000 + 1;
    Tb = sysclk * 0 / 10000 + 1;
    Tc = sysclk * 50 / 10000 + 1;
    Td = sysclk * 70 / 10000 + 1;

    // on the safe side?
    Ta = sysclk * 200 / 10000 + 1;
    Tb = sysclk * 200 / 10000 + 1;
    Tc = sysclk * 200 / 10000 + 1;
    Td = sysclk * 200 / 10000 + 1;

    timing = (Ta << 0) | (Tb << 8) | (Tc << 16) | (Td << 24);
    *PBI_DEFAULT_TIMING = timing;
}
```

```
}  
  
void g_WritePFlashInfo2(DWORD nAddress,LPTSTR sFileName,HWND hWnd)  
{  
    RMuint32 FlashBaseAddress;  
    RMuint32 FlashAddress;  
    RMuint32 flashsize;  
    RMuint32 dFilesize=0;  
    char    chTemp[4],chFileName[256];  
    DWORD   nTemp;  
    BOOL    b=FALSE;  
  
    for(nTemp=0;nTemp < (DWORD)strlen(sFileName);nTemp++)  
    {  
        sprintf(chTemp,"%c",sFileName[nTemp]);  
        if(nTemp == 0)  
        {  
            strcpy(chFileName,chTemp);  
        }  
        else  
        {  
            strcat(chFileName,chTemp);  
        }  
    }  
  
    setup_pbi ();  
  
    // allocate 64MB of virtual space to map the flash into  
    volatile RMuint8 *pVirtualAddress = (volatile RMuint8 *)VirtualAlloc (0, 1024*1024*64,  
MEM_RESERVE, PAGE_NOACCESS);  
    if (pVirtualAddress != NULL)  
    {  
        // CS2 @ 0x48000000  
        // use the MMU to map 0x48000000 to this memory  
        // VirtualCopy (in WinCE 5.0?) has a limit of 32MB  
        b = VirtualCopy ((PVOID)pVirtualAddress, (PVOID)(0x48000000 >> 8),  
            1024*1024*32, PAGE_PHYSICAL | PAGE_READWRITE | PAGE_NOCACHE);  
        ASSERT (b == TRUE);  
        if (b == FALSE)  
        {  
            VirtualFree ((PVOID)pVirtualAddress, 0, MEM_RELEASE);  
            return;  
        }  
        // copy the next 32MB  
        b = VirtualCopy ((PVOID)(pVirtualAddress + 1024*1024*32),  
(PVOID)(0x4A000000 >> 8),  
            1024*1024*32, PAGE_PHYSICAL | PAGE_READWRITE | PAGE_NOCACHE);  
        ASSERT (b == TRUE);  
        if (b == FALSE)  
        {  
            VirtualFree ((PVOID)pVirtualAddress, 0, MEM_RELEASE);  
            return;  
        }  
    }  
    else  
    {  
        ASSERT (0);  
        RETAILMSG(1, (TEXT("Can't allocate 64M virtual space!\n")));  
        return;  
    }  
  
    FlashBaseAddress = (RMuint32)pVirtualAddress;  
    RETAILMSG(1, (TEXT("Flash base address is %08lx:\n"),FlashBaseAddress));  
  
    RETAILMSG(1, (TEXT("cfi_query_parameters @ %08lx:\n"),0x48000000));  
    flashsize = cfi_query_parameters (FlashBaseAddress);  
    RETAILMSG(1, (TEXT("Flash size is %08lx:\n"),flashsize));  
}
```

```
switch(nAddress)
{
    case 0:
        FlashAddress=(RMuint32)pVirtualAddress + CENKADDRESS;
        break;
    case 1:
        FlashAddress=(RMuint32)pVirtualAddress + BOOTADDRESS;
        break;
    case 2:
    default:
        RETAILMSG(1, (TEXT("Can't write this address In this version!")));
        VirtualFree ((PVOID)pVirtualAddress, 0, MEM_RELEASE);
        return;
}

RETAILMSG(1, (TEXT("Write Flash Address from @ %08lx:\n"),FlashAddress));

if (flashsize)
{
    FILE *f = fopen(chFileName, "rb");    //Open the bin file
    if(f == NULL)
    {
        RETAILMSG(1, (TEXT("Can't Open the file!\n")));
        VirtualFree ((PVOID)pVirtualAddress, 0, MEM_RELEASE);
        return;
    }
    fseek( f, 0, SEEK_END);    //Seek to end and get the file size
    dFilesize=ftell(f);
    RETAILMSG(1, (TEXT("File size is %d bytes!\n"),dFilesize));
    fseek( f, 0, SEEK_SET);    //Seek to the begin

    RMint32 sectorlength;
    sectorlength = cfi_query_first_sector_length ();
    RETAILMSG(1, (TEXT("Fist regions' sector length is %08lx:\n"),sectorlength));
    //128K for each sector

    dProgress_old=0;
    dProgress=0;
    numread=0;
    readCounts=0;

    while( !feof( f))
    {
        if(g_bWriteOK == 1)
        {
            RETAILMSG(1, (TEXT("Begin read file...!\n")));
            numread = fread( p_data, 1, sectorlength, f );
            if(numread)
            {
                RETAILMSG(1, (TEXT("Read one sector file... OK!\n")));
                //Check the update progress
                readCounts=readCounts + numread;

                dProgress=(readCounts / (dFilesize / 100));
                RETAILMSG(1, (TEXT("Read fiile %d %%\n"),dProgress));

                memcpy( p_data + numread, (RMuint8 *)FlashAddress + numread,
sectorlength - numread);
                RETAILMSG(1, (TEXT("memcpy ok!\n")));

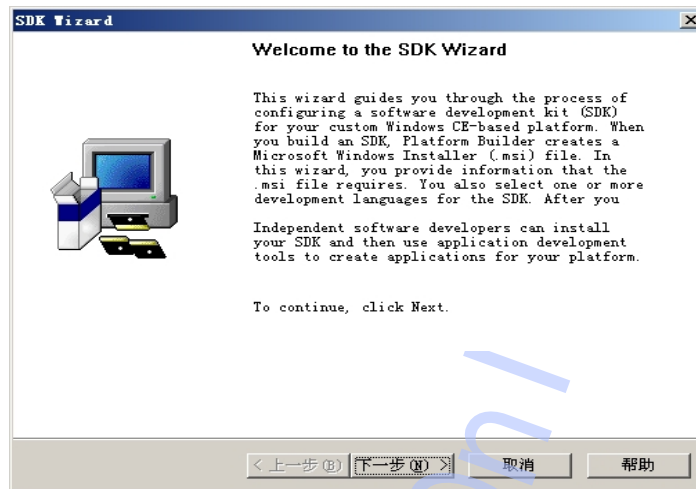
                switch(nAddress)
                {
                    case 0://NK.bin                0x400000
                        RETAILMSG(1, (TEXT("Write NK.bin!\n")));
                        //FlashAddress - (RMuint32)pVirtualAddress
                        cfi_erase (FlashBaseAddress, (FlashAddress -
(RMuint32)pVirtualAddress), sectorlength);
                        RETAILMSG(1, (TEXT("Erase OK!\n")));
                }
            }
        }
    }
}
```


SIGMA DESIGNS®

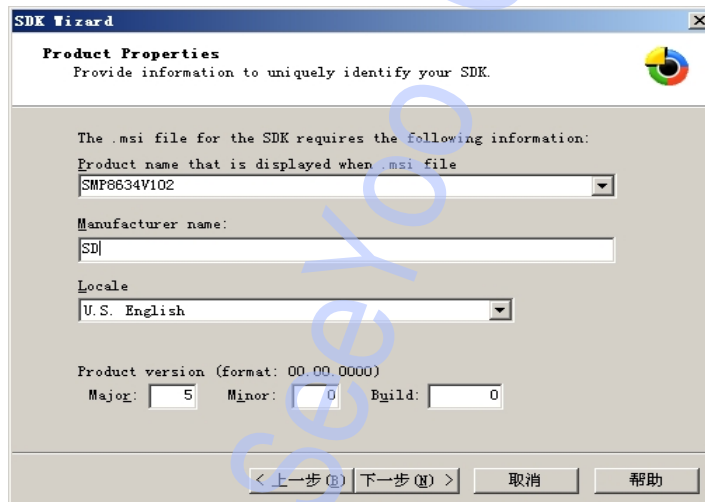
4) How to make application for SMP8634 wince system

A) Made the SDK of current CE project

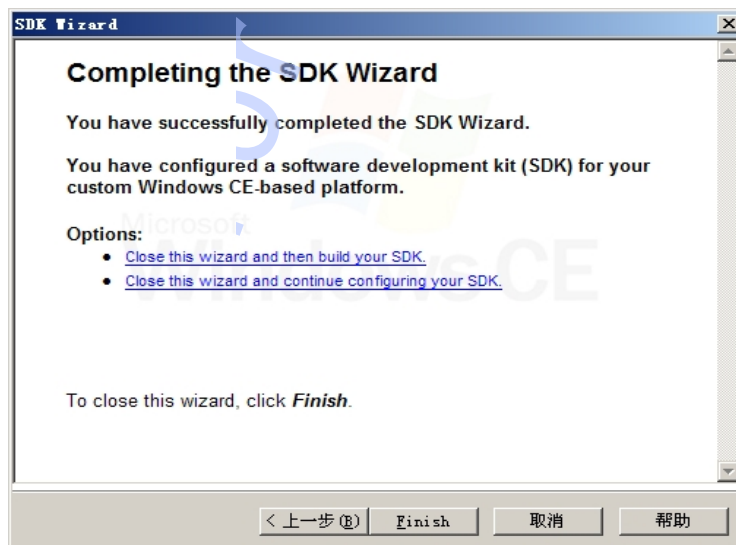
1. Use "Platform----SDK----New SDK..." Menu command to open the "SDK Wizard" dialog, use the wizard make a SDK for current CE project step by step



<pic43>



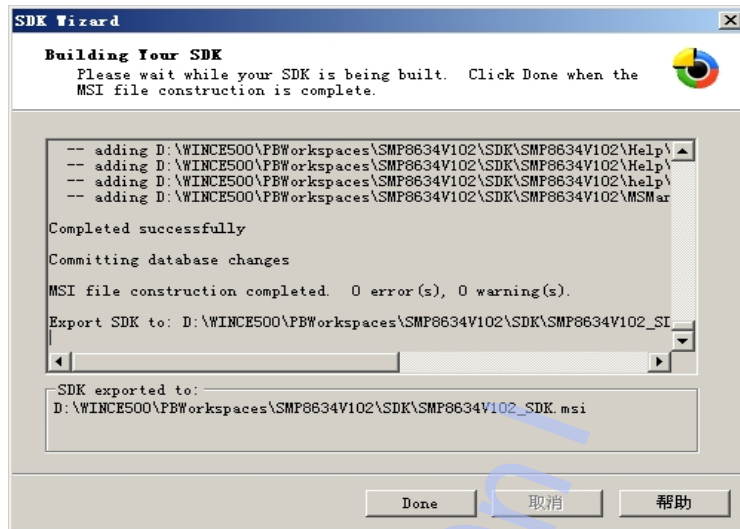
<pic44>



<pic45>

SIGMA DESIGNS®

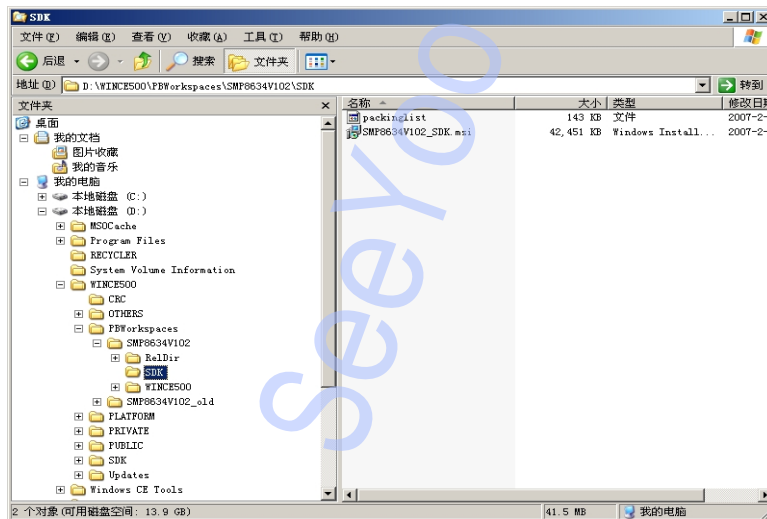
2. Use "Platform----SDK----Build SDK..." Menu command to open the "SDK Wizard" dialog, use the wizard to build the SDK



<pic46>

B) Install the SDK to PC

You can get the SDK in "\\WINCE500\FBWorkspace\your CE project name)\SDK" directory, install it to your PC system.



<pic47>

C) Use EVC++4.0 with SP4 to make application FYI Microsoft MSDN

5) How to test the SMP8634 wince feature

A) SMP8634 ETH

1. Set the IP address for SMP8634 ETH device

Open the **platform.reg** file of CE project, add follow settings before "ENDIF BSP_SMP863X_MAC86XX" line

```
[HKEY_LOCAL_MACHINE\Comm\MAC86XX1\Parms\TcpIp]
"EnableDHCP"=dword:0
;For 060920_KB924785
"AutoTimeout"=dword:3E8
"DefaultGateway"="192.168.1.93"
"LLInterface"=""
"UseZeroBroadcast"=dword:0
"IpAddress"="192.168.1.250"
"Subnetmask"="255.255.255.0"
"DNS"=multi_sz:"202.96.128.166","202.96.134.133"
```

2. SMP8634 ETH device speed

FTP download: 1440 K Bytes/S

HTTP get buffer: 42M Bits/s

3. Set MAC address

- a. For bootloader

The MAC address is defined in Smp863x\Src\Libs\Board_callbacks\sigma_STB.c file, as follow value

```
BYTE gbPermanentMacAddress[6] = { 0x10,0x00,0x00,0x04,0x05,0x06 };
```

Or Build with the void BCLBK_GetMAS_Address(UINT16 mac[3]) function

If you change the value or function, please rebuild bootloader, and update the boot.nb0

- b. For SMP863X MAC device of kernel

Modify the follow function in smp863x_ndis.c, rebuild MAC driver project and kernel

```
NDIS_STATUS SMP863x_Eth_Initialize (
    OUT PNDIS_STATUS OpenErrorStatus,
    OUT PUINT SelectedMediumIndex,
    IN PNDIS_MEDIUM MediumArray,
    IN UINT MediumArraySize,
    IN NDIS_HANDLE MiniportAdapterHandle,
    IN NDIS_HANDLE WrapperConfigurationContext
)
{
    ULONG i;
    NDIS_STATUS Status;
    PSMP863X_ADAPTER Adapter;
    PNDIS_PACKET Packet;
    PNDIS_BUFFER NdisBuffer;
    ULONG VA;
    DWORD hi_mac, low_mac;

    DEBUGMSG (1, (TEXT("SMP863x_Eth_Initialize\r\n")));

    // Search for the medium type (802.3) in the given array.
    // Support 802.3 only
    for (i=0; i<MediumArraySize; i++)
    {
        if (MediumArray[i] == NdisMedium802_3)
        {
            break;
        }
    }
    if (i == MediumArraySize)
    {
        return NDIS_STATUS_UNSUPPORTED_MEDIA;
    }
    *SelectedMediumIndex = i;

    Status = NdisAllocateMemory ((PVOID *)&Adapter,
        sizeof(SMP863X_ADAPTER), 0, HighestAcceptableMax);
    if (Status != NDIS_STATUS_SUCCESS)
    {

```



```
        return (Status);
    }
    NdisZeroMemory (Adapter, sizeof(SMP863X_ADAPTER));
    Adapter->MiniportAdapterHandle = MiniportAdapterHandle;

    NdisMSetAttributesEx (Adapter->MiniportAdapterHandle,
        (NDIS_HANDLE)Adapter,
        0,
        NDIS_ATTRIBUTE_BUS_MASTER |
        NDIS_ATTRIBUTE_DESERIALIZE |
        /*NDIS_ATTRIBUTE_ALWAYS_GIVES_RX_PACKET_OWNERSHIP */
        NDIS_ATTRIBUTE_NO_HALT_ON_SUSPEND,
        0
    );

    InitializeCriticalSection (&Adapter->Lock);

    Adapter->NumMulticastAddressesInUse = -1;

    // disable interrupts
    em86xx_write_reg (EM86XX_IER_REG, 0);

    // allocate memory for rx descriptors
    NdisMAllocateSharedMemory (
        Adapter->MiniportAdapterHandle,
        N_RX_DESC * sizeof (em86xx_desc),
        FALSE,
        &Adapter->pRxDesc,
        &Adapter->PhysAddrOfRxDesc);
    RETAILMSG ((Adapter->pRxDesc == 0), (TEXT("MAC863x:
NdisMAllocateSharedMemory (RxDesc) failed!\r\n")));
    // allocate memory for rx buffers
    NdisMAllocateSharedMemory (
        Adapter->MiniportAdapterHandle,
        N_RX_DESC * RX_BUF_SIZE_PER_DESC,
        FALSE,
        &Adapter->pRxBuffer,
        &Adapter->PhysAddrRxBuffer);
    RETAILMSG ((Adapter->pRxBuffer == 0), (TEXT("MAC863x:
NdisMAllocateSharedMemory (RxBuffer) failed!\r\n")));
    // allocate memory for tx descriptors
    NdisMAllocateSharedMemory (
        Adapter->MiniportAdapterHandle,
        N_TX_DESC * sizeof (em86xx_desc),
        FALSE,
        &Adapter->pTxDesc,
        &Adapter->PhysAddrOfTxDesc);
    RETAILMSG ((Adapter->pTxDesc == 0), (TEXT("MAC863x:
NdisMAllocateSharedMemory (TxDesc) failed!\r\n")));
    // allocate memory for tx buffers
    NdisMAllocateSharedMemory (
        Adapter->MiniportAdapterHandle,
        N_TX_DESC * TX_BUF_SIZE_PER_DESC,
        FALSE,
        &Adapter->pTxBuffer,
        &Adapter->PhysAddrTxBuffer);
    RETAILMSG ((Adapter->pTxBuffer == 0), (TEXT("MAC863x:
NdisMAllocateSharedMemory (TxBuffer) failed!\r\n")));
    // Use NdisGetPhysicalAddressLow to get the lower 32 bits of the physical address

    // allocate memory for rx packets
    NdisAllocatePacketPool (&Status, &Adapter->RxPacketPool, N_RX_DESC,
        PROTOCOL_RESERVED_SIZE_IN_PACKET);
    RETAILMSG ((Status != NDIS_STATUS_SUCCESS), (TEXT("MAC863x:
NdisAllocatePacketPool (RxPacketPool) failed!\r\n")));
    ASSERT (Status == NDIS_STATUS_SUCCESS);
    NdisAllocateBufferPool (&Status, &Adapter->RxBufferPool, N_RX_DESC);
```

```
    RETAILMSG ((Status != NDIS_STATUS_SUCCESS), (TEXT("MAC863x:
NdisAllocateBufferPool (RxBufferPool) failed!\r\n")));
    ASSERT (Status == NDIS_STATUS_SUCCESS);
    VA = (ULONG)Adapter->pRxBuffer;
    for (i=0; i<N_RX_DESC; i++)
    {
        NdisAllocatePacket (&Status, &Packet, Adapter->RxPacketPool);
        RETAILMSG ((Status != NDIS_STATUS_SUCCESS), (TEXT("MAC863x:
NdisAllocatePacket (RxPacketPool) failed!\r\n")));
        ASSERT (Status == NDIS_STATUS_SUCCESS);

        NDIS_SET_PACKET_HEADER_SIZE (Packet, 14);

        NdisAllocateBuffer (&Status, &NdisBuffer, Adapter->RxBufferPool, (PVOID)VA,
RX_BUF_SIZE_PER_DESC);
        RETAILMSG ((Status != NDIS_STATUS_SUCCESS), (TEXT("MAC863x:
NdisAllocateBuffer (RxPacketPool) failed!\r\n")));
        ASSERT (Status == NDIS_STATUS_SUCCESS);
        VA += RX_BUF_SIZE_PER_DESC;

        NdisChainBufferAtFront (Packet, NdisBuffer);

        Adapter->RxPackets[i] = Packet;
    }

    // initialize the descriptors
    em86xx_eth_reset_desc (Adapter);

    hi_mac = em86xx_read_reg (EM86XX_MACAHR_REG);
    low_mac = em86xx_read_reg (EM86XX_MACALR_REG);
    RETAILMSG (1, (TEXT("MAC863X: hi_mac = %08lx, low_mac = %08lx\r\n"),
hi_mac, low_mac));

    if (((hi_mac & 0xffff) == 0xffffffff) ||
        (low_mac == 0xffffffff))
    {
        // here is the preferred method to get a MAC address
        //
        // Sigma Designs ID is 00-16-e8
        // the last 48 bits we get from the serial number of the chip
        // the serial number is found by querying XOS
        //
        // use Sigma MAC address
        // use the rxbuffer to get the serial number
        #define XRPC_ID_GETSERIAL 0
        /*
        XRPC_HEADER_BLOCK *pB = (XRPC_HEADER_BLOCK *)Adapter->pRxBuffer;
        DWORD base_addr = (DWORD)NdisGetPhysicalAddressLow
(Adapter->PhysAddrRxBuffer);
        pB->xrpcid = XRPC_ID_GETSERIAL;
        RETAILMSG (1, (TEXT("base_addr = %d\r\n"), base_addr));    //mike*/

        XRPC_HEADER_BLOCK *pB = (XRPC_HEADER_BLOCK *)Adapter->pRxBuffer;
        DWORD base_addr = 0;
        base_addr = (DWORD)NdisGetPhysicalAddressLow
(Adapter->PhysAddrRxBuffer);

        RETAILMSG (1, (TEXT("base_addr = %d\r\n"), base_addr));    //mike

        pB->xrpcid = XRPC_ID_GETSERIAL;
        pB->param0 = 0;
        pB->param1 = 0;
        pB->param2 = 0;
        pB->param3 = 0;
        pB->param4 = 0;
        pB->headerandblocksize = sizeof(XRPC_HEADER_BLOCK);

        if (do_xrpc (0, base_addr))
        {
```

```
RETAILMSG (1, (TEXT("MAC863X: serial# =
%08lx%08lx%08lx%08lx\r\n"),
    pB->param3,
    pB->param2,
    pB->param1,
    pB->param0));
    Adapter->CurrentAddress[3] = Adapter->PermanentAddress[3] =
(BYTE)(pB->param3 >> 16);
    Adapter->CurrentAddress[4] = Adapter->PermanentAddress[4] =
(BYTE)(pB->param3 >> 8);
    Adapter->CurrentAddress[5] = Adapter->PermanentAddress[5] =
(BYTE)(pB->param3 >> 0);
}
else
{
    RETAILMSG (1, (TEXT("MAC863X: XRPC_ID_GETSERIAL failed!\r\n")));
    Adapter->CurrentAddress[3] = Adapter->PermanentAddress[3] = 0x00;
    Adapter->CurrentAddress[4] = Adapter->PermanentAddress[4] = 0x04;
    Adapter->CurrentAddress[5] = Adapter->PermanentAddress[5] = 0x10;
}
Adapter->CurrentAddress[2] = Adapter->PermanentAddress[2] = 0xe8;
Adapter->CurrentAddress[1] = Adapter->PermanentAddress[1] = 0x16;
Adapter->CurrentAddress[0] = Adapter->PermanentAddress[0] = 0x00;

RETAILMSG (1, (TEXT("MAC863X: Using Sigma MAC address:
%02x-%02x-%02x-%02x-%02x-%02x\r\n"),
    (BYTE)Adapter->CurrentAddress[0],
    (BYTE)Adapter->CurrentAddress[1],
    (BYTE)Adapter->CurrentAddress[2],
    (BYTE)Adapter->CurrentAddress[3],
    (BYTE)Adapter->CurrentAddress[4],
    (BYTE)Adapter->CurrentAddress[5]
));
}
else
{
    RETAILMSG (1, (TEXT("MAC863X: Using pre-programmed MAC address:
%02x-%02x-%02x-%02x-%02x-%02x\r\n"),
    (BYTE)(hi_mac >> 8),
    (BYTE)(hi_mac),
    (BYTE)(low_mac >> 24),
    (BYTE)(low_mac >> 16),
    (BYTE)(low_mac >> 8),
    (BYTE)(low_mac)
));

    // use the one already programmed
    Adapter->CurrentAddress[0] = Adapter->PermanentAddress[0] =
(BYTE)(low_mac);
    Adapter->CurrentAddress[1] = Adapter->PermanentAddress[1] =
(BYTE)(low_mac >> 8);
    Adapter->CurrentAddress[2] = Adapter->PermanentAddress[2] =
(BYTE)(low_mac >> 16);
    Adapter->CurrentAddress[3] = Adapter->PermanentAddress[3] =
(BYTE)(low_mac >> 24);
    Adapter->CurrentAddress[4] = Adapter->PermanentAddress[4] =
(BYTE)(hi_mac);
    Adapter->CurrentAddress[5] = Adapter->PermanentAddress[5] =
(BYTE)(hi_mac >> 8);
}

// initialize nic
em86xx_eth_hw_init (Adapter);

// hook interrupt
Status = NdisMRegisterInterrupt (
    &Adapter->Interrupt,
    Adapter->MiniportAdapterHandle,
```

```
38,  
38,  
TRUE,          // RequestISR  
FALSE,        // SharedInterrupt  
NdisInterruptLevelSensitive  
);  
ASSERT (Status == NDIS_STATUS_SUCCESS);  
  
// enable interrupts  
em86xx_write_reg (EM86XX_IER_REG, ETH_IRQ_FLAGS);  
  
// check link status  
if (phy_is_connected ((void *)Adapter))  
{  
    Adapter->LinkConnection = NDIS_STATUS_MEDIA_CONNECT;  
}  
else  
{  
    Adapter->LinkConnection = NDIS_STATUS_MEDIA_DISCONNECT;  
}  
  
NdisInitializeTimer (&Adapter->Timer, SMP863x_Timer, Adapter);  
NdisSetTimer (&Adapter->Timer, 1000);  
  
#if 0  
Adapter->DebugThread = CreateThread (  
    0,  
    0,  
    (LPTHREAD_START_ROUTINE)EthDebugThread,  
    (LPVOID)Adapter,  
    0,  
    NULL);  
#endif  
  
return NDIS_STATUS_SUCCESS;  
}
```

B) SMP8634 USB

1. Modify registry setting for SMP8634 USB device

Open the **platform.reg** file of CE project, find follow settings

```
; @CESYSGEN IF CE_MODULES_USBD  
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\EHCI86XX]  
"Prefix"="HCD"  
"Dll"="ehci86xx.dll"  
"IsrDll"="giisr.dll"  
"IsrHandler"="ISRHandler"  
"PhysicalMemoryAddress"=dword:23C00000  
"PhysicalMemoryLength"=dword:00400000  
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\OHCI86XX]  
"Prefix"="HCD"  
"Dll"="ohci86xx.dll"  
"Order"=dword:2  
"Class"=dword:0c  
"SubClass"=dword:03  
"ProgIF"=dword:10  
"IsrDll"="giisr.dll"  
"IsrHandler"="ISRHandler"  
"HcdCapability"=dword:5 ;HCD_SUSPEND_ON_REQUEST|HCD_SUSPEND_RESUME  
; @CESYSGEN ENDIF CE_MODULES_USBD  
Or  
; @CESYSGEN IF CE_MODULES_USBD  
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\EHCI86XX]  
"Prefix"="HCD"  
"Dll"="ehci86xx.dll"  
"Order"=dword:2  
"IsrDll"="giisr.dll"  
"IsrHandler"="ISRHandler"  
"HcdCapability"=dword:4 ;HCD_SUSPEND_ON_REQUEST
```

```
"PhysicalMemoryAddress"=dword:23C00000
"PhysicalMemoryLength"=dword:00400000
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\OHCI86XX]
"Prefix"="HCD"
"Dll"="ohci86xx.dll"
"Order"=dword:2
"IsrDll"="giisr.dll"
"IsrHandler"="ISRHandler"
"HcdCapability"=dword:5 ;HCD_SUSPEND_ON_REQUEST|HCD_SUSPEND_RESUME
; @CESYSGEN ENDIF CE_MODULES_USBD
```

Make sure these code is OK.

2. USB storage device speed(Test with SAMSUNG 40G USB HDD)
RevA/RevB/ES6/Es7 version chip
Read only ---- 25834000 bps, about 3229250 bytes/s
Write only ---- 13640000 bps, about 1705000 bytes/s

RevC/ES9 version chip
Read only ---- 83736000 bps, about 10467000 bytes/s
Write only ---- 45088000 bps, about 5636000 bytes/s

3. For the RevA/RevB/ES6/ES7 version SMP8634 chip, the speed of two USB port is different, the above one is USB1.1 only, the under-port is USB2.0.
If link two USB device to the two USB ports and boot up the SMP8634 board, the above one USB port may lost power, it must be reconnect.
For the RevC/ES9 version SMP8634 chip, the speed of two USB are all USB2.0
Link the low speed USB device(USB mouse/keyboard) to under-port, high speed USB device(USB disk/HDD) to above-port and boot up the SMP8634 board, the power of two USB ports is no problem.

4. Set the memory of USB EHCI device to DDR0,
"PhysicalMemoryAddress"=dword:10202800
"PhysicalMemoryLength"=dword:00100000
Reserve the part memory in config.bib
GLOBAL_MEM 90202800 00100000 RESERVED ; 1MB USB
This setting can upgrade the speed of USB port, but it will slow down the speed of SMP8634 MAC Ethernet, the speed will down to 7M bits/s.

5. Support USB host only

C) SMP8634 ATAPI

1. HDD supported
Recommend 160G big size, must be Microsoft FAT32 partition
2. HDD speed(default setting and function)
Read only ---- 69198000 bps, about 8649750 bytes/s
Write only ---- 58032000 bps, about 7254000 bytes/s
3. How to up HDD speed
Use "HalAllocateCommonBuffer" function to allocate physically contiguous memory for transfer
4. SMP8634 support 2 states IDE device, primary master and primary slave, the DMA mode is for primary master HDD only.
5. Link one DVD-ROM and one HDD to IDE port at one time
Set HDD to Master mode, set DVD-ROM to Slave mode, modify IDE registry setting in

platform.reg as follow sample,
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\bsp_atapi\Device0]
"IClass"=multi_sz:"{A4E7EDDA-E575-4252-9D6B-4195D48BB865}"
"Prefix"="DSK"
"Dll"="smp863x_atapi.dll"
"InterruptDriven"=dword:01 ; enable interrupt driven I/O
"DMA"=dword:0
"DoubleBufferSize"=dword:10000 ; 128 sector (65536 byte) double buffer
"DrqDataBlockSize"=dword:0 ; 1 sector (512 byte) DRQ data block - NOT USED
"WriteCache"=dword:01 ; enable on-disk write cache
"LookAhead"=dword:01 ; enable on-disk look-ahead
"DeviceId"=dword:00 ; device 0, i.e., primary master
;[7:3][2:0] [type of transfer][mode] [view as a byte]

```

; 0x00 - 0x0    PIO default mode          0x00
; 0x00 - 0x1    PIO default mode, diasable IORDY  0x01
; 0x01 - 0x(0-4) PIO flow control transfer mode  0x08 - 0x0c
; 0x04 - 0x(0-2) Multiword DMA mode            0x20 - 0x22
; 0x08 - 0x(0-5) Ultra DMA mode                0x40 - 0x45
"TransferMode"=dword:FF          ; fastest supported
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\bsp_atapi\Device1]
"IClass"=multi_sz:"{A4E7EDDA-E575-4252-9D6B-4195D48BB865}"
"Prefix"="DSK"
"Dll"="smp863x_atapi.dll"
"InterruptDriven"=dword:01      ; enable interrupt driven I/O
"DMA"=dword:02
"DoubleBufferSize"=dword:10000 ; 128 sector (65536 byte) double buffer
"DrqDataBlockSize"=dword:0     ; 1 sector (512 byte) DRQ data block
"WriteCache"=dword:01          ; enable on-disk write cache
"LookAhead"=dword:01           ; enable on-disk look-ahead
"DeviceId"=dword:01            ; device 1, i.e., primary slave
"TransferMode"=dword:FF        ; fastest supported

```

In this mode, the speed of HDD is slow down to 22M bits/s(read) and 31M bits/s(write).The read speed of DVD-ROM is about 11.3M bits/s.

D) SMP8634 Pflash Storage

The deive driver is disabled.

E) SMP8634 IR remote control

Please add BSP_SMP863X_IR=1 to CE project.

Follow sample code shows how to add IR remoter control to your application.

```

#include <commctrl.h>
#include <oal.h>
#include "bsp_ir.h"

#define MAX_LOADSTRING 100

// Global Variables:

typedef unsigned long RMount32;

//About IR port
HANDLE          g_hIR;          //handel of IR
HANDLE          g_hGetIRThread;
static  BOOL    g_bGetReady;

HANDLE g_OpenIRD();
DWORD  g_GetIRD(HANDLE);
BOOL   g_CloseIRD(HANDLE);

UINT   SDMC_GetIRInputThread();

// Forward declarations of functions included in this code module:

int WINAPI WinMain( HINSTANCE hInstance,
                   HINSTANCE hPrevInstance,
                   LPTSTR lpCmdLine,
                   int nCmdShow)
{
    MSG msg;
    HACCEL hAccelTable;

    // Perform application initialization:
    if (!InitInstance (hInstance, nCmdShow))
    {
        return FALSE;
    }

    hAccelTable = LoadAccelerators(hInstance, (LPCTSTR)IDC_SMP8634IRDEMO);

```

```
DWORD dwThreadID;
//Start Remote Control
g_hIR=g_OpenIRD(); //Initialize IR Port
g_bGetReady=TRUE;
if(g_hIR == FALSE)
{
#ifdef DEBUG_MODE
    OutputDebugString (TEXT("Can't Open IR Port!\n"));
#endif
    g_bGetReady=FALSE;
}
else
{
    g_hGetIRThread = CreateThread(NULL,
                                  2048,

(LPTHREAD_START_ROUTINE)SDMC_GetIRInputThread,
                                  (LPVOID)0,
                                  0,
                                  &dwThreadID);

    if(g_hGetIRThread != NULL)
    {
#ifdef DEBUG_MODE
        OutputDebugString (TEXT("Create thread!\n"));
#endif
        SetThreadPriority(g_hGetIRThread,THREAD_PRIORITY_LOWEST);
    }
    else
    {
        g_bGetReady=FALSE;
    }
}

// Main message loop:
while (GetMessage(&msg, NULL, 0, 0))
{
    if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
}

return msg.wParam;
}

.....

HANDLE g_OpenIRD()
{
    HANDLE hIRD = CreateFile(TEXT("IRD1:"),
        GENERIC_READ | GENERIC_WRITE,
        0,
        NULL,
        OPEN_EXISTING,
        FILE_ATTRIBUTE_NORMAL,
        NULL);
    if (!hIRD || hIRD==INVALID_HANDLE_VALUE)
    {
        OutputDebugString (TEXT("IRD device was not found!\n"));
        return NULL;
    }
    else
    {
        OutputDebugString (TEXT("IRD device was found!\n"));
        return hIRD;
    }
}
```

```
}

BOOL g_CloseIRD(HANDLE hIRD)
{
    CloseHandle((HANDLE)hIRD);
    return TRUE;
}

DWORD g_GetIRD(HANDLE hIRD)
{
    RMuint32 dummy;
    BOOL      bResult=FALSE;
    DWORD     dTime=1000;
    DWORD     dKey=0;

    bResult = DeviceIoControl((HANDLE)hIRD, IOCTL_IR_READ_KEY, &dTime,
sizeof(dTime), &dKey, sizeof(dKey), &dummy, NULL);
    if (!bResult )
    {
        // RETAILMSG(1, (L"IOCTL_IR_READ_KEY failed (0x%X)\n",GetLastError()));
    }
    else
    {
        // RETAILMSG (1, (TEXT("IR key pressed: %lu\r\n"), dKey));
    }

    return dKey;
}

UINT SDMC_GetIRInputThread()
{
    //my code
    HWND hwndControl;
    DWORD nSymbol=0;
    TCHAR chTemp[MAX_LOADSTRING]=TEXT("");
    RECT  rt;

    OutputDebugString (TEXT("Thread Begin!\n"));

    while(g_bGetReady && g_hIRD)
    {
        nSymbol=g_GetIRD(g_hIRD);
        if(nSymbol !=0)
        {
            OutputDebugString (TEXT("IR Input!\n"));
            wprintf(chTemp,TEXT("%lu"),nSymbol);
            OutputDebugString(chTemp);

            hwndControl=GetForegroundWindow();
            SetWindowText(hwndControl,chTemp);

            GetClientRect(hwndControl, &rt);

            DrawText(GetDC(hwndControl), chTemp, _tcslen(chTemp), &rt,
DT_SINGLELINE | DT_VCENTER | DT_CENTER);
        }
    }

    OutputDebugString(TEXT("Thread End!\n"));

    return 0;
}
//-----
}
```

F) SMP8634 I2C (I2C Slave)

This is an open source project. It is based on Gbus I/O API.

1. DLL

- a. Bsp_i2c.dll(bsp_i2c.lib)


```
    b. I2C_iisr.dll
2. Library
   Bsp_i2c_lib.lib
3. Test application
   i2c_test.exe
extern "C"{
#include <bsp.h>
#include "bsp_i2c.h"
}

int usage(int line)
{
    RETAILMSG(1, (L"\n"));
    RETAILMSG(1, (L"usage: i2c_test -b <devaddr> [-r] [-c] [-m] [-1]\n")); //mike
    RETAILMSG(1, (L"-b i2c device address\n"));
    RETAILMSG(1, (L"-r i2c register address\n"));
    RETAILMSG(1, (L"-m registers count. Default: if -r present=1, else 255 \n"));
    RETAILMSG(1, (L"-c i2c clock divider,100000 normal(default),400000 fast \n"));
    RETAILMSG(1, (L"-l list all i2c address \n")); //mike
    RETAILMSG(1, (L"\n"));
    return line;
}

int main(int argc,char**argv)
{
    BYTE reg=0, i2c_dev_addr, i2c_reg_addr=0;
    BOOL ret;

    if (argc <2)
    {
        return usage(__LINE__);
    }

    RMuint32 hi2c = BSP_I2C_Open(),k,first_reg=0,max = 256,i2c_clk_div=400000;
    if (!hi2c)
    {
        RETAILMSG(1, (L"BSP_I2C_Open() FAILED\n"));
        return __LINE__;
    }

    DWORD dList=0,*dShow;
    dShow = (DWORD *)malloc( max );

    if(argc == 2) //mike
    {
        if (argv[1][1]=='l')
        {
            for(dList =0;dList < max;dList ++ )
            {
                ret = BSP_I2C_Read(hi2c,dList,0x00,&reg,1);
                if(ret)
                {
                    dShow[dList]=1;
                }
                else
                {
                    dShow[dList]=0;
                }
            }

            RETAILMSG(1, (L"I2C Device List: \n"));
            RETAILMSG(1, (L"  0 1 2 3 4 5 6 7 8 9 A B C D E F \n"));

            for(dList =0;dList < max;dList = dList + 16)
            {
                RETAILMSG(1, (L"%X: %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d", (dList / 16), dShow[dList + 0],dShow[dList + 1],dShow[dList + 2],dShow[dList + 3],dShow[dList + 4],dShow[dList + 5],dShow[dList + 6],dShow[dList + 7],dShow[dList + 8],dShow[dList + 9],dShow[dList + 10],dShow[dList + 11],dShow[dList + 12],dShow[dList + 13],dShow[dList + 14],dShow[dList + 15]));
            }
        }
    }
}
```

```
7],dShow[dList + 8],dShow[dList + 9],dShow[dList + 10],dShow[dList + 11],dShow[dList +
12],dShow[dList + 13],dShow[dList + 14],dShow[dList + 15] ));
    }
}
else
{
    for(k=0;k<(RMuint32)argc;k++)
    {
        if (!k) continue;
        DWORD arg;

        if (k+1 >= (RMuint32)argc)
        {
            return usage(__LINE__);
        }

        if (argv[k+1][1]!='x')
            sscanf(argv[k+1],"%x",&arg);
        else
            sscanf(argv[k+1],"%d",&arg);

        if (argv[k][1]=='b')
        {
            i2c_dev_addr = (BYTE)arg;
        }else if(argv[k][1]=='r')
        {
            i2c_reg_addr = (BYTE)arg;
            first_reg=i2c_reg_addr;
            if (max==256) max=1;
        }else if(argv[k][1]=='c')
        {
            i2c_clk_div=arg;
        }else if(argv[k][1]=='m')
        {
            max = arg;
        }
        k++;
    }

    BSP_I2C_SetSpeed(hi2c,i2c_clk_div);
    max += first_reg;

    dList=first_reg;

    for (;first_reg < max;first_reg++,i2c_reg_addr++)
    {
        ret = BSP_I2C_Read(hi2c,i2c_dev_addr,i2c_reg_addr,&reg,1);

        if(max == 256 && dList == 0) //mike
        {
            if(ret)
            {
                dShow[first_reg]=reg;
            }
            else
            {
                dShow[first_reg]=0xff;
            }
        }
    }

    if(max == 256) //mike
    {
        RETAILMSG(1, (L"I2C 0x%X Device Sub Address Value List: \n",
i2c_dev_addr));
        RETAILMSG(1, (L" 0 1 2 3 4 5 6 7 8 9 A B C D E F
\n"));
    }
}
```

```

        for(dList =0;dList < max;dList = dList + 16)
        {
            RETAILMSG(1, (L"%X: %.2X %.2X %.2X %.2X %.2X %.2X %.2X
%.2X %.2X %.2X %.2X %.2X %.2X %.2X %.2X %.2X", (dList / 16), dShow[dList +
0],dShow[dList + 1],dShow[dList + 2],dShow[dList + 3],dShow[dList + 4],dShow[dList +
5],dShow[dList + 6],dShow[dList + 7],dShow[dList + 8],dShow[dList + 9],dShow[dList +
10],dShow[dList + 11],dShow[dList + 12],dShow[dList + 13],dShow[dList +
14],dShow[dList + 15] ));
        }
    }
}

free(dShow);
BSP_I2C_Close(hi2c);
return ret;
}

```

G) SMP8634 Smart Card reader

H) SMP8634 COM serial
test_serial.exe

I) SMP8634 DDI

1. DirectDraw Sample
CEDshow

CEDshow Sample Application

=====

Cedshow is the sample application that shows how to use the multimedia driver APIs such as driver escapes, alpha blending, DirectDraw, hardware demux, etc.

THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR PURPOSE.

Sigma Designs won't provide any supports of using this source code.

2. Set UI to a save TV position
Find follow setting in **platform.reg** file,

```

"OutputPosX"=dword:0 ; output window position x
"OutputPosY"=dword:0 ; output window position y
"OutputPosWidth"=dword:1000 ; output window position width
"OutputPosHeight"=dword:1000 ; output window position height

```

Modify them as follow sample

```

"OutputPosX"=dword:120 ; output window position x
"OutputPosY"=dword:120 ; output window position y
"OutputPosWidth"=dword:de0 ; output window position width
"OutputPosHeight"=dword:de0 ; output window position height

```

3. Video output mode list
Default registry setting for video output

```

"DigitalOutput"=dword:21 ; EMhwlibTVStandard_HDMI_720p59
"DigitalColorSpace"=dword:3 ; EMhwlibColorSpace_RGB_0_255,
"MainAnalogOutput"=dword:6f ; EMhwlibTVStandard_NTSC_M
"MainAnalogColorSpace"=dword:4 ; EMhwlibColorSpace_YUV_601
"ComponentAnalogOutput"=dword:65 ; EMhwlibTVStandard_720p59

"ComponentMode"=dword:6 ; EMhwlibComponentMode_YUV_SMPTE
"ComponentOrder"=dword:0 ; EMhwlibComponentOrder_RGB
"ComponentColorSpace"=dword:4 ; EMhwlibColorSpace_YUV_601

"VGAOutput"=dword:0 ; 0 = disable VGA output, 1 = enable VGA output.
; Also, when it's 1, need to set ComponentAnalogOutput to CVT_xxx or VESA_xxx,
; ComponentMode to RGB_SCART, ComponentColorSpace to RGB_0_255

```

```

enum EMhwlibTVStandard {
    EMhwlibTVStandard_Custom = 0, // Sync parameters have been changed and
    don't match a standard anymore
}

```

SIGMA DESIGNS®

EMhwhlibTVStandard_CVT_640x480x50,	//1
EMhwhlibTVStandard_CVT_640x480x60,	//2
EMhwhlibTVStandard_CVT_640x480x75,	//3
EMhwhlibTVStandard_CVT_640x480x85,	//4
EMhwhlibTVStandard_CVT_800x600x50,	//5
EMhwhlibTVStandard_CVT_800x600x60,	//6
EMhwhlibTVStandard_CVT_800x600x75,	//7
EMhwhlibTVStandard_CVT_800x600x85,	//8
EMhwhlibTVStandard_CVT_1024x768x50,	//9
EMhwhlibTVStandard_CVT_1024x768x60,	//a
EMhwhlibTVStandard_CVT_1024x768x75,	//b
EMhwhlibTVStandard_CVT_1024x768x85,	//c
EMhwhlibTVStandard_CVT_1152x864x50,	//d
EMhwhlibTVStandard_CVT_1152x864x60,	//e
EMhwhlibTVStandard_CVT_1152x864x75,	//f
EMhwhlibTVStandard_CVT_1152x864x85,	//10
EMhwhlibTVStandard_CVT_1280x960x50,	//11
EMhwhlibTVStandard_CVT_1280x960x60,	//12
EMhwhlibTVStandard_CVT_1280x960x75,	//13
EMhwhlibTVStandard_CVT_1280x960x85,	//14
EMhwhlibTVStandard_CVT_1280x1024x50,	//15
EMhwhlibTVStandard_CVT_1280x1024x60,	//16
EMhwhlibTVStandard_CVT_1280x1024x75,	//17
EMhwhlibTVStandard_CVT_1280x1024x85,	//18
EMhwhlibTVStandard_CVT_1600x1200x50,	//19
EMhwhlibTVStandard_CVT_1600x1200x60,	//1a
EMhwhlibTVStandard_DBL3_1952x1232,	//1b
EMhwhlibTVStandard_DBL3_1952x30,	//1c
EMhwhlibTVStandard_HDMI_640x480p59,	//1d
EMhwhlibTVStandard_HDMI_640x480p60,	//1e
EMhwhlibTVStandard_HDMI_480p59,	//1f
EMhwhlibTVStandard_HDMI_480p60,	//20
EMhwhlibTVStandard_HDMI_720p59,	//21
EMhwhlibTVStandard_HDMI_720p60,	//22
EMhwhlibTVStandard_HDMI_1080i59,	//23
EMhwhlibTVStandard_HDMI_1080i60,	//24
EMhwhlibTVStandard_HDMI_480i59,	//25
EMhwhlibTVStandard_HDMI_480i60,	//26
EMhwhlibTVStandard_HDMI_720x240p59,	//27
EMhwhlibTVStandard_HDMI_720x240p60,	//28
EMhwhlibTVStandard_HDMI_2880x480i59,	//29
EMhwhlibTVStandard_HDMI_2880x480i60,	//2a
EMhwhlibTVStandard_HDMI_2880x240p59,	//2b
EMhwhlibTVStandard_HDMI_2880x240p60,	//2c
EMhwhlibTVStandard_HDMI_1440x480p59,	//2d
EMhwhlibTVStandard_HDMI_1440x480p60,	//2e
EMhwhlibTVStandard_HDMI_1080p59,	//2f
EMhwhlibTVStandard_HDMI_1080p60,	//30
EMhwhlibTVStandard_HDMI_576p50,	//31
EMhwhlibTVStandard_HDMI_720p50,	//32
EMhwhlibTVStandard_HDMI_1080i50,	//33
EMhwhlibTVStandard_HDMI_576i50,	//34
EMhwhlibTVStandard_HDMI_720x288p50,	//35
EMhwhlibTVStandard_HDMI_2880x576i50,	//36
EMhwhlibTVStandard_HDMI_2880x288p50,	//37
EMhwhlibTVStandard_HDMI_1440x576p50,	//38
EMhwhlibTVStandard_HDMI_1080p50,	//39
EMhwhlibTVStandard_HDMI_1080p23,	//3a
EMhwhlibTVStandard_HDMI_1080p24,	//3b
EMhwhlibTVStandard_HDMI_1080p25,	//3c
EMhwhlibTVStandard_HDMI_1080p29,	//3d
EMhwhlibTVStandard_HDMI_1080p30,	//3e
EMhwhlibTVStandard_HDMI_2880x480p59,	//3f
EMhwhlibTVStandard_HDMI_2880x480p60,	//40
EMhwhlibTVStandard_HDMI_2880x576p50,	//41
EMhwhlibTVStandard_HDMI_1080i50_1250,	//42
EMhwhlibTVStandard_HDMI_1080i100,	//43
EMhwhlibTVStandard_HDMI_720p100,	//44

EMhwhlibTVStandard_HDMI_576p100,	//45
EMhwhlibTVStandard_HDMI_576i100,	//46
EMhwhlibTVStandard_HDMI_1080i119,	//47
EMhwhlibTVStandard_HDMI_1080i120,	//48
EMhwhlibTVStandard_HDMI_720p119,	//49
EMhwhlibTVStandard_HDMI_720p120,	//4a
EMhwhlibTVStandard_HDMI_480p119,	//4b
EMhwhlibTVStandard_HDMI_480p120,	//4c
EMhwhlibTVStandard_HDMI_480i119,	//4d
EMhwhlibTVStandard_HDMI_480i120,	//4e
EMhwhlibTVStandard_HDMI_576p200,	//4f
EMhwhlibTVStandard_HDMI_576i200,	//50
EMhwhlibTVStandard_HDMI_480p239,	//51
EMhwhlibTVStandard_HDMI_480p240,	//52
EMhwhlibTVStandard_HDMI_480i239,	//53
EMhwhlibTVStandard_HDMI_480i240,	//54
EMhwhlibTVStandard_1080p60,	//55
EMhwhlibTVStandard_1080p59,	//56
EMhwhlibTVStandard_1080p50,	//57
EMhwhlibTVStandard_1080i60,	//58
EMhwhlibTVStandard_1080i59,	//59
EMhwhlibTVStandard_1080i50,	//5a
EMhwhlibTVStandard_1080i48,	//5b
EMhwhlibTVStandard_1080i47,	//5c
EMhwhlibTVStandard_1080p30,	//5d
EMhwhlibTVStandard_1080p29,	//5e
EMhwhlibTVStandard_1080p25,	//5f
EMhwhlibTVStandard_1080p24,	//60
EMhwhlibTVStandard_1080p23,	//61
EMhwhlibTVStandard_1080i50_1250,	//62
EMhwhlibTVStandard_1080p50_1250,	//63
EMhwhlibTVStandard_720p60,	//64
EMhwhlibTVStandard_720p59,	//65
EMhwhlibTVStandard_720p50,	//66
EMhwhlibTVStandard_720p30,	//67
EMhwhlibTVStandard_720p29,	//68
EMhwhlibTVStandard_720p25,	//69
EMhwhlibTVStandard_720p24,	//6a
EMhwhlibTVStandard_720p23,	//6b
EMhwhlibTVStandard_ITU_Bt656_525,	//6c
EMhwhlibTVStandard_ITU_Bt656_240p,	//6d
EMhwhlibTVStandard_NTSC_M_Japan,	//6e
EMhwhlibTVStandard_NTSC_M,	//6f
EMhwhlibTVStandard_PAL_60,	//70
EMhwhlibTVStandard_PAL_M,	//71
EMhwhlibTVStandard_480p59,	//72
EMhwhlibTVStandard_NTSC_M_Japan_714,	//73
EMhwhlibTVStandard_NTSC_M_714,	//74
EMhwhlibTVStandard_PAL_60_714,	//75
EMhwhlibTVStandard_PAL_M_714,	//76
EMhwhlibTVStandard_480p59_714,	//77
EMhwhlibTVStandard_ITU_Bt656_625,	//78
EMhwhlibTVStandard_ITU_Bt656_288p,	//79
EMhwhlibTVStandard_PAL_BG,	//7a
EMhwhlibTVStandard_PAL_N,	//7b
EMhwhlibTVStandard_576p50,	//7c
EMhwhlibTVStandard_PAL_BG_702,	//7d
EMhwhlibTVStandard_PAL_N_702,	//7e
EMhwhlibTVStandard_576p50_702,	//7f
EMhwhlibTVStandard_VESA_640x350x85,	//80
EMhwhlibTVStandard_VESA_640x400x85,	//81
EMhwhlibTVStandard_VESA_720x400x85,	//82
EMhwhlibTVStandard_VESA_640x480x60,	//83
EMhwhlibTVStandard_VESA_640x480x72,	//84
EMhwhlibTVStandard_VESA_640x480x75,	//85
EMhwhlibTVStandard_VESA_640x480x85,	//86
EMhwhlibTVStandard_VESA_848x480x60,	//87
EMhwhlibTVStandard_VESA_800x600x56,	//88

```
EMhwlibTVStandard_VESA_800x600x60, //89
EMhwlibTVStandard_VESA_800x600x72, //8a
EMhwlibTVStandard_VESA_800x600x75, //8b
EMhwlibTVStandard_VESA_800x600x85, //8c
EMhwlibTVStandard_VESA_1024x768x43, //8d
EMhwlibTVStandard_VESA_1024x768x60, //8e
EMhwlibTVStandard_VESA_1024x768x70, //8f
EMhwlibTVStandard_VESA_1024x768x75, //90
EMhwlibTVStandard_VESA_1024x768x85, //91
EMhwlibTVStandard_VESA_1152x864x75, //92
EMhwlibTVStandard_VESA_1280x768x60RB, //93
EMhwlibTVStandard_VESA_1280x768x60, //94
EMhwlibTVStandard_VESA_1280x768x75, //95
EMhwlibTVStandard_VESA_1280x768x85, //96
EMhwlibTVStandard_VESA_1280x960x60, //97
EMhwlibTVStandard_VESA_1280x960x85, //98
EMhwlibTVStandard_VESA_1280x1024x60, //99
EMhwlibTVStandard_VESA_1280x1024x75, //9a
EMhwlibTVStandard_VESA_1280x1024x85, //9b
EMhwlibTVStandard_VESA_1360x768x60, //9c
EMhwlibTVStandard_VESA_1366x768x60, //9d
EMhwlibTVStandard_VESA_1400x1050x60RB, //9e
EMhwlibTVStandard_VESA_1400x1050x60, //9f
EMhwlibTVStandard_VESA_1400x1050x75, //a0
EMhwlibTVStandard_VESA_1400x1050x85, //a1
EMhwlibTVStandard_VESA_1600x1200x60, //a2
EMhwlibTVStandard_VESA_1600x1200x65, //a3
EMhwlibTVStandard_VESA_1600x1200x70, //a4
EMhwlibTVStandard_VESA_1600x1200x75, //a5
EMhwlibTVStandard_VESA_1600x1200x85, //a6
EMhwlibTVStandard_VESA_1792x1344x60, //a7
EMhwlibTVStandard_VESA_1792x1344x75, //a8
EMhwlibTVStandard_VESA_1856x1392x60, //a9
EMhwlibTVStandard_VESA_1856x1392x75, //aa
EMhwlibTVStandard_VESA_1920x1200x60RB, //ab
EMhwlibTVStandard_VESA_1920x1200x60, //ac
EMhwlibTVStandard_VESA_1920x1200x75, //ad
EMhwlibTVStandard_VESA_1920x1200x85, //ae
EMhwlibTVStandard_VESA_1920x1440x60, //af
EMhwlibTVStandard_VESA_1920x1440x75, //b0
EMhwlibTVStandard_VESA_640x350x70, //b1
EMhwlibTVStandard_VESA_640x480i30, //b2
EMhwlibTVStandard_VESA_640x480i60, //b3
EMhwlibTVStandard_VESA_720x400x70, //b4
EMhwlibTVStandard_VESA_640x480x66, //b5
EMhwlibTVStandard_VESA_832x624x75, //b6
EMhwlibTVStandard_VESA_1152x870x75, //b7
EMhwlibTVStandard_VESA_1280x720x60, //b8
EMhwlibTVStandard_VESA_1280x720x75, //b9
EMhwlibTVStandard_VESA_1440x900x60RB, //ba
EMhwlibTVStandard_VESA_1440x900x60, //bb
EMhwlibTVStandard_VESA_1440x900x75, //bc
EMhwlibTVStandard_VESA_1680x1050x60RB, //bd
EMhwlibTVStandard_VESA_1680x1050x60, //be
EMhwlibTVStandard_VESA_1920x1080x60i, //bf
};

enum EMhwlibAGCVersion {
    EMhwlibAGCVersion_ConstantBPP = 0,
    EMhwlibAGCVersion_AlternateBPP, //1
};

enum EMhwlibComponentMode {
    EMhwlibComponentMode_Disable = 0, //1
    EMhwlibComponentMode_RGB_SCART, //2
    EMhwlibComponentMode_RGB_SOG, //2
    EMhwlibComponentMode_RGB_SMPTE, //3
    EMhwlibComponentMode_YUV_BETACAM, //4
};
```

```

EMhwlibComponentMode_YUV_M2, //5
EMhwlibComponentMode_YUV_SMPTE //6
};

enum EMhwlibComponentOrder {
EMhwlibComponentOrder_RGB = 0, // no swap
EMhwlibComponentOrder_RGB, //1
EMhwlibComponentOrder_GRB, //2
EMhwlibComponentOrder_GBR, //3
EMhwlibComponentOrder_BRG, //4
EMhwlibComponentOrder_BGR //5
};

enum EMhwlibColorSpace {
EMhwlibColorSpace_None = 1,
EMhwlibColorSpace_RGB_16_235, //2
EMhwlibColorSpace_RGB_0_255, //3
EMhwlibColorSpace_YUV_601, //4
EMhwlibColorSpace_YUV_709, //5
EMhwlibColorSpace_YUV_601_0_255, //6
EMhwlibColorSpace_YUV_709_0_255, //7
EMhwlibColorSpace_xvYCC_601, //8
EMhwlibColorSpace_xvYCC_709, //9
EMhwlibColorSpace_xvYCC_601_0_255, //a
EMhwlibColorSpace_xvYCC_709_0_255, //b
};

enum EMhwlibSamplingMode {
EMhwlibSamplingMode_444 = 1,
EMhwlibSamplingMode_411, //2
EMhwlibSamplingMode_422, //3
EMhwlibSamplingMode_420, //4
EMhwlibSamplingMode_420_MPEG1 //5
};

enum EMhwlibColorMode {
EMhwlibColorMode_LUT_1BPP = 1,
EMhwlibColorMode_LUT_2BPP, //2
EMhwlibColorMode_LUT_4BPP, //3
EMhwlibColorMode_LUT_8BPP, //4
EMhwlibColorMode_TrueColor, //5
EMhwlibColorMode_TrueColorWithKey, //6
EMhwlibColorMode_VideoInterleaved, //7
EMhwlibColorMode_VideoNonInterleaved, //8
EMhwlibColorMode_16BPP_Alpha_LUT //9
};

enum EMhwlibInputColorFormat {
EMhwlibInputColorFormat_24BPP = 1,
EMhwlibInputColorFormat_24BPP_8565, //2
EMhwlibInputColorFormat_24BPP_5676, //3
EMhwlibInputColorFormat_32BPP, //4
EMhwlibInputColorFormat_16BPP_565, //5
EMhwlibInputColorFormat_16BPP_1555, //6
EMhwlibInputColorFormat_16BPP_4444, //7
EMhwlibInputColorFormat_31BPP_7888 //8
};

```

4. Enable HDCP

Find follow setting in **platform.reg** file,

"EnableHDCP"=dword:0 ; 0 = disable the HDCP, 1 = enable the HDCP

Modify them as follow sample

"EnableHDCP"=dword:1 ; 0 = disable the HDCP, 1 = enable the HDCP

5. SMP863x.dll Registry Entries

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\SMP86xx]
```

```
"Dll"="smp863x.dll"
```

```
"Prefix"="SDH"
```

```
"Index"=dword:1
```

SIGMA DESIGNS®

```
"Order"=dword:1
"SMP863X_RESERVED_START_DRAM0"=dword:10202800
"SMP863X_RESERVED_DRAM0_SIZE" =dword:01B00000
"SMP863X_RESERVED_START_DRAM1"=dword:0C000000
"SMP863X_RESERVED_DRAM1_SIZE" =dword:00000000
"PRIMARY_DISPLAY_SURFACE_DRAMBANK"=dword:0
"DEFAULT_ASPECT_RATIO_X"=dword:10
"DEFAULT_ASPECT_RATIO_Y"=dword:9
```

The driver needs memory to be allocated. This memory is should be reserved in you config.bib file, and then put into the registry.

SMP863X_RESERVED_START_DRAM0 specifies the start physical address in DRAM0.
SMP863X_RESERVED_DRAM0_SIZE specifies the length of this reserved memory. This memory is used to allocate required Audio DSP memory, and optionally display memory (see PRIMARY_DISPLAY_SURFACE_DRAMBANK).

SMP863X_RESERVED_START_DRAM1 specifies the start physical address in DRAM1.
SMP863X_RESERVED_DRAM1_SIZE specifies the length of this reserved memory. This memory is used to allocate Video Decoder memory, and optionally display memory (see PRIMARY_DISPLAY_SURFACE_DRAMBANK).

PRIMARY_DISPLAY_SURFACE_DRAMBANK is used to tell the decoder which DRAM bank to allocate the display memory from. The proper choice here is a trade-off between available memory and memory bandwidth. Since the main video decoder is running out of bank 1, if you want a high resolution graphics mode (greater than 576P), then it is better to run the display from bank 0. Note that you will have to allocate more memory in bank 0 (through SMP863X_RESERVED_DRAM0_SIZE). The amount of memory you require would depend on the graphics resolution. For example, if you wanted a GDI graphics plane of 1920x1080, then the amount of memory required would be $(1920 \times 1080 \times 4) = (8294400 \text{ bytes})$. You also have to add about 2048 bytes for some display structures.

DEFAULT_ASPECT_RATIO_X and **DEFAULT_ASPECT_RATIO_Y** are used to specify the default aspect ratio. Note that these values are in HEX, so a 16:9 aspect ration would be:

```
"DEFAULT_ASPECT_RATIO_X"=dword:10
"DEFAULT_ASPECT_RATIO_Y"=dword:9
```

So a 4:3 aspect ration would be:

```
"DEFAULT_ASPECT_RATIO_X"=dword:4
"DEFAULT_ASPECT_RATIO_Y"=dword:3
```

J) SMP8634 HW cursor

Find follow setting in **platform.reg** file,
"EnableHwCursor"=dword:1 ; 0 = disable, 1 = enable

K) SMP8634 WAVE device

```
; Sigma Designs 8634 wave built-in driver
;
; Supports only wave out
[HKEY_LOCAL_MACHINE\Drivers\Builtin\Audio]
"Prefix"="WAV"
"Dll"="wave863x.dll"
"Index"=dword:1
"Order"=dword:1
; "Order"=dword:10
```

L) SMP8634 DDR RAM

1.Physical address
DRAM0 ---- 0x10000000-0x20000000.
DRAM1 ---- 0x20000000-0x30000000

2.Enhance DRAM stability
Remove RP54, RP59, RP64, RP69

In serial port window, the debug message shows follow info.

```
xenv cs2 ok
power supply: ok
dram0 ok (7) --> This number is must bigger than 6
dram1 ok (7) --> This number is must bigger than 6
zboot ok
```


M) SMP8634 RTC(RealTime Clock)

RTC driver for the SMP8634

=====

Overview

This driver allows the persistence of the realtime clock of Windows CE to be persistent across power and/or reset cycles.

This driver has been tested with xos version 0xe0 and above.

This driver should be working for production as well as development chips.

Your board MUST have the RTC support for this driver to work.

How to install

Please read all the steps before installing.

0. Open up the file rtc.c.

There should be a define called SMP8634_RTC_4KB_PHYSICAL_ADDRESS.

```
#define SMP8634_RTC_4KB_PHYSICAL_ADDRESS 0x101FF000 // << change this  
address as necessary
```

You can change this address if you reserve a 4kb portion inside config.bib.

1. add the bsp_rtc directory into the /smp863x/src/libs directory

2. change the dirs directory in the /smp863x/src/libs to include the compilation of the bsp_rtc directory

```
SYNCHRONIZE_BLOCK=1
```

```
DIRS=\  
  board_callbacks \  
  bsp_cach \  
  gbus_ke \  
  bsp_pci \  
  bsp_freq \  
  bsp_flash \  
  bsp_ide_lib \  
  bootfs \  
  bsp_xrpc \  
  bsp_reboot \  
  bsp_eth_lib \  
  winmain_main \  
  dbg_helper \  
  gpio \  
  power \  
  bsp_rtc \  
  
```

3. change the 'sources' file in the /smp863x/src/kernel/kern directory so that the kernel is linked with the smp863x_rtc_timer.lib library instead of the oal_rtc_timer.lib library.

```
TARGETNAME=kern  
TARGETTYPE=PROGRAM  
RELEASETYPE=PLATFORM  
SYNCHRONIZE_DRAIN=1  
WINCECPU=1  
EXEENTRY=StartUp  
EXEBASE=0x92600000  
LDEFINES=-subsystem:native /DEBUG /DEBUGTYPE:CV /FIXED:NO  
SOURCES= \  
  kitl.c \  
  
```

```
TARGETLIBS= \  
  $(COMMONOAKROOT)\lib\$(CPUDEPPATH)\nk.lib \  
  $(PLATCOMMONLIB)\$(CPUDEPPATH)\oal_intr_common.lib \  
  $(PLATCOMMONLIB)\$(CPUDEPPATH)\oal_power_stub.lib \  
  $(PLATCOMMONLIB)\$(CPUDEPPATH)\oal_ioctl.lib \  
  $(PLATCOMMONLIB)\$(CPUDEPPATH)\oal_pci.lib \  
  $(PLATCOMMONLIB)\$(CPUDEPPATH)\oal_io_pci.lib \  
  
```

```

$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_timer_mips_rtc.lib \
$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_timer_varidle.lib \
$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_timer_vartick.lib \
$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_ilt_stub.lib \
$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_other.lib \
$( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_log.lib \
$( _COMMONOAKROOT)\lib\$( _CPUINDPATH)\ddk_io.lib \
$( _COMMONOAKROOT)\lib\$( _CPUDEPPATH)\pcireg.lib \
$( _COMMONOAKROOT)\lib\$( _CPUINDPATH)\fulllibc.lib \
$( _TARGETPLATROOT)\lib\$( _CPUDEPPATH)\bsp_cache.lib \
$( _TARGETPLATROOT)\lib\$( _CPUDEPPATH)\oal.lib \
$( _TARGETPLATROOT)\lib\$( _CPUDEPPATH)\bsp_eth_lib.lib \
$( _TARGETPLATROOT)\lib\$( _CPUDEPPATH)\smp863x_rtc_timer.lib \
# $( _PLATCOMMONLIB)\$( _CPUDEPPATH)\oal_rtc_timer.lib \

```

4. Do the same for the 'sources' file in the /smp863x/src/kernel/kernkitl and /smp863x/src/kernel/kernkitlprof directories.

5. Recompile the kernel and make your platform again.

N) SMP8634 GPIO

1. Library
gpio_lib.lib
2. Test application
Gbio_test.exe

```

#include <bsp.h>
#include "..\inc\bsp_gpio.h"

```

```

int main(int argc,char**argv)
{
    int i;
    for (i=0;i<MAX_PIN_INDEX;i++)
    {
        RETAILMSG(1, (TEXT(" pin[%02d].dir      =
%s\n") ,i,gpio_get_direction(i)?TEXT("write"):TEXT("read")));
        RETAILMSG(1, (TEXT(" pin[%02d].state    = %.1d\n") ,i,gpio_read(i)));
        RETAILMSG(1, (TEXT("\n")));
    }
    return __LINE__;
}

```

O) SMP8634 PWM

- | | | | | |
|--------|-------|------|-------|------------------|
| GPIO15 | ----- | PWM0 | ----- | T30(Check point) |
| GPIO14 | ----- | PWM1 | ----- | T29(Check point) |

1. Library
Gbus_ke.lib
2. Test application

```

#include <bsp.h>

```

```

#ifndef REG_BASE_system_block
#define REG_BASE_system_block    0x00010000 /* width RUint32 */
#endif
#ifndef SYS_gpio15_pwm
#define SYS_gpio15_pwm           ,    0x0510 /* width RUint32 */
#endif
#ifndef SYS_gpio14_pwm
#define SYS_gpio14_pwm           0x0514 /* width RUint32 */
#endif
#define GBUS_PWM0_CONTROL        (REG_BASE_system_block +
SYS_gpio15_pwm) //GPIO 15
#define GBUS_PWM1_CONTROL        (REG_BASE_system_block +
SYS_gpio14_pwm) //GPIO 14
#define PWM_ENABLE                0x01000000
#define PWM_PRE_DIV(x)            (x<<16) //the minimum space rate value
#define PWM_LEVEL(x)              (x<<4)

//  main  ////////////////////////////////////////
//

```

```
int main(int argc,char**argv)
{
    //need to open GBUS
    struct gbus *pgbus;
    unsigned int val = 0;
    unsigned int backlight = 100; // this value can be change for backlight.

    pgbus = gbus_open(0);

    //for GPIO 15
    val = PWM_ENABLE|PWM_PRE_DIV(7)|PWM_LEVEL(backlight);
    RETAILMSG(1, (TEXT("Write PWM0 val = %d \n"), val));
    gbus_write_uint32(pgbus, GBUS_PWM0_CONTROL,val);
    val=0;
    val = gbus_read_uint32(pgbus,GBUS_PWM0_CONTROL);
    RETAILMSG(1, (TEXT("Read PWM0 val return = %d \n"), val));
    val &= 0xffff;
    val >>= 4;
    RETAILMSG(1, (TEXT("Read PWM0 val = %d \n"), val));

    //for GPIO 14
    backlight=50;
    val = PWM_ENABLE|PWM_PRE_DIV(7)|PWM_LEVEL(backlight);
    RETAILMSG(1, (TEXT("Write PWM1 val = %d \n"), val));
    gbus_write_uint32(pgbus, GBUS_PWM1_CONTROL,val);
    val=0;
    val = gbus_read_uint32(pgbus,GBUS_PWM1_CONTROL);
    RETAILMSG(1, (TEXT("Read PWM1 val return = %d \n"), val));
    val &= 0xffff;
    val >>= 4;
    RETAILMSG(1, (TEXT("Read PWM1 val = %d \n"), val));

    gbus_close(pgbus);

    return __LINE__;
}
```

P) SMP8634 Hardware Demux

The latest version DVB tuner source filter supports three models turner

1) Philips tuner Tu1216 ----- DVB-T

Note: The default I2C address for Philips Tu1216 is 0x10,0x11, but for vantage board, the two address is hold by other device, so must use UP resistor to change it(change R248 to R247 with 10KΩ)

Then the address will be changed to 0x14,0x15, so the parameter for driver will be changed from 0x10,0xC0 to 0x14(TDA1004HT),0xC0(TDA6651)

2) Philips Tu1236D ----- ATSC/NTSC

3) Philips FQ1216ME ----- PAL/SECAM

1. Dll

a. dump.dll

This is a direct show filter which save date from source filter

b. tunersrc.dll

This is a direct show source filter for DVB tuner

c. HWDEMUX863x.DLL

This is a direct show filter which uses the hardware demux block of the SMP8634.

It is intended to be used when the incoming transport stream is originating from the the SPI or SSI inputs of the SMP8634.

Current version releases follow interfaces

```
// {bfc6c826-4b93-4a66-8f58-ed0b7047311c}
DEFINE_GUID(CLSID_HWTSDemux863x,
0xbfc6c826, 0x4b93, 0x4a66, 0x8f, 0x58, 0xed, 0x0b, 0x70, 0x47, 0x31, 0x1c);
```

```
// {2eb12b15-60f2-446c-a7b5-0ac88bc3dcd}
DEFINE_GUID(IID_IHWTSDemux863x,
0x2eb12b15, 0x60f2, 0x446c, 0xa7, 0xb5, 0x0a, 0xc8, 0x8b, 0xc3, 0xdc, 0xda);
```

```
#ifndef _IHWDEMUX863X_H
#define _IHWDEMUX863X_H

////////////////////////////////////

// {02a53ccf-7917-4bee-9da5-c000c5816e48}
DEFINE_GUID(MEDIATYPE_PSI_PAT,
0x02a53ccf, 0x7917, 0x4bee, 0x9d, 0xa5, 0xc0, 0x00, 0xc5, 0x81, 0x6e, 0x48);

// {d2964e1d-9ea8-4e27-87c8-17cb560069a0}
DEFINE_GUID(MEDIATYPE_PSI_PMT,
0xd2964e1d, 0x9ea8, 0x4e27, 0x87, 0xc8, 0x17, 0xcb, 0x56, 0x00, 0x69, 0xa0);

// {2f8fe1e1-89bd-4ff9-a1eb-fae6475bca84}
DEFINE_GUID(MEDIATYPE_PSI_CAT,
0x2f8fe1e1, 0x89bd, 0x4ff9, 0xa1, 0xeb, 0xfa, 0xe6, 0x47, 0x5b, 0xca, 0x84);

// {3b0bf0d8-cb5d-4b2c-874d-b9847931a3d9}
DEFINE_GUID(MEDIATYPE_PSI_ECM0,
0x3b0bf0d8, 0xcb5d, 0x4b2c, 0x87, 0x4d, 0xb9, 0x84, 0x79, 0x31, 0xa3, 0xd9);

// {ab26f0e9-a50b-4382-9595-40419687d9e7}
DEFINE_GUID(MEDIATYPE_PSI_ECM1,
0xab26f0e9, 0xa50b, 0x4382, 0x95, 0x95, 0x40, 0x41, 0x96, 0x87, 0xd9, 0xe7);

// {0d7de6c1-d66c-4461-94a5-a60f4e14f488}
DEFINE_GUID(MEDIATYPE_TSPayload,
0x0d7de6c1, 0xd66c, 0x4461, 0x94, 0xa5, 0xa6, 0x0f, 0x4e, 0x14, 0xf4, 0x88);

// {1692a63f-5664-46bc-a078-abbdbd9eb9ea}
DEFINE_GUID(MEDIATYPE_PCR,
0x1692a63f, 0x5664, 0x46bc, 0xa0, 0x78, 0xab, 0xbd, 0xbd, 0x9e, 0xb9, 0xea);

// {dcd8bbf3-01ce-40a8-b791-4968ce340115}
DEFINE_GUID(MEDIATYPE_Private,
0xdcd8bbf3, 0x01ce, 0x40a8, 0xb7, 0x91, 0x49, 0x68, 0xce, 0x34, 0x01, 0x15);

// {86f2e4c5-a89f-4fa1-855b-d63d276aa291}
DEFINE_GUID(MEDIATYPE_PSI_SECTION,
0x86f2e4c5, 0xa89f, 0x4fa1, 0x85, 0x5b, 0xd6, 0x3d, 0x27, 0x6a, 0xa2, 0x91);

#define MAX_OUTPUT_PINS 32

typedef enum
{
    eOutputPinType_TS = 1, // Only 1 output
    eOutputPinType_PSI_PAT, // Only 1 output
    eOutputPinType_PSI_PMT, // Only 1 output
    eOutputPinType_PSI_CAT, // Only 1 output
    eOutputPinType_PSI_ECM0, // Only 1 output
    eOutputPinType_PSI_ECM1, // Only 1 output
    eOutputPinType_VPES, // Multiple outputs
    eOutputPinType_APES, // Multiple outputs
    eOutputPinType_PCR, // Only 1 output (first A/V pins)
    eOutputPinType_VPayload, // Multiple outputs
    eOutputPinType_APayload, // Multiple outputs
    eOutputPinType_PSISection, // Multiple outputs - generic PSI section filter
} eOutputPinType;

typedef enum
{
    eInputSpi_None,
    eInputSpi_Serial,
    eInputSpi_Parallel
} eInputSpi;
```

```
#define MAX_TS_PROGRAM_NUMBER 128 // Max program numbers of the transport
stream
#define MAX_ES_PIDS_PER_PROGRAM 30 // Max elementary stream PIDs per program
typedef struct
{
    WORD pid;
    WORD type;
} PID;
typedef struct
{
    PID esPid [MAX_ES_PIDS_PER_PROGRAM];
    WORD ecmPid [MAX_ES_PIDS_PER_PROGRAM];
    DWORD count;
} ES_PIDS;
typedef struct
{
    ES_PIDS video;
    ES_PIDS audio;
    DWORD pcrPid;
    DWORD progNum;
} PROGRAM_PIDS;
typedef struct
{
    PROGRAM_PIDS progs [MAX_TS_PROGRAM_NUMBER];
    DWORD count; // Total number of programs
} STREAM_PIDS;
```

```
typedef struct
{
    IPin *pPin[MAX_OUTPUT_PINS];
    WORD pid[MAX_OUTPUT_PINS];
    DWORD progNum[MAX_OUTPUT_PINS];
    eOutputPinType pinType[MAX_OUTPUT_PINS];
    DWORD pinCount;
} PIN_PID_INFO;
```

/*
For match comparison any of the first 96 bits in the section table header have to match
the positive or negative criteria imposed by the filter.

Positive match means that all the positive masked bits from the PSI header have to match
bit by bit the compare value.

Negative match means that all the negative masked bits from the PSI header have to not
match
at least one bit from the compare value.

The logic operation performed is:

```
PositiveMask = mask & mode  
NegativeMask = mask & !mode  
MatchCondition =( PositiveMask & header == PositiveMask & comp ) && ( !NegativeMask  
|| (NegativeMask &(header^comp))).
```

Match example:

```
StSectionFilterMask matchSectionTable[2] = {  
    { 0xff, 0xff, // no expand or and link  
      {0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, //  
mask  
      {0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, //  
mode= all positive match  
      {0xc7, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}}, //  
ATSC_MGT  
  
    { 0xff, 0xff, // no expand or and link  
      {0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, //  
mask  
      {0xF9, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, //  
mode=positive& negative match
```

```
{0xce, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00}, //
0xc8, 0xca, 0xcc
}
*/
typedef struct { /* based on struct PSFMatchSection_type */
    unsigned char expand_link_index; // reserved for extending the length of section filter,
    value between 0...31 or 0xFF if not used
    unsigned char and_link_index; // reserved for "logical and" two section filters,
    value between 0...31 or 0xFF if not used
    unsigned char mask[12]; // 96 bit mask - 1 means the mask is active
    unsigned char mode[12]; // 96 bit mask - 1 means positive match, 0
    means negative match
    unsigned char comp[12]; // 96 bit value
} StSectionFilterMask;

interface IHWTSDemux863x : public IUnknown
{
    // NOTE: application must call this API first once the filter is loaded to
    // set the demux task properties. The valid task indexes are from 0 to 2.
    // bTSCapture specifies whether this task is used to capture the transport
    // stream or not. Note that in order to capture the TS stream, the input SPI
    // must NOT be eInputSpi_None. If there is already one demux task, this API
    // will delete that demux task before setting the new values.
    virtual HRESULT STDMETHODCALLTYPE SetDemuxTaskProp (INT taskIndex, eInputSpi
    inputSpi, BOOL bTSCapture) = 0;

    // Set the input SPI interface. Default is parallel. If the input is from the local
    // file, the filter will switch automatically when it detects the source filter
    // supporting the IAsyncReader.
    // Call before the pin connections
    virtual HRESULT STDMETHODCALLTYPE SetInputSpi (eInputSpi spi) = 0;

    // Add a new output pin type. Max total output pin is 32. Use GetPinPidInfo to
    // get the current number of output pins.
    // For non-payload pins, only one pin of each type (i.e PAT, PMT, CAT, TS).
    // If the task is for capturing the TS stream, only one output pin eOutputPinType_TS is
    supported.
    // NOTE: This API doesn't accept eOutputPinType_PSISection. Use
    AddOutputPinSectionFilter instead.
    // Call before the pin connections
    virtual HRESULT STDMETHODCALLTYPE AddOutputPin (eOutputPinType pinType) = 0;

    // Add a generic section filter output pin. By default, the h/w demux filter creates section
    filters
    // for PAT, PMT, CAT, ECM0, and ECM1. Also see more notes of StSectionFilterMask from
    above.
    // The output pin type will be eOutputPinType_PSISection and the media type is
    MEDIATYPE_PSI_SECTION.
    // Call before the pin connections
    virtual HRESULT STDMETHODCALLTYPE AddOutputPinSectionFilter (TCHAR
    *pszPinName, StSectionFilterMask *pSectionMask, WORD wSectionPid) = 0;

    // Get all available audio and video PIDS in the stream.
    virtual HRESULT STDMETHODCALLTYPE GetPinPidInfo (PIN_PID_INFO *pPinPidInfo,
    STREAM_PIDS* pStreamPids) = 0;

    // Set the PID of a particular output pin.
    virtual HRESULT STDMETHODCALLTYPE SetPid (IPin *pOutputPin, DWORD dwProgNum,
    WORD wPid) = 0;

    // Set the data flow mode for the first A/V output pins. In the pass-through mode,
    // the data is sent direct to the decoders. So the output pins won't output any
    // media samples. In the non-pass-through mode, the data is sent through the output
    // pins in the media samples format. Note that the data transfer is slower in this mode.
    // This API has no affect for the demux task that captures the TS stream.
    // Call before the pin connections
    virtual HRESULT STDMETHODCALLTYPE SetPassThrough (BOOL bPassThrough) = 0;
};
```

////////////////////////////////////

#endif

2. Library
 - a. turnerapi.lib
 - b. tunhw.lib(tuner hardware)
 - c. tmbsl.lib
 - d. helper.lib
3. Test application
 - a. tunerapp.exe(It is included in tunerapp dll project)
 - b. hwdemuxtest.exe

Note:

1)Except 1.c, this part is not included in default BSP and multimedia package, it's just for professional DVB developer.
2)Registry setting("DelaySettingCodecTypes"=dword:1) can make the hardware demux shows follow error message and make the system breakdown

728202 PID:d72a0bda TID:36e4c852 HWDEMUX: ParsePAT section_syntax_indicator 0 !
728440 PID:d7997dc2 TID:36e4c852 Exception 004 Thread=96e46dd8 Proc=d7997dc2 'device.exe'

728440 PID:d7997dc2 TID:36e4c852 AKY=00000089
PC=01ccf800(smp863x.dll+0x000cf800) RA=01ca3dd8(smp863x.dll+0x000a3dd8)
BVA=b01b0009

728442 PID:d7997dc2 TID:36e4c852 SMP863X: Exception during IOCTL:
728443 PID:d7997dc2 TID:36e4c852 SMP863X: dwCode = 0x900226d0
728444 PID:d7997dc2 TID:36e4c852 SMP863X: Function = 0x00000044 (68)
728445 PID:d7997dc2 TID:36e4c852 SMP863X: pBufIn = 0x1014fcc8
728446 PID:d7997dc2 TID:36e4c852 SMP863X: dwLenIn = 0x00000048
728447 PID:d7997dc2 TID:36e4c852 SMP863X: pBufOut = 0x00000000
728448 PID:d7997dc2 TID:36e4c852 SMP863X: dwLenOut = 0x00000000
728449 PID:d7997dc2 TID:36e4c852 SMP863X: pdwActualOut = 0x1014fcc0

Comment the DelaySettingCodecTypes"=dword:1 setting or set DelaySettingCodecTypes"=dword:0 can repair this problem

Q) SMP8634 Hardware Jpeg decode

1. Library
 - jpeg_api.lib
2. Head file
 - jpeg_api.h
3. Demo
 - HwPlayJpeg

R) SMP8634 LVDS output

Modify LVDS registry setting in **platform.reg** as follow sample, this sample setting sets a 800x600 size video output from LVDS.

```
[HKEY_LOCAL_MACHINE\System\GDI\Drivers]  
"Display"="ddi_86xx.dll"
```

```
;  
; See smp863x_formats.h for the proper hex values for each video mode  
;
```

```
"DigitalOutput"=dword:0  
"DigitalColorSpace"=dword:3  
"MainAnalogOutput"=dword:6f  
"MainAnalogColorSpace"=dword:4  
"ComponentAnalogOutput"=dword:65  
"ComponentMode"=dword:6  
"ComponentOrder"=dword:0  
"ComponentColorSpace"=dword:4  
"VGAOutput"=dword:0 ; 0 = disable VGA output, 1 = enable VGA output.  
; Also, when it's 1, need to set ComponentAnalogOutput to CVT_xxx or VESA_xxx,  
; ComponentMode to RGB_SCART, ComponentColorSpace to RGB_0_255  
"ScreenWidth"=dword:00000320 ; 800  
"ScreenHeight"=dword:00000258 ; 600  
"OutputPosX"=dword:0 ; output window position x  
"OutputPosY"=dword:0 ; output window position y
```

```
"OutputPosWidth"=dword:1000 ; output window position width
"OutputPosHeight"=dword:1000 ; output window position height
"MemorySize"=dword:00800000 ; extra memory size for h/w acceleration,
; see note in the smp863x.reg
; regarding the PRIMARY_DISPLAY_SURFACE_DRAMBANK
"EnableHwCursor"=dword:1 ; 0 = disable, 1 = enable
"LiveDetectHdmiConnection"=dword:1 ; 0 = driver won't do live hdmi detection, 1 =
check live hdmi connection
"EnableHDCP"=dword:0 ; 0 = disable the HDCP, 1 = enable the HDCP
"DefaultKeyColor"=dword:00010101 ; In RGB format: R=3rd byte, G= nd byte, B=1st byte
"EnableEDIDDetection"=dword:0 ; 0 = disable, 1 = enable. Disable/enable the HDMI
preferred mode.
"HDMI2CModule"=dword:2 ; 0 = software, 1 = hardware, 2 = built-in hdmi
"InvertClock"=dword:1 ; 0 = do not invert digital video clock, 1 = invert (default is inverted)

"LVDS_Enable"=dword:1
"LVDS_GPIOFieldIDOutput"=dword:B ; board dependent
"LVDS_GPIOPanelOn"=dword:E ; board dependent
```

```
; Default custom digital output video mode - 800X600
[HKEY_LOCAL_MACHINE\System\GDI\Drivers\CustomDigitalVideoMode]
```

```
"PixelClock"=dword:2122800
"HActive"=dword:320
"HFrontPorch"=dword:0
"HSyncWidth"=dword:80
"HBackPorch"=dword:0
"HSyncPolarity"=dword:0 ; TRUE: positive, FALSE: negative
"VActive"=dword:258
"VFrontPorch"=dword:1
"VSyncWidth"=dword:3
"VBackPorch"=dword:14
"VSyncPolarity"=dword:0 ; TRUE: positive, FALSE: negative
"Interlaced"=dword:0
```

Note: $(HFrontPorch + HSyncWidth + HBackPorch + HActive) \times (VFrontPorch + VFrontPorch + VBackPorch + VActive) = PixelClock$

S) SMP8634 Multi-Decoder

1. This driver now supports multiple instances.
The first instance will use the main video scaler and decoder 16.
The memory will be taken from dram1.
The second decoder will use the vcr scaler and decoder 0.
The memory will be taken from dram0.

Note that if there is not enough memory allocated in dram0, the decoder will not play.

In the second instance of the filter, the audio is disabled. In addition, only the first instance supports HD resolutions. All other instances support SD resolutions only.

If "DelaySettingCodecTypes" is set to 1, then multiple instances are not supported.

2. Test the multi-decoder feature

Open a HTML page with windows media player OCX, render a media file with it, open the CEPlayer, render the second media file with it. If the multi-decoder setting is no problem, there are two media shows in video output now.

T) SMP8634 VGA output

U) SMP8634 Audio and Video Captures filter

6) How to upgrade/modify the CE boot loader

A) Update boot.bin

Open the boot.bin file of CE project, download it to SMP8634 board as same as nk.bin file.

Make sure see the output message "INFO:lanch address in boot address space spin forever" in the serial window, reboot SMP8634, then new CE boot loader is ready for SMP8634 board

B) Update XENV

Setxenv.exe

This demo shows how to write flash with CFI command.

The new sample is based on MMU mode.

C) Set the default value for boot loader

If we write a new CE bootloader binary file a empty flash, the default setting of bootloader is NULL, so we have to set and save it in bootloader main menu for each time.

In fact, we can set the default value in bootloader too, please check the follow function in Src\Bootloader\main.c file as you need.

```
//-----  
//  
// Function: OEMPreDownload  
//  
// This function is called before downloading an image. There is place  
// where user can be asked about device setup.  
//  
// UULONG OEMPreDownload()  
{  
    UULONG rc = BL_ERROR;  
    BOOT_CFG *pCfg = OALPAtOCA(IMAGE_BOOT_CONFIG_FLASH_PA_START);  
  
    OALLog(L"INFO: Preadownload....\r\n");  
  
    // First try to check if there is an eboot config structure  
    if (  
        pCfg->signature == BOOT_CFG_SIGNATURE &&  
        pCfg->version == BOOT_CFG_VERSION  
    ) {  
        OALLog(L"INFO: Boot configuration found at 0x%08x\r\n", pCfg);  
        memcpy(&g_bootCfg, pCfg, sizeof(g_bootCfg));  
  
        g_bootCfg.bspFlags &= BSP_BOOTF_IsPciHost;  
  
    } else {  
        OALLog(  
            L"WARN: Boot config wasn't found at 0x%08x, using defaults\r\n",  
            pCfg  
        );  
        memset(&g_bootCfg, 0, sizeof(g_bootCfg));  
        g_bootCfg.signature = BOOT_CFG_SIGNATURE;  
        g_bootCfg.version = BOOT_CFG_VERSION;  
        g_bootCfg.bootDevLoc.Ifctype = PCIBus;  
        g_bootCfg.bootDevLoc.BusNumber = 0;  
        g_bootCfg.bootDevLoc.LogicalLoc = 0x00000100;  
        g_bootCfg.bootDevLoc.Pin = 1;  
        g_bootCfg.kitlFlags =  
OAL_KITL_FLAGS_ENABLED|OAL_KITL_FLAGS_DHCP|OAL_KITL_FLAGS_VMINI;  
        g_bootCfg.kitlDevLoc.Ifctype = PCIBus;  
        g_bootCfg.kitlDevLoc.BusNumber = 0;  
        g_bootCfg.kitlDevLoc.LogicalLoc = 0x00000100;  
        g_bootCfg.kitlDevLoc.Pin = 1;  
        g_bootCfg.ipAddress = 0;  
        g_bootCfg.ipMask = 0;  
        g_bootCfg.ipRoute = 0;  
        g_bootCfg.bspFlags = BSP_BOOTF_IsPciHost;  
        {  
            PWCHAR pdest = g_bootCfg.BootFileName,  
                psrc = L"nk.bin";  
            while(*pdest++=*psrc++);  
        }  
    }  
}
```

```
        g_bootCfg.protected_flash_area_start =
IMAGE_FLASH_PROTECTED_AREA_START;
        g_bootCfg.protected_flash_area_size =
IMAGE_FLASH_PROTECTED_AREA_SIZE;
    }

    g_bspFlags = g_bootCfg.bspFlags;
    if (g_bspFlags & BSP_BOOTF_IsPciHost){
        OALPCIInit();
        // We must initialize PCI bus first
        OALPCIConfig(0, 0, 0, 0, 0, 0, NULL);
    }

    // Call configuration menu
    BLMenu();

    if (!(g_bspFlags & BSP_BOOTF_IsPciHost)){
        OALLog(L" IsPciHost is not enabled, return BL_ERROR\r\n ");
        return BL_ERROR;
    }
    memset(&g_memfile,0,sizeof(g_memfile));

    if (!g_bspFlags & BSP_BOOTF_IsPciHost) return BL_ERROR;

    // Image download depend on protocol
    g_DownloadDeviceType = OALKitIDeviceType(&g_bootCfg.bootDevLoc,
g_bootDevices);
    switch (g_DownloadDeviceType){
    case OAL_KITL_TYPE_FLASH:
        g_memfile.NkBinAddr = OALPtoCA(IMAGE_FLASH_PA_START +
g_bootCfg.nk_bin_flash_offset);
        g_memfile.Offset = 0;
        g_DownloadDeviceType = OAL_KITL_TYPE_FLASH;
        rc = BLFlashDownload(g_memfile.NkBinAddr);
        break;
    case OAL_KITL_TYPE_ETH:
        rc = BLEthDownload(&g_bootCfg, g_bootDevices);
        break;
    case OAL_KITL_TYPE_MEM:
        g_memfile.NkBinAddr = (PVOID)OALPtoCA(IMAGE_BOOT_NK_BIN_PA_START);
        rc = BLMemDownload();
        break;
    case OAL_KITL_TYPE_IDE:
        rc = IdeDownload(&g_bootCfg, g_bootDevices);
        break;
    case OAL_KITL_TYPE_PFLASH:
        rc = PFlashDownload(&g_bootCfg, g_bootDevices);
        break;
    }
    return rc;
}
```

7) How to modify wince feature on SMP8634

A) Web server

1. Open the platform.reg file of CE project, add follow settings at the end of file

```
;Web server
HKEY_LOCAL_MACHINE\COMM\HTTPD\ROOTS\WebAdmin]
@="\\windows\httpdadm.dll"
"a"=dword:0
```

```
[HKEY_LOCAL_MACHINE\COMM\HTTPD\ROOTS\BasicOnly]
@="\\\"
"a"=dword:1
"Basic"=dword:1
"NTLM"=dword:0
"dirbrowse"=dword:1
```

```
[HKEY_LOCAL_MACHINE\COMM\HTTPD\ROOTS\NTLMOnly]
@="\\\"
"a"=dword:1
"Basic"=dword:0
"NTLM"=dword:1
"dirbrowse"=dword:1
```

```
[HKEY_LOCAL_MACHINE\COMM\HTTPD\ROOTS\BothAuth]
@="\\\"
"a"=dword:1
"Basic"=dword:1
"NTLM"=dword:1
"dirbrowse"=dword:1
```

2. Make the nk.bin file, download it to SMP8634 board
3. Use the IE of PC system to open the <http://192.168.1.250> link, set the password of admin and reboot the web server
4. Use the IE of PC to open the <http://192.168.1.250/Webadmin> link, setup the web server

B) FTP server

- Open the platform.reg file of CE project, add follow settings at the end of file

```
;Ftp server
[HKEY_LOCAL_MACHINE\COMM\FTPD]
"IsEnabled"=dword:1
"AllowAnonymous"=dword:1
"AllowAnonymousUpload"=dword:1
"AllowAnonymousVroots"=dword:1
"DefaultDir"="\\Hard Disk\\"
"AllowLowPortValues"=dword:1
```

Default FTP login account

User: anonymous

Passwd: anonymous

C) Telnet server

D) SMB server

- Open the project.reg file of CE project, add follow settings at the end of file

```
;File server
[HKEY_LOCAL_MACHINE\Services\Smbserver]
"AdapterList"="MAC86XX1"
```

```
[HKEY_LOCAL_MACHINE\Services\Smbserver\Shares\Root]
"Path"="\\Hard Disk"
"Type"=dword:0
"UserList"="admin"
```

E) Hive-based Registry

- 1.HDD Device

- Open the project.reg file of CE project, add follow settings at the end of file

```
; HIVE BOOT SECTION
[HKEY_LOCAL_MACHINE\init\BootVars]
"Start DevMgr"=dword:1
[HKEY_LOCAL_MACHINE\System\StorageManager\Profiles\HDProfile\FATFS]
"MountFlags"=dword:2
; END HIVE BOOT SECTION
```

Open the platform.reg file of CE project, add follow line above 'IF BSP_SMP863X_ATAPI' line

```
; HIVE BOOT SECTION
```

Add follow line under 'ENDIF BSP_SMP863X_ATAPI' line

```
; END HIVE BOOT SECTION
```

Add follow line each under

```
[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\bsp_atapi],[HKEY_LOCAL_MACHINE\Drivers\BuiltIn\bsp_atapi\Device0] and [HKEY_LOCAL_MACHINE\Drivers\BuiltIn\bsp_atapi\Device1]
```

```
"Flags"=dword:1000
```

Add follow Environment Variable Values to CE project.

```
PRJ_ENABLE_FSMOUNTASROOT = 1
```

```
PRJ_ENABLE_REGFLUSH_THREAD = 1
```

```
PRJ_BOOTDEVICE_ATAPI = 1
```

Sysgen Current CE project, and make the new NK.bin

F) IE 6.0 for Windows CE

1) Open the platform.reg file of CE project, add follow settings at the end of file, you can set the IE start page to <http://192.168.1.98> and set the web proxy of IE to 192.168.1.98:2080

```
;;;;;;;;;; IE6 Start Page ;;;;;;;;;;
```

```
[HKEY_CURRENT_USER\Software\Microsoft\Internet Explorer\Main]
```

```
"Start Page"="http://192.168.1.98:80"
```

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings]
```

```
"EnableAutodial"=dword:0
```

```
[HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Connections]
```

```
"DefaultConnectionSettings"=hex:3c,00,00,00, 03,00,00,00, 03,00,00,00, 11,00,00,00, 31,39,32,2E, 31,36,38,2E, 31,2E,39,33, 3A,32,30,38, 30,07,00,00, 00,3C,6C,6F, 63,61,6C,3E, 00,00,00,00, 00,00,00,00, 00,00,00,00, 00,00,00,00, 00,00,00,00, 00,00,00,00
```

2) URL Security Zones Policy Settings(For PPLive)

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Internet Settings\Zones\1]
```

```
"CurrentLevel"=dword:10000
```

```
"MinLevel"=dword:10000
```

```
"RecommendedLevel"=dword:10000
```

```
"1200"=dword:00
```

```
"1201"=dword:00
```

```
"1202"=dword:00
```

```
"1400"=dword:00
```

```
"1401"=dword:00
```

```
"1402"=dword:00
```

```
"1405"=dword:00
```

```
"1406"=dword:00
```

```
"1407"=dword:00
```

```
"1601"=dword:00
```

```
"1602"=dword:00
```

```
"1604"=dword:00
```

```
"1605"=dword:00
```

```
"1606"=dword:00
```

```
"1607"=dword:00
```

```
"1608"=dword:00
```

```
"1609"=dword:00
```

```
"1800"=dword:00
```

```
"1802"=dword:00
```

```
"1803"=dword:00
```

```
"1804"=dword:00
```

```
"1805"=dword:00
```

```
"1A00"=dword:00
```

```
"1A02"=dword:00
```

```
"1A03"=dword:00
```

```
"1A04"=dword:00
```

```
"1A05"=dword:00
```

```
"1A06"=dword:00
```

```
"1A10"=dword:00
```

SIGMA DESIGNS®

```
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Internet
Settings\Zones\2]
"CurrentLevel"=dword:10000
"MinLevel"=dword:10000
"RecommendedLevel"=dword:10000
"1200"=dword:00
"1201"=dword:00
"1202"=dword:00
"1400"=dword:00
"1401"=dword:00
"1402"=dword:00
"1405"=dword:00
"1406"=dword:00
"1407"=dword:00
"1601"=dword:00
"1602"=dword:00
"1604"=dword:00
"1605"=dword:00
"1606"=dword:00
"1607"=dword:00
"1608"=dword:00
"1609"=dword:00
"1800"=dword:00
"1802"=dword:00
"1803"=dword:00
"1804"=dword:00
"1805"=dword:00
"1A00"=dword:00
"1A02"=dword:00
"1A03"=dword:00
"1A04"=dword:00
"1A05"=dword:00
"1A06"=dword:00
"1A10"=dword:00
[HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Internet
Settings\Zones\3]
"CurrentLevel"=dword:10000
"MinLevel"=dword:10000
"RecommendedLevel"=dword:10000
"1200"=dword:00
"1201"=dword:00
"1202"=dword:00
"1400"=dword:00
"1401"=dword:00
"1402"=dword:00
"1405"=dword:00
"1406"=dword:00
"1407"=dword:00
"1601"=dword:00
"1602"=dword:00
"1604"=dword:00
"1605"=dword:00
"1606"=dword:00
"1607"=dword:00
"1608"=dword:00
"1609"=dword:00
"1800"=dword:00
"1802"=dword:00
"1803"=dword:00
"1804"=dword:00
"1805"=dword:00
"1A00"=dword:00
"1A02"=dword:00
"1A03"=dword:00
"1A04"=dword:00
"1A05"=dword:00
"1A06"=dword:00
"1A10"=dword:00
```

G)Media Player

For some audio problems of media file,
;060815_KB923828 - Files with large preroll values may cause out-of-memory issues.
[HKEY_CURRENT_USER\Software\Microsoft\Netshow\Player\General]
"Maximum Buffering Time"=dword:2710 ; = 10000 ms
"Buffering Time"=dword:2710 ;=10000 ms

For some BUG of windows CE5.0
;070925_KB942195 - A high priority thread may precipitate starvation in the MP3 decoder pipeline resulting in audio glitches.
[HKEY_LOCAL_MACHINE\Software\Microsoft\DirectX\DirectShow\MP3Decoder]
"MinBufferSize"=dword:4000

;050805_KB904255 - This update allows Windows CE 5.0 to access certain storage media configured with non-standard FAT formatting tools.
[HKEY_LOCAL_MACHINE\System\StorageManager\FATFS]
"BypassFATSectorCheck"=dword:1

;050812_KB902443 - This update resolves some audio performance issues that may occur when playing certain WMV files.
[HKEY_CURRENT_USER\Software\Microsoft\Netshow\Player\General]
"DispatchAdvanceTimeAudioVideo"=dword:3E8 ;=1000

;051026_KB904256 - MP3 files with certain metadata may not play correctly.
[HKEY_CURRENT_USER\Software\Microsoft\Multimedia\DirectShow\MPEG1 Audio]
"ID3v2SizeLimit"=dword:400
[HKEY_CURRENT_USER\Software\Microsoft\Multimedia\DirectShow\MPEG1 Audio]
"FrameSearchLimit"=dword:18

;051028_KB903076 - This update addresses an issue with the stream bandwidth controller and a possible out-of-memory condition when buffering.
[HKEY_CURRENT_USER\Software\Microsoft\NetShow\Player]
"DisconnectTimeout"=dword:1D4C0 ;120000ms

;050117_KB890935 - Playback of large .asf files may result in a memory leak.
[HKEY_CURRENT_USER\Software\Microsoft\NetShow\Player\General]
; "MaxStreamerMessages"=dword:00000064
"MaxStreamerMessages"=dword:00640000

;060927_KB924605 - Virtual memory issues may cause playback to spontaneously abort.
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectShow]
"UsingSharedMemory"=dword:1
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectShow]
"SharedMemoryThreshold"=dword:100000

;060815_KB923828 - Under certain circumstances a deadlock may occur between the ASF file streamer and the splitter.
;This is the low watermark. We will just issue a freeze if we go below multiplier * pre-roll value.
[HKEY_CURRENT_USER\Software\Microsoft\Netshow\Player]
"PrerollMinMultiplier"=dword:1e ; = 30 decimal

;This is a high watermark, meaning that we won't buffer more than what is specified by pre-roll * multiplier.
[HKEY_CURRENT_USER\Software\Microsoft\Netshow\Player]
"PrerollMaxMultiplier"=dword:c8 ; = 200 decimal

;051128_KB910643 - Playback of WMA files may consume large amount of memory.
[HKEY_LOCAL_MACHINE\Software\Microsoft\DirectX\DirectShow\WMADecoder]
"MaxOutputFrameSize"=dword:500000

;050513_KB897325 - The default thread priorities of both the video decoder thread and renderer thread need to be adjusted to properly decoder and display a frame at the right time.
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\DirectShow\ThreadPriority]
"AsyncVideoRenderer"=dword:80

H) Microsoft Network Media Device

FYI Microsoft help information.

I) Standard Shell

1. About Taskbar, add follow setting to project.reg

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Shell\AutoHide]
```

```
@=" " ; ASCII 0x01
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Shell\OnTop]
```

```
@=""
```

2. Let MediaPlayer open the .ts or .VOB extension file as a MPEG file, add follow setting to platform.reg (Not recommend)

```
[HKEY_CLASSES_ROOT\.ts]
```

```
"Content Type"="video/mpeg"
```

```
[HKEY_CLASSES_ROOT\.ts]
```

```
@="videofile"
```

```
"Content Type"="video/mpeg"
```

```
[HKEY_CLASSES_ROOT\MIME\Database\Content Type\video/mpeg]
```

```
"Extension"=".ts"
```

```
;vob
```

```
[HKEY_CLASSES_ROOT\.vob]
```

```
"Content Type"="video/mpeg"
```

```
[HKEY_CLASSES_ROOT\.vob]
```

```
@="videofile"
```

```
"Content Type"="video/mpeg"
```

```
[HKEY_CLASSES_ROOT\MIME\Database\Content Type\video/mpeg]
```

```
"Extension"=".vob"
```

J) DirectDVD

It'll be released.

K) Wire less Ethernet

1. RealTek RTL8180 Native Wi-Fi STA/AP Driver(802.11a/b)
2. MetaLink
3. VIA

8) How to upgrade XOS of SMP8634 chip

This step must work in release mode nk.bin system, don't use other mode nk.bin to do it.

Copy the XOS binary file and xrpc.exe file to "\\Wince500\\PBWorkspace\\(your CE project name)\\RelDir\\SMP863X_MIPSII_Release" directory, download the nk.bin to SMP8634 board boot the CE system up.

In platform builder target control window type "s xrpc Release\\xrpc_xload_xosu-xosMe0-8634_ES4_dev.bin" command.

First time serial window will get a error message, don't care about it, type the command again, then the SMP8634 board will show a green window and shutdown.

Reboot the board, then the new XOS is ready.

In serial port window, the first line message after reboot shows the version of XOS

```
xosMe0 serial#d94c4b9ea50ceb457882f9d5033ec114 subid 0xc4
xenv cs2 ok
power supply: ok
dram0 ok (7)
dram1 ok (8)
zboot (0) ok
```

9) How to get XOS information

```
xrpc_info3
```

10) How to make the boot Screen

A) Copy the SMP863X_RestoreFB.lib(DEV for ES6/ES7/ES9 chip, PROD for REV A/B/C chip) file to "\\Wince500\\Platform\\SMP863X\\lib\\MIPSII\\retail" directory, make sure it is in "bootloader" project link additional libraries

B) Build a release mode CE project and get the NK.bin file

C) Build a new BootFB.exe file with BootFB source code, copy it to CE project release directory

D) Download NK.bin to SMP8634 board with platform builder, run the CE system up

E) In platform builder target control window type "s bootfb.exe \\Release\\bootfb.bin" command, create the bootfb.bin and bootfb.bin.h file in CE project release directory, Detach Device

F) Open the main.c file of bootloader project

1. Add follow define under "//#define RESET_8139_ON_LAUNCH1" line

```
#define SETUP_FRAMEBUFFER_IN_BOOTLOADER 2
```

2. Find follow code

```
#ifdef SETUP_FRAMEBUFFER_IN_BOOTLOADER
// these functions are defined in SMP863x_RestoreFB.lib
int RestoreFB (BYTE *p, DWORD n, DWORD *pPhysicalAddressOfFrameBuffer, DWORD
*pWidth, DWORD *pHeight);
int DVI_Init ();
int DVI_Enable ();
int DVI_Disable ();
```

```
#endif
```

Make sure these code is OK.

3. Find follow code

```
#ifdef SETUP_FRAMEBUFFER_IN_BOOTLOADER
//#include
"C:\\Wince500\\PBWorkspaces\\SMP8634Generic\\RelDir\\SMP863X_MIPSII_Release\\bo
otfb.bin.h"
#endif
```

Modify it to

```
#ifdef SETUP_FRAMEBUFFER_IN_BOOTLOADER
#include "D:\\Wince500\\PBWorkspaces\\(your CE project
name)\\RelDir\\SMP863X_MIPSII_Release\\bootfb.bin.h"
#endif
```

4. Find follow code at the end of "OEMPlatformInit()" function

```
#ifdef SETUP_FRAMEBUFFER_IN_BOOTLOADER
{
    DWORD fb, width, height, i, j, saved_remap_register;
    RMuint8 *p;

    // save the current remap register
    saved_remap_register = INREG32 (cpu_va+CPU_remap+4*4);
    OALLog(L"saved_remap_register = %08lx\r\n", saved_remap_register);
```



```
0x0C000000 // re-program the re-map register so that DRAM1 starts at KSEG1 address
OUTREG32 (cpu_va+CPU_remap+4*4, 0x20000000);
RestoreFB (bootfb, sizeof (bootfb), &fb, &width, &height);
OALLog(L"FB @ 0x%08lx (%d x %d)\r\n", fb, (int)width, (int)height);
if ((fb & 0xF0000000) == 0x20000000)
    fb -= 0x14000000;
// enable HDMI output in "dvi" mode
DVI_Init ();
DVI_Enable ();
// paint the screen gray or draw whatever you want
p = (RMuint8 *)OALPtoUA(fb);
for (i=0; i<height; i++)
{
    for (j=0; j<width; j++)
    {
        p[0] = 0x80; // B
        p[1] = 0x80; // G
        p[2] = 0x80; // R
        p[3] = 0xff; // alpha
        p += 4;
    }
}

// restore the old remap register value
OUTREG32 (cpu_va+CPU_remap+4*4, saved_remap_register);
}
#endif
```

Make sure these code is OK.

5. Run "Built current project" menu command for bootloader project

6. Download new boot.bin file to SMP8634 board and reboot it, about 6 seconds, you can see a grey boot screen from HDMI/YPbPr/Video port.

11) How to show a bitmap logo in boot screen

12) How to show a progress in boot screen

13) How to dynamic change video output ddi_esc

```
#include <windows.h>
#include "../include/ddi_86xx/ddi86xxesc.h"
#include "../include/ddi_86xx/smp863x_formats.h"

static void unicode2char (char *unicode, char *name, int l)
{
    int i;
    for (i=0; i<l; i++)
    {
        name[0] = unicode[0];
        name++;
        unicode += 2;
    }
    name[0] = 0;
}

static int unicodestrln (char *unicode)
{
    int i;
    i = 0;
    while (1)
    {
        if (unicode[0] == 0)
            break;
        unicode += 2;
        i++;
    }
}
```

```
    return i;
}

int WINAPI WinMain (
    HINSTANCE hInstance,
    HINSTANCE hPrevInstance,
    LPTSTR    lpCmdLine,
    int      nCmdShow)
{
    char commandline[512];
    StDDIDisplayOutput DisplayOutput;
    int mode;

    unicode2char ((char *)lpCmdLine, commandline, unicodestrlen ((char *)lpCmdLine));

    mode = atoi (commandline);
    if (mode < 0)
        mode = 0;
    if (mode > 4)
        mode = 4;

    // dynamic resolution change
    if (mode == 0)
    {
        // use when user views on composite/svideo
        DisplayOutput.AnalogStandard = EMhwlibTVStandard_NTSC_M;
        DisplayOutput.ComponentStandard = EMhwlibTVStandard_NTSC_M;
        DisplayOutput.DigitalStandard = EMhwlibTVStandard_HDMI_480i59;
        RETAILMSG (1, (TEXT("*** DYNAMIC OUTPUT MODE CHANGE: Setting display mode
to NTSC, NTSC, 480i\r\n")));
    }
    else
    if (mode == 1)
    {
        // use when user has a 480p capable tv, and they are watching on component or
        hdmi
        DisplayOutput.AnalogStandard = EMhwlibTVStandard_NTSC_M;
        DisplayOutput.ComponentStandard = EMhwlibTVStandard_480p59;
        DisplayOutput.DigitalStandard = EMhwlibTVStandard_HDMI_480p59;
        RETAILMSG (1, (TEXT("*** DYNAMIC OUTPUT MODE CHANGE: Setting display mode
to NTSC, 480P, 480P\r\n")));
    }
    else
    if (mode == 2)
    {
        // use when user has a 720p capable tv, and they are watching on component or
        hdmi
        DisplayOutput.AnalogStandard = EMhwlibTVStandard_NTSC_M;
        DisplayOutput.ComponentStandard = EMhwlibTVStandard_720p59;
        DisplayOutput.DigitalStandard = EMhwlibTVStandard_HDMI_720p59;
        RETAILMSG (1, (TEXT("*** DYNAMIC OUTPUT MODE CHANGE: Setting display mode
to NTSC, 720P, 720P\r\n")));
    }
    else
    if (mode == 3)
    {
        // use when user has a 1080i capable tv, and they are watching on component or
        hdmi
        DisplayOutput.AnalogStandard = EMhwlibTVStandard_NTSC_M;
        DisplayOutput.ComponentStandard = EMhwlibTVStandard_1080i59;
        DisplayOutput.DigitalStandard = EMhwlibTVStandard_HDMI_1080i59;
        RETAILMSG (1, (TEXT("*** DYNAMIC OUTPUT MODE CHANGE: Setting display mode
to NTSC, 1080i, 1080i\r\n")));
    }
    else
    if (mode == 4)
    {
        DisplayOutput.AnalogStandard = EMhwlibTVStandard_NTSC_M ;
    }
}
```

```
DisplayOutput.ComponentStandard = EMhwlibTVStandard_NTSC_M;  
DisplayOutput.DigitalStandard = EMhwlibTVStandard_Custom;
```

```
DisplayOutput.ComponentMode = EMhwlibComponentMode_YUV_SMPTE;  
DisplayOutput.ComponentOrder = EMhwlibComponentOrder_RGB;  
DisplayOutput.ComponentColorSpace = EMhwlibColorSpace_YUV_601;
```

```
DisplayOutput.DigitalOutputVideoMode.PixelClock = 0x2122800;//0x2625A00;  
DisplayOutput.DigitalOutputVideoMode.HActive = 0x320;  
DisplayOutput.DigitalOutputVideoMode.HFrontPorch = 0x0;  
DisplayOutput.DigitalOutputVideoMode.HSyncWidth = 0x80;  
DisplayOutput.DigitalOutputVideoMode.HBackPorch = 0x0;  
DisplayOutput.DigitalOutputVideoMode.HSyncPolarity = 0x0;  
DisplayOutput.DigitalOutputVideoMode.VActive = 0x258;  
DisplayOutput.DigitalOutputVideoMode.VFrontPorch = 0x1;  
DisplayOutput.DigitalOutputVideoMode.VSyncWidth = 0x3;  
DisplayOutput.DigitalOutputVideoMode.VBackPorch = 0x14;  
DisplayOutput.DigitalOutputVideoMode.VSyncPolarity = 0x0;  
DisplayOutput.DigitalOutputVideoMode.Interlaced = 0x0;
```

```
DisplayOutput.LVDS.Enable = 0x1;  
DisplayOutput.LVDS.GPIOFieldIDOutput = 0xB;  
DisplayOutput.LVDS.GPIOPanelOn = 0xE;
```

```
RETAILMSG (1, (TEXT("** DYNAMIC OUTPUT MODE CHANGE: Setting display mode  
to 800X600\r\n")));  
}
```

```
HDC hdc = GetDC (0);  
if (hdc)  
{  
    if(mode < 4)  
    {  
        ExtEscape (  
            hdc,  
            DDI86XX_ESC_SET_DISPLAY_OUTPUT,  
            sizeof (StDDIDisplayOutput),  
            (LPCSTR)&DisplayOutput,  
            0,  
            0  
        );  
    }  
    else  
    {  
        ExtEscape (  
            hdc,  
            DDI86XX_ESC_SET_DISPLAY_OUTPUT_EX,  
            sizeof (StDDIDisplayOutput),  
            (LPCSTR)&DisplayOutput,  
            0,  
            0  
        );  
    }  
    ReleaseDC (0, hdc);  
}  
  
return 0;  
}
```

14) Current Driver escape codes

```
// Driver escape codes
```

```
// Set the display output - Use StDDIDisplayOutput as the input  
#define DDI86XX_ESC_SET_DISPLAY_OUTPUT 100000  
// Get the current display output  
#define DDI86XX_ESC_GET_DISPLAY_OUTPUT 100001  
// Set the HDMI audio info.
```

```
#define DDI86XX_ESC_SET_HDMI_AUDIO_INFO 100002
// Set the flicker
#define DDI86XX_ESC_SET_ANTIFLICKER 100003
// Query the OSD window position. Use StDDIPos
#define DDI86XX_ESC_GET_GDI_POS 100004
// Use StDDIDisableDisplayOutput
#define DDI86XX_ESC_DISABLE_OUTPUTS 100005
// Set the display aspect ratio - Use StDDIOutputAspectRatio
#define DDI86XX_ESC_SET_OUTPUT_ASPECT_RATIO 100006
// Set the macrovision and cgms levels - Use StDDIMacrovisionCGMS
#define DDI86XX_ESC_SET_MACROVISION_CGMS 100007
// Permanently hide/show the hardware cursor
#define DDI86XX_ESC_HIDE_HW_CURSOR 100008
#define DDI86XX_ESC_SHOW_HW_CURSOR 100009
// Brightness, Contrast, and Saturation controls - Use StDDIBCS
#define DDI86XX_ESC_SET_BCS 100013
#define DDI86XX_ESC_GET_BCS 100017
// set the alpha blending value for a region - Use StDDIAlpha
// BE CAREFUL: this function does not do any range or error checking
#define DDI86XX_ESC_SET_ALPHA_VALUE 100018
// check to see if the HDMI has been set at least once.
#define DDI86XX_ESC_CHECK_HDMI_AUDIO_INFO 100019
// Set the display aspect ratio - Use StDDIScalingMode
#define DDI86XX_ESC_SET_SCALING_MODE 100020
// Set/get the key color - Use StDDIKeyColor
#define DDI86XX_ESC_SET_KEYCOLOR 100021
#define DDI86XX_ESC_GET_KEYCOLOR 100022
// set "advanced" analog copy control - Use StDDICopyControl
#define DDI86XX_ESC_SET_COPY_CONTROL 100023
// set/get volume level - Use StDDIVolume
#define DDI86XX_ESC_SET_VOLUME 100024
#define DDI86XX_ESC_GET_VOLUME 100025
// mute/un-mute all audio outputs - Use StDDIAudioMute
#define DDI86XX_ESC_AUDIO_MUTE 100026
// Set the OSD window position. Use StDDIPos
#define DDI86XX_ESC_SET_GDI_POS 100027
// Enable/disable the HDCP. Pass a boolean value to lpszInData of ExtEscape.
#define DDI86XX_ESC_ENABLE_HDCP 100028
// Enable/disable the EDID (HDMI preferred mode) detection. Pass a boolean value to
lpszInData of ExtEscape.
#define DDI86XX_ESC_ENABLE_EDID_DETECTION 100029
// Disable/clear the stretch blitting. Default is enable. Pass a boolean value to lpszInData of
ExtEscape.
#define DDI86XX_ESC_DISABLE_STRETCH_BLT 100030
// Extended version to set the display output - Use StDDIDisplayOutput as
// the input. The application must set all the parameters. The non-extended
// one only requires the standards since the driver will use the current component
// parameters. Note that DDI86XX_ESC_GET_DISPLAY_OUTPUT will return all the
parameters.
#define DDI86XX_ESC_SET_DISPLAY_OUTPUT_EX 100031
// Blt the picture from the decoder onto OSD surface.
#define DDI86XX_ESC_PIC_BITBLT 100032
// Enable/disable the HDMI audio (switch to DVI mode).
// Pass a boolean value to lpszInData of ExtEscape.
#define DDI86XX_ESC_DISABLE_HDMI_AUDIO 100033
// Dump the EDID info. Pass a pre-allocated buffer (char*) to lpszOutData and the buffer size
// to cbOutput of ExtEscape. Recommend the buffer size of at least 4KB.
#define DDI86XX_ESC_DUMP_EDID 100034
// Get the current HDMI connection state. TRUE=connect, FALSE=disconnect. Pass a boolean
value to lpszOutData.
#define DDI86XX_ESC_GET_HDMI_CONNECTION 100035
// DEBUG purpose only - Get the last HDMI debug message.
// Pass a pre-allocated buffer (char*) to lpszOutData and the buffer size
// to cbOutput of ExtEscape. Recommend the buffer size of at least 4KB.
// If the buffer size is less than the valid debug message, it will return 0.
#define DDI86XX_ESC_GET_LAST_HDMI_DEBUGMSG 100036
```

```
// To receive an event when the HDMI connection changes, RegisterWindowMessage the
// below message
// and use the return id as your message id in your application WndProc. Please see
// HWND_BROADCAST
// in PostMessage for more details of how to catch this event.
// When receiving this event, use the DDI86XX_ESC_GET_HDMI_CONNECTION to check for
// current
// HDMI connection.
#define SMP863X_HDMI_CONNECTION_MESSAGE
TEXT("SMP863x_HDMI_CONNECTION_CHANGES")

typedef struct {
    unsigned long PixelClock;
    unsigned long HActive;
    unsigned long HFrontPorch;
    unsigned long HSyncWidth;
    unsigned long HBackPorch;
    unsigned long HSyncPolarity;    // TRUE: positive, FALSE: negative
    unsigned long VActive;
    unsigned long VFrontPorch;
    unsigned long VSyncWidth;
    unsigned long VBackPorch;
    unsigned long VSyncPolarity;    // TRUE: positive, FALSE: negative
    unsigned long Interlaced;
} StDDIVideoMode;

// LVDS specific parameters. The GPIO pin numbers are board dependent, be careful!
typedef struct {
    unsigned long Enable;
    unsigned long GPIOFieldIDOutput; // GPIO to enable Field ID output to the LVDS parity
    line
    unsigned long GPIOPanelOn;    // GPIO to turn on the panel
} StDDILVDS;

typedef struct {
    enum EMhwlibTVStandard AnalogStandard;
    enum EMhwlibTVStandard DigitalStandard;
    enum EMhwlibTVStandard ComponentStandard;

    // The below parameters require to use the DDI86XX_ESC_SET_DISPLAY_OUTPUT_EX.
    // Using the DDI86XX_ESC_SET_DISPLAY_OUTPUT will result an invalid output.

    // These parameters apply to component output
    enum EMhwlibComponentMode ComponentMode;
    enum EMhwlibComponentOrder ComponentOrder;
    enum EMhwlibColorSpace ComponentColorSpace;

    enum EMhwlibColorSpace AnalogColorSpace;
    enum EMhwlibColorSpace DigitalColorSpace;

    // Invert/clear the digital video clock (inverted by default)
    unsigned long InvertClock;

    // Set to 1 to enable VGA output. Also need to set component parameters correctly for
    // the VGA mode.
    // Do not set DigitalStandard to custom when it's 1.
    unsigned long VGAOutput;

    // Custom digital output video mode. DigitalStandard must be set to
    // EMhwlibTVStandard_Custom.
    // Outputs on analog and component will disabled.
    StDDIVideoMode DigitalOutputVideoMode;

    StDDILVDS LVDS;
} StDDIDisplayOutput;

typedef struct {
    unsigned char AdaptativeEnable;
```

```
/** Member default range 0 -> 1 */
unsigned long Taps;
/** Member default range 0 -> 3 */
unsigned long AntiFlickerColor;
/** Member default range 0 -> 3 */
unsigned long AntiFlickerAlpha;
} StDDIAntiFlicker;

typedef struct {
    int x;
    int y;
    int width;
    int height;
} StDDIPos;

typedef struct
{
    int disable;
} StDDIDisableDisplayOutput;

typedef struct
{
    int x;
    int y;
} StDDIOutputAspectRatio;

#define DDI_SCALING_MODE_LETTERBOX    1
#define DDI_SCALING_MODE_PANSCAN    2
#define DDI_SCALING_MODE_NONE        3
typedef struct
{
    int ScalingMode;
} StDDIScalingMode;

typedef struct
{
    int macrovision_level; // range is 0-3 for NTSC, 0-1 for PAL
    int cgms_level;        // range is 0-3
} StDDIMacrovisionCGMS;

typedef struct
{
    // each output has separate controls for brightness, saturation, and contrast control

    int digital_brightness; // range: -128 to 127, default = 0
    int digital_contrast;   // range: 0 to 255, default = 128
    int digital_saturation; // range: 0 to 255, default = 128

    int mainanalog_brightness; // range: -128 to 127, default = 0
    int mainanalog_contrast;   // range: 0 to 255, default = 128
    int mainanalog_saturation; // range: 0 to 255, default = 128

    int component_brightness; // range: -128 to 127, default = 0
    int component_contrast;   // range: 0 to 255, default = 128
    int component_saturation; // range: 0 to 255, default = 128
} StDDIBCS;

typedef struct
{
    unsigned long alpha; // range: 0 - 255, 0=transparent, 0xff=opaque
    int x;                // x offset of region
    int y;                // y offset of region
    int w;                // width of region
    int h;                // height of region
    unsigned long surface_addr; // start address of the surface
    int stride;          // stride of surface
} StDDIAlpha;
```

```
typedef struct
{
    unsigned char r;
    unsigned char g;
    unsigned char b;
} StDDIKeyColor;

typedef struct
{
    enum EMhwlibAGCVersion agc_version;
    unsigned long agc_level; // macrovision pulses 0, 1, 2, 3. Usually is same value as
    aps.

    unsigned long cgmsa; // copy generation management system 0, 1, 2, 3. Sent on
    line 20 and // in "Copy and Redistribution Control Packet" on
    line21_xds.
    unsigned long aps_level; // analog protection system 0, 1, 2, 3. Sent on line 20 and
    // in "Copy and Redistribution Control Packet" on
    line21_xds.
    unsigned long rcd; // redistribution control descriptor. Sent on line 20 and
    // in "Copy and Redistribution Control Packet" on
    line21_xds.
    unsigned long asb; // analog source bit. Sent on line 20 and
    // in "Copy and Redistribution Control Packet" on
    line21_xds.
} StDDICopyControl;

typedef struct
{
    long Volume; // specified in dB, range is (0, -49) 0dB is max, <= -49dB is
    mute
} StDDIVolume;

typedef struct
{
    int mute; // 0=un-mute 1=mute
} StDDIAudioMute;

#include "bitblt.h"

typedef struct
{
    StDecodedInfo SurInfo;
    StPictureParams PicParams;
    StDDIPos SrcWnd;
    StDDIPos DstWnd;
} StDDIPictureBlit;
```

15) Extended Project

All of these extended project are from internet source or 3rd CO. , if you have interest about them, please learn and check them by yourself, there is not any supporting for them with SigmaDesigns.

A) WebCam

Windows CE WebCam Project: This project covers a Windows CE USB video spec webcam driver that supports Windows CE 4.2, 5.0, and 6.0.

License: Microsoft Limited Permissive License (Ms-LPL)

Published: October 12, 2006

Related Links

- [Shared Source Licenses](#)
- [Microsoft Permissive License](#)
- [Microsoft Community License](#)
- [Microsoft Reference License](#)

This license governs use of the accompanying software. If you use the software, you accept this license. If you do not accept the license, do not use the software.

SIGMA DESIGNS[®]

1. Definitions

The terms "reproduce," "reproduction," "derivative works," and "distribution" have the same meaning here as under U.S. copyright law.

A "contribution" is the original software, or any additions or changes to the software.

A "contributor" is any person that distributes its contribution under this license.

"Licensed patents" are a contributor's patent claims that read directly on its contribution.

Top of page

2. Grant of Rights

(A) Copyright Grant- Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free copyright license to reproduce its contribution, prepare derivative works of its contribution, and distribute its contribution or any derivative works that you create.

(B) Patent Grant- Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free license under its licensed patents to make, have made, use, sell, offer for sale, import, and/or otherwise dispose of its contribution in the software or derivative works of the contribution in the software.

Top of page

3. Conditions and Limitations

(A) No Trademark License- This license does not grant you rights to use any contributors' name, logo, or trademarks.

(B) If you bring a patent claim against any contributor over patents that you claim are infringed by the software, your patent license from such contributor to the software ends automatically.

(C) If you distribute any portion of the software, you must retain all copyright, patent, trademark, and attribution notices that are present in the software.

(D) If you distribute any portion of the software in source code form, you may do so only under this license by including a complete copy of this license with your distribution. If you distribute any portion of the software in compiled or object code form, you may only do so under a license that complies with this license.

(E) The software is licensed "as-is." You bear the risk of using it. The contributors give no express warranties, guarantees or conditions. You may have additional consumer rights under your local laws which this license cannot change. To the extent permitted under your local laws, the contributors exclude the implied warranties of merchantability, fitness for a particular purpose and non-infringement.

(F) Platform Limitation- The licenses granted in sections 2(A) & 2(B) extend only to the software or derivative works that you create that run on a Microsoft Windows operating system product.

Link: <http://www.codeplex.com/cewebcam>

B)DOOM

SMP8634 WinCE platform support DirectDraw, so we can try build DOOM windows project on it.

C)PPLive

PPLive build a demo for SMP8634 wince platform

D)EMule

EMule is a windows GPL project, so you can try build a version for SMP8634 platform

E)JavaVM(Personal JAVA)

The pjavawince-1.1-beta1-mips version JavaVM can be installed in SMP8634 platform, so we can try run some JAVA application with it.

16) About Windows CE Test kits(CETK)

- A) Get the real-time CPU/Memory usage
There are two exe files(cetkperf.exe and cpumon.exe.) in WinCE install directory, You can do as follow steps,
 1. Copy the cetkperf.exe(the version for MIPSII) to wince system, run it as command "cetkperf <the name of your winXP PC>", and make sure you can get "*** Could not Connect to Server Socket" message in debug window
 2. Run cpumon.exe file in your winXP system, you can get the real-time CPU and memory usage

17) BUG list

- A) Can't Open Slave IDE device DMA mode
- B) There is noise in audio output when play a MPEG2 ProgramStream with AC3 audio
- C) No audio output when play a WMA pro audio only file
- D) No audio output when play a media file with DTS audio

our SeeYoo On!