



不可能への挑戦

株式会社日昇テクノロジー

低価格、高品質が不可能？

日昇テクノロジーなら可能にする

# ARM/Cortex-A8・TMS320C64x DSP 搭載開発キットマニュアル

株式会社日昇テクノロジー

<http://www.csun.co.jp>

info@csun.co.jp

2010/08/06



[copyright@2010](#)



修正履歴 .....	6
第一章 概要 .....	7
1.1 概要仕様 .....	7
1.2 電気仕様 .....	9
1.3 インタフェース概要 .....	9
第二章 ハードウェア詳細説明 .....	10
2.1 電源入力インタフェース .....	10
2.2 電源出力インタフェース .....	10
2.3 電源 ON/OFF .....	10
2.4 S-VIDEO インタフェース .....	10
2.5 HDMI インタフェース .....	11
2.6 TFT_LCD インタフェース .....	12
2.7 AUDIO OUT インタフェース .....	13
2.8 Camera image sensor インタフェース .....	13
2.9 MIC IN インタフェース .....	14
2.10 キーボードインタフェース .....	15
2.11 シリアルポート .....	15
2.12 LAN インタフェース .....	15
2.13 USB OTG インタフェース .....	16
2.14 USB HOST インタフェース .....	16
2.15 SD/MMC カードインタフェース .....	17
2.16 JTAG インタフェース .....	18
2.17 拡張インタフェース .....	18
2.18 KEY .....	19
2.19 LED .....	20
第三章 Linux の体験 .....	21
3.1 初期時のシステム構成 .....	21
3.2 BSP の特性 .....	22
3.3 システムの起動 .....	22
3.3.1 PC 側でハイパーターミナル使用時の設定 .....	22
3.3.2 Nand Flash から起動 .....	23
3.3.3 SD カードから起動 .....	23
3.4 液晶の設定 .....	23
3.4.1 デフォルトの 4.3 インチ液晶 .....	23
3.4.2 5.6 インチ液晶 .....	23



3.4.3 7インチ液晶 .....	24
3.4.4 DVI .....	24
3.4.5 VGA .....	24
3.5 テストプログラム .....	25
3.5.1 LED テスト .....	25
3.5.2 マトリクスキーボードテスト .....	25
3.5.3 タッチパネルテスト .....	25
3.5.4 RTC テスト .....	26
3.5.5 MMC/SC カードテスト .....	26
3.5.6 USB OTG テスト .....	26
3.5.7 AUDIO テスト .....	28
3.5.8 LAN テスト .....	28
3.5.9 CAMERA テスト .....	29
3.6 Demo .....	29
3.6.1 Angstrom(GPE) .....	29
3.6.2 Google android .....	29
3.6.3 DVSDK .....	30
第四章 Linux の開発 .....	32
4.1 開発環境 .....	32
4.1.1 クロスコンパイルツールをインストールする .....	32
4.1.2 他のツールのインストール .....	32
4.1.3 環境変数の追加 .....	32
4.2 システムのコンパイル .....	32
4.2.1 コンパイルファイルの準備 .....	32
4.2.2 x-loader イメージファイルのコンパイル .....	32
4.2.3 u-boot.bin ファイルのコンパイル .....	33
4.2.4 uImage カーネルファイルのコンパイル .....	33
4.2.5 ファイルシステムファイルのコンパイル .....	34
4.3 システムのコンフィグ .....	34
4.3.1 カーネルコンフィグの変更 .....	34
4.3.2 コンパイルする .....	38
4.3.3 テスト .....	38
第五章 Linux システムイメージの書き込み .....	40
5.1 SD カード起動時システムイメージの更新 .....	40
5.1.1 SD カードの準備 .....	40
5.1.2 イメージファイルの更新 .....	40



5.1.3 u-boot パラメータ設定.....	41
5.2 NAND Flash のシステムイメージの更新.....	41
5.2.1 x-load の更新.....	41
5.2.2 u-boot.bin ファイルの更新.....	41
5.2.3 カーネルの更新.....	42
5.2.4 ファイルシステムの更新.....	43
5.2.5 u-boot パラメータの設定.....	43
6.1 サンプルLED 点灯.....	44
6.1.1 coding.....	44
6.1.2 クロスコンパイル.....	45
6.1.3 ダウンロードして実行する.....	45
第七章 WindowsCE 6.0 概要.....	46
7.1 初期時イメージの紹介.....	46
7.2 BSP の特徴.....	48
7.3 WinCE システムの起動.....	49
7.3.1 操作中、PC 側のハイパーターミナルの設定.....	49
7.3.2 NAND Flash 起動.....	49
7.3.3 SD カードから起動.....	49
7.3.4 テストプログラム.....	49
第八章 WinCE6.0 の開発.....	50
8.1 開発環境の構築.....	50
8.2 コンパイル.....	50
8.2.1 コンパイルファイルの準備.....	50
8.2.2 システムのコンパイル.....	51
8.2.3 システムのコンフィグ.....	52
第九章 WinCE システムの更新.....	53
9.1 SD カードシステムイメージの更新.....	53
9.1.1 SD カードの準備.....	53
9.1.2 イメージの更新.....	53
9.2 NAND Flash システムイメージの更新.....	53
9.2.1 更新ファイルの準備.....	53
9.2.2 イメージの更新.....	53
第十章 WinCE アプリの開発.....	54
10.1 アプリのインタフェースと使用例.....	54
10.1.1 GPIO インタフェースの定義.....	54
10.1.2 使用例.....	54



---

10.2 インタフェースアプリ開発例 .....	55
10.2.1 新しいプロジェクトの作成 .....	55
10.2.2 ソースコード .....	56
10.2.3 コンパイルと実行 .....	56
付録一 Linux USB Ethernet/RNDIS Gadget ドライバのインストール .....	57
付録二 Linux Boot Disk Format .....	60
付録三 TFTP サーバーの構築 .....	64
付録四 WinCE 関連リソースの URL .....	66
付録五 ボードの寸法図 .....	67
付録六 各デバイスの接続 I/F .....	68

※ 使用されたソースコードは<http://www.csun.co.jp/>からダウンロードできます。



## 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2010/6/4
2	Ver1.1	Linux カーネルバージョンアップ	2010/8/6

## 第一章 概要

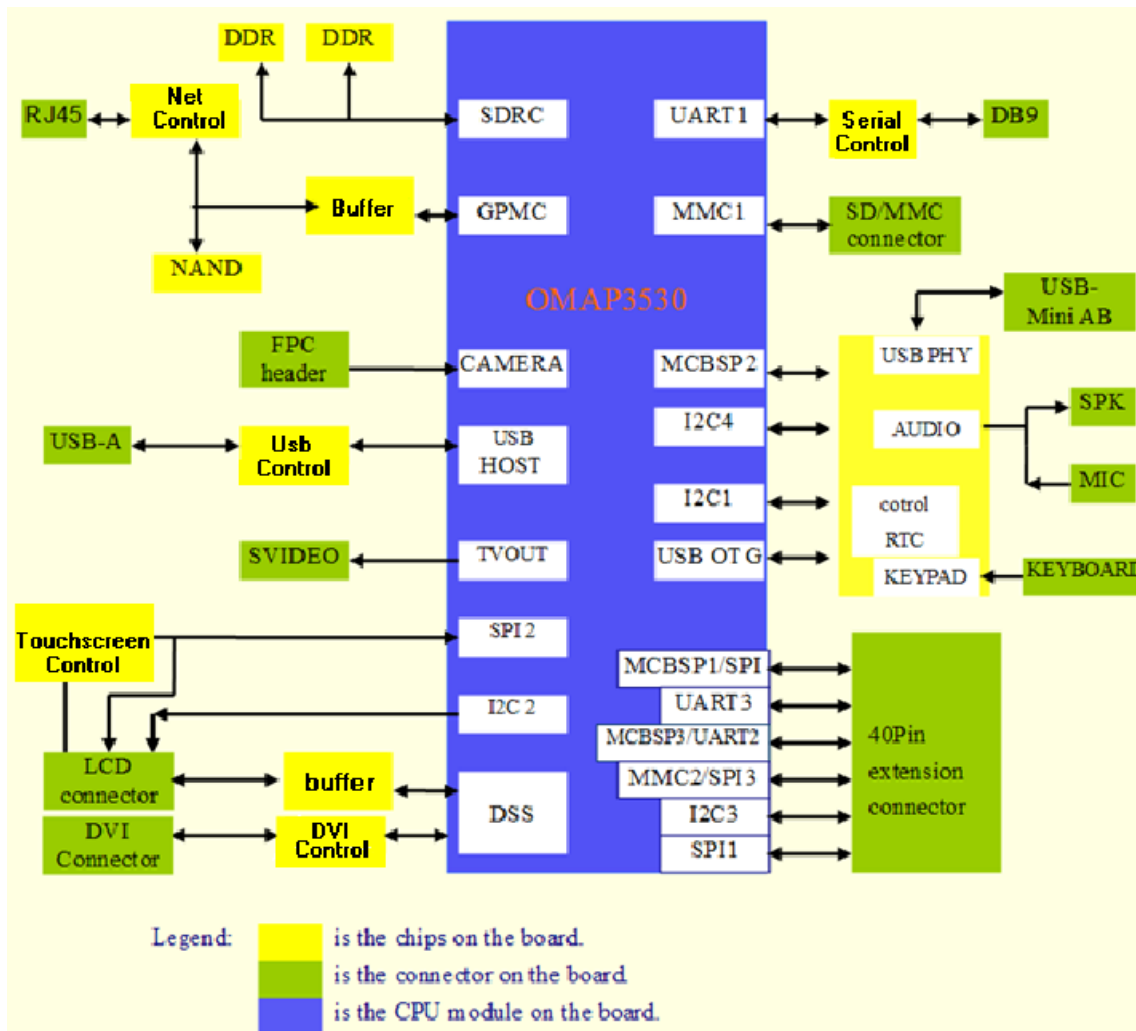
ARM コアプロセッサ-Cortex-A8 を採用した TI 社の OMAP3530、周波数 600MHz、412MHz TMS320C64x DSP。

標準外付け: Ethernet、S-VIDEO、AUDIO IN/OUT、USB Host、USB Device、SD/MMC COM、SPI、I2C、JTAG、CAMERA、タッチパネル付き TFT、PS/2、バス、HDMI など。

開発環境としては linux-2.6.28 及び WinCE6.0 を提供している。各ドライバーソースも提供している。その上、google android 及び angstrom(GPE)のリリース版のデモを提供している。DVI 出力は 720P の表示レベルで、OMAP3530 のパワーフルな計算力が楽しめる。

- ・HDMI: High Definition Multimedia Interface
- ・DVI: Digital Visual Interface
- ・GPMC バス: General-Purpose Memory Controller、OMAP3530 のシステムバス

### 1.1 概要仕様





## CPU:

- OMAP3530
- 600-MHz ARM Cortex™-A8 Core
- 430-MHz TMS320C64x™ DSP Core
- 16kB I-Cache, 16kB D-Cache, 256kB L2 メモリ
- 64kB SRAM, 112kB ROM

## メモリ:

- 128MByte 32bit DDR SDRAM, 166MHz
- 128MByte 16 bit NAND Flash

## Audio/Video インタフェース:

- 4ライン S-VIDEO I/F x 1
- HDMI I/F x 1
- Audio In I/F x 1
- ダブルチャンネル Audio Out I/F x 1

## タッチパネル付き TFT

- 解像度: 480 (W) x 272 (H) dots
- RGB: 391680 colors
- Brightness: Typical 350 cd/m<sup>2</sup> (min 300 cd/m<sup>2</sup>)
- 4ラインタッチパネル

## 通信インタフェース:

- シリアルポート:
  - 3 線シリアルポート x 1
  - 5 線シリアルポート x 1
- USB I/F:
  - USB2.0 OTG x 1、High-speed、480Mbps
  - USB2.0 HOST x 1、High-speed、480Mbps
- SD/MMC I/F:
  - SD/MMC x 1、3.3V 及び 1.8V のロジック電圧をサポート
  - SD/MMC x 1、1.8V のロジック電圧をサポート
- Ethernet I/F: 10/100Mbps, RJ45 connector
- McSPI I/F x 1、多チャンネル SPI
- McBSP I/F x 1、多機能シリアル I/F
- I2C x 1
- HDQ x 1(HDQ/1-Wire)

## 入カインタフェース:



- ・CAMERA I/F x 1、CCD 或いは CMOS カメラを接続用
- ・6x6 マトリクスキーボード
- ・14ピン JTAG I/F x 1
- ・Boot キー x 1
- ・Reset キー x 1
- ・ユーザーキー x 1
- ・ON/OFF キー x 1

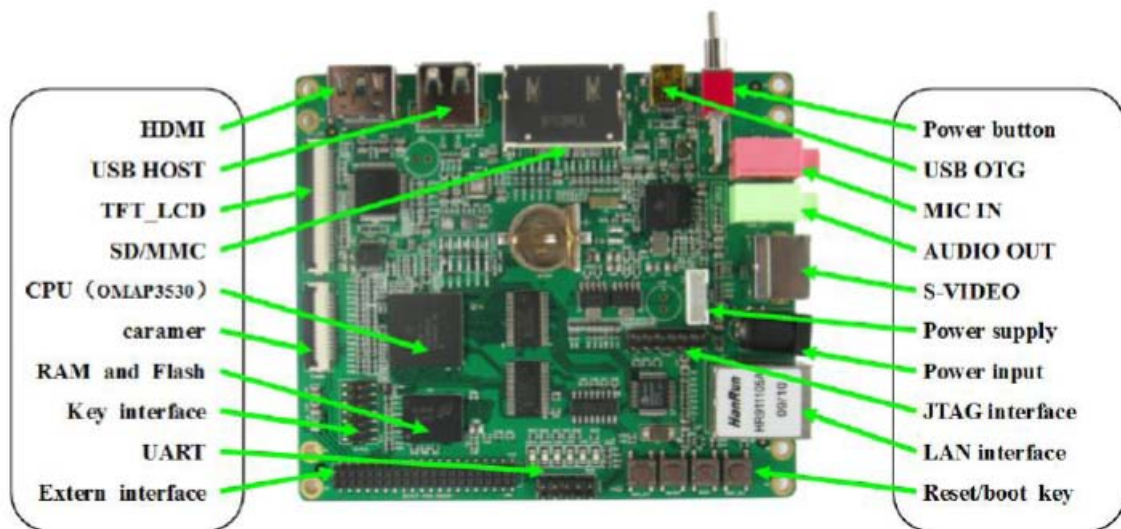
## LED:

- ・3.3v 電源指示 LED(LED33) x 1
- ・5v 電源指示 LED(LED50) x 1
- ・ユーザーLED x 1
- ・システム用 LED x 3(LED1、LED2、LED3)、ユーザーLED としても利用可

## 1.2 電気仕様


- ・外形寸法: 110 mm x 95 mm ※突起物は除く
- ・Input Voltage: +5V
- ・Power Consumption: 0.5A @ 5V
- ・操作温度: 0 `C ~ 70 `C
- ・操作湿度: 20% ~ 90%

## 1.3 インタフェース概要




## 第二章 ハードウェア詳細説明

### 2.1 電源入力インタフェース

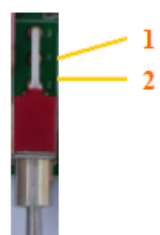
Pin	Signal	Function	Pin out
1	GND	Power input (+5V)	
2	+5V	Power supply (+5V) 2A (Type)	

### 2.2 電源出力インタフェース

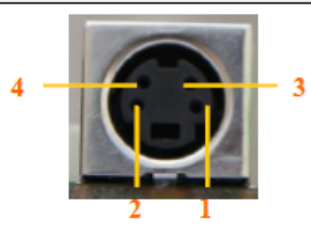
外部デバイスに 5V, 4.2V, 3.3V の電源を提供する。

Pin	Signal	Function	Pin out
1	VDD50	5V output	
2	VDD42	4.2V output	
3	VDD33	3.3V output	
4	ADCIN	ADC input	
5	GND	GND	


### 2.3 電源 ON/OFF

Pin	Signal	Function	Pin out
1	DC IN	VDD Input	
2	VDD50	+5V	
3	NC	NC	


### 2.4 S-VIDEO インタフェース

Pin	Signal	Function	Pin out
1	GND	GND	
2	GND	GND	
3	OUTPUT1	VIDEO Y	
4	OUTPUT2	VIDEO C	

## 2.5 HDMI インタフェース

Pin	Signal	Function	Pin out
1	DAT2+	TMDS data 2+	
2	DAT2_S	TMDS data 2 shield	
3	DAT2-	TMDS data 2-	
4	DAT1+	TMDS data 1+	
5	DAT1_S	TMDS data 1 shield	
6	DAT1-	TMDS data 1-	
7	DAT0+	TMDS data 0+	
8	DAT0_S	TMDS data 0 shield	
9	DAT0-	TMDS data 0-	
10	CLK+	TMDS data clock+	
11	CLK_S	TMDS data clock shield	
12	CLK-	TMDS data clock-	
13	CEC	Consumer Electronics Control	
14	NC	NC	
15	SCL	IIC master serial clock	
16	SDA	IIC serial bidirectional data	
17	GND	GND	
18	5V	5V	
19	HPLG	Hot plug and play detect	

## 2.6 TFT\_LCD インタフェース

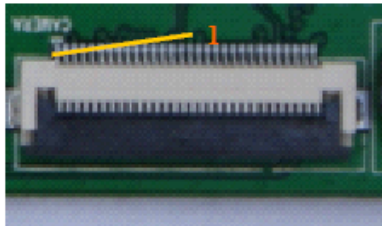
Pin	Signal	Function	Pin out
1	DSS_D0	LCD Pixel data bit 0	
2	DSS_D1	LCD Pixel data bit 1	
3	DSS_D2	LCD Pixel data bit 2	
4	DSS_D3	LCD Pixel data bit 3	
5	DSS_D4	LCD Pixel data bit 4	
6	DSS_D5	LCD Pixel data bit 5	
7	DSS_D6	LCD Pixel data bit 6	
8	DSS_D7	LCD Pixel data bit 7	
9	GND	GND	
10	DSS_D8	LCD Pixel data bit 8	
11	DSS_D9	LCD Pixel data bit 9	
12	DSS_D10	LCD Pixel data bit 10	
13	DSS_D11	LCD Pixel data bit 11	
14	DSS_D12	LCD Pixel data bit 12	
15	DSS_D13	LCD Pixel data bit 13	
16	DSS_D 14	LCD Pixel data bit 14	
17	DSS_D15	LCD Pixel data bit 15	
18	GND	GND	
19	DSS_D16	LCD Pixel data bit 16	
20	DSS_D17	LCD Pixel data bit 17	
21	DSS_D18	LCD Pixel data bit 18	
22	DSS_D19	LCD Pixel data bit 19	
23	DSS_D20	LCD Pixel data bit 20	
24	DSS_D21	LCD Pixel data bit 21	
25	DSS_D22	LCD Pixel data bit 22	
26	DSS_D23	LCD Pixel data bit 23	
27	GND	GND	
28	DEN	AC bias control (STN) or pixel data enable (TFT)	
29	HSYNC	LCD Horizontal Synchronization	
30	VSYNC	LCD Vertical Synchronization	
31	GND	GND	

32	CLK	LCD Pixel Clock
33	GND	GND
34	X+	X+ Position Input
35	X-	X- Position Input
36	Y+	Y+ Position Input
37	Y-	Y- Position Input
38	SPI_CLK	SPI clock
39	SPI_MOSI	Slave data in, master data out
40	SPI_MISO	Slave data out, master data in
41	SPI_CS	SPI enable
42	IIC_CLK	IIC master serial clock
43	IIC_SDA	IIC serial bidirectional data
44	GND	GND
45	VDD18	1.8V
46	VDD33	3.3V
47	VDD50	5V
48	VDD50	5V
49	RESET	Reset
50	PWREN	Power on enable

## 2.7 AUDIO OUT インタフェース


Pin	Signal	Function	Pin out
1	GND	GND	
2	NC	NC	
3	Right	Right output	
4	NC	NC	
5	Left	Left output	

## 2.8 Camera image sensor インタフェース

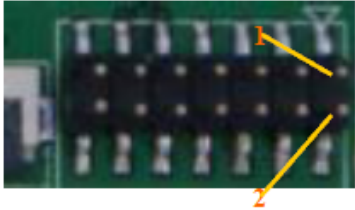
Pin	Signal	Function	Pin out
1	GND	GND	
2	D0	Digital image data bit 0	
3	D1	Digital image data bit 1	
4	D2	Digital image data bit 2	
5	D3	Digital image data bit 3	
6	D4	Digital image data bit 4	

7	D5	Digital image data bit 5
8	D6	Digital image data bit 6
9	D7	Digital image data bit 7
10	D8	Digital image data bit 8
11	D9	Digital image data bit 9
12	D10	Digital image data bit 10
13	D11	Digital image data bit 11
14	GND	GND
15	PCLK	Pixel clock
16	GND	GND
17	HS	Horizontal synchronization
18	VDD50	5V
19	VS	Vertical synchronization
20	VDD33	3.3V
21	XCLKA	Clock output a
22	XCLKB	Clock output b
23	GND	GND
24	FLD	Field identification
25	WEN	Write Enable
26	STROBE	Flash strobe control signal
27	SDA	IIC master serial clock
28	SCL	IIC serial bidirectional data
29	GND	GND
30	VDD18	1.8V

### 2.9 MIC IN インタフェース

Pin	Signal	Function	Pin out
1	GND	GND	
2	NC	NC	
3	MIC MAIN P	Right input	
4	NC	NC	
5	MIC MAIN N	Left input	

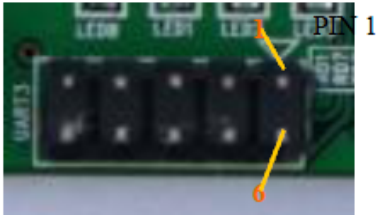
## 2.10 キーボードインタフェース

Pin	Signal	Function	Pin out
1	KC0	Keypad matrix column 0 output	
2	KR0	Keypad matrix row 0 input	
3	KC1	Keypad matrix column 1 output	
4	KR1	Keypad matrix row 1 input	
5	KC2	Keypad matrix column 2 output	
6	KR2	Keypad matrix row 2 input	
7	KC3	Keypad matrix column 3 output	
8	KR3	Keypad matrix row 3 input	
9	KC4	Keypad matrix column 4 output	
10	KR4	Keypad matrix row 4 input	
11	KC5	Keypad matrix column 5 output	
12	KR5	Keypad matrix row 5 input	
13	VDD18	1.8V	
14	GND	GND	

## 2.11 シリアルポート

### 3 ラインシリアルポートインタフェース


- Signal input/output level: RS232C
- Maximum data rate: 115.2kbps
- Flow Control: None
- Connector: 10-pin (5 × 2) 2.54mm pitch connector

Pin	Signal	Function	Pin out
1	NC	NC	
2	TXD	Transit data	
3	RXD	Receive data	
4	NC	NC	
5	GND	GND	
6	NC	NC	
7	NC	NC	
8	NC	NC	
9	NC	NC	

## 2.12 LAN インタフェース

DM9000 に 10/100M Ethernet モジュールが含まれている。

本ボードはストレートケーブル経由で hub に接続、或いはクロスケーブル経由で PC と直接接続できる。


Pin	Signal	Function	Pin out
1	TX+	TX+ output	
2	TX-	TX- output	
3	RX+	RX+ input	
4	VDD25	2.5V Power for TX/RX	
5	VDD25	2.5V Power for TX/RX	
6	RX-	RX- input	
7	NC	NC	
8	NC	NC	
9	VDD	3.3V Power for LED	
10	LED1	Speed LED	
11	LED2	Link LED	
12	VDD	3.3V Power for LED	

### 2.13 USB OTG インタフェース

Mini USB A/B インタフェース。直接 TPS65930 マイコンの USB OTG ポートと接続する。

Data Transfer Mode: USB2.0 High Speed (480Mbps)

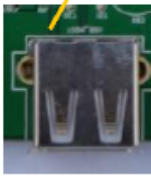
Power Supply: Voltage: +5V

Pin	Signal	Function	Pin out
1	VBUS	+5V	
2	DN	USB Data-	
3	DP	USB Data+	
4	ID	USB ID	
5	GND	GND	

### 2.14 USB HOST インタフェース

Data Transfer Mode: USB2.0 High Speed (480Mbps)

Power Supply: Voltage: +5V


Pin	Signal	Function	Pin out
1	VBUS	+5V	
2	DN	USB Data-	
3	DP	USB Data+	
4	GND	GND	



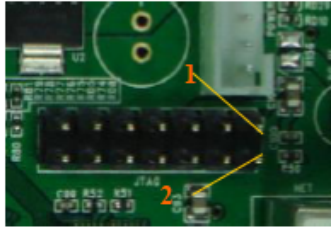
### 2.15 SD/MMC カードインタフェース

挿入・保護の自動検査機能付き。4bit/8bit の SD カードをサポートする。

1.8V/3.3V ロジック電圧をサポートする。


Pin	Signal	Function	Pin out
1	MINISD_CD1	Mini SD Card detect 1	
2	MINISD_CD2	Mini SD Card detect 2	
3	DAT2	MMC card data 2	
4	DAT3	MMC card data 3	
5	DAT4	MMC card data 4	
6	MINISD_DAT2	Mini SD card data 2	
7	CMD	SD/MMC Command Signal	
8	MINISD_DAT3	Mini SD card data 3	
9	DAT5	MMC card data 5	
10	MINISD_CMD	Mini SD card command	
11	VSS	GND	
12	MINISD_VSS	GND	
13	NC	NC	
14	VDD	VDD	
15	NC	NC	
16	MINISD_VDD	VDD	
17	CLK	MMC card clock	
18	MINISD_CLK	Mini SD card clock	
19	DAT6	MMC card data 6	
20	MINISD_VSS	GND	
21	VSS	GND	
22	MINISD_DAT0	Mini SD card data 0	
23	DAT7	MMC card data 7	
24	MINISD_DAT1	Mini SD card data 1	
25	DAT0	MMC card data 0	
26	DAT1	MMC card data 1	
27	SD_CD	SD Card detect	
28	SD_WP	SD write protect	
29	GND	GND	
30	GND	GND	

## 2.16 JTAG インタフェース

Pin	Signal	Function	Pin out
1	TMS	Test mode select	
2	NTRST	Test system reset	
3	TDI	Test data input	
4	GND	GND	
5	VIO	1.8V	
6	NC	NC	
7	TDO	Test data output	
8	GND	GND	
9	RTCK	Receive test clock	
10	GND	GND	
11	TCK	Test clock	
12	GND	GND	
13	EMU0	Test emulation 0	
14	EMU1	Test emulation 1	

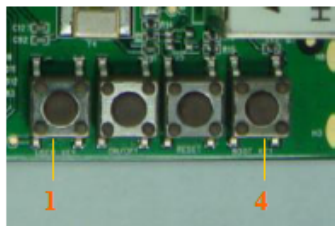
## 2.17 拡張インタフェース

BSP1, BSP3, UART1, I2C3, SPI1, MMC2 インタフェースなど、ロジック電圧は 1.8V。

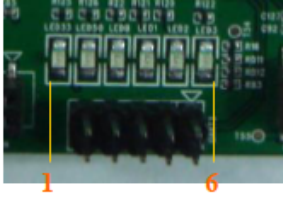
Pin	Signal	Function	Pin out
1	GND	GND	
2	BSP1_DX	Transmitted serial data 1	
3	BSP1_DR	Received serial data 1	
4	BSP1_CLKR	Received clock 1	
5	BSP1_FSX	Transmit frame synchronization 1	
6	BSP1_CLKX	Transmit clock 1	
7	BSP1_CLKS	External clock input 1	
8	BSP1_FSR	Receive frame synchronization 1	
9	UART1_CTS	UART1 clear to send	
10	UART1_RTS	UART1 request to send	
11	UART1_RX	UART1 receive data	
12	UART1_TX	UART1 transmit data	
13	GND	GND	
14	MMC2_CLK	MMC2 card clock	
15	MMC2_CMD	MMC2 Command Signal	
16	MMC2_D0	MMC2 card data 0	
17	MMC2_D1	MMC2 card data 1	
18	MMC2_D2	MMC2 card data 2	

19	MMC2_D3	MMC2 card data 3
20	MMC2_D4	MMC2 card data 4
21	MMC2_D5	MMC2 card data 5
22	MMC2_D6	MMC2 card data 6
23	MMC2_D7	MMC2 card data 7
24	BSP3_DX	Transmitted serial data 3
25	BSP3_DR	Received serial data 3
26	BSP3_CLKX	Transmit clock 3
27	BSP3_FSX	Transmit frame synchronization 3
28	GND	GND
29	IIC3_SCL	IIC3 master serial clock
30	IIC3_SDA	IIC3 serial bidirectional data
31	SPI1_SIMO	Slave data in, master data out
32	SPI1_SOMI	Slave data out, master data in
33	SPI1_CLK	SPI1 clock
34	SPI1_CS0	SPI enable 0
35	SPI1_CS3	SPI enable 3
36	HDQ_SIO	Bidirectional HDQ
37	VDD33	3.3V
38	VDD18	1.8V
39	VDD50	5V
40	GND	GND

## 2.18 KEY

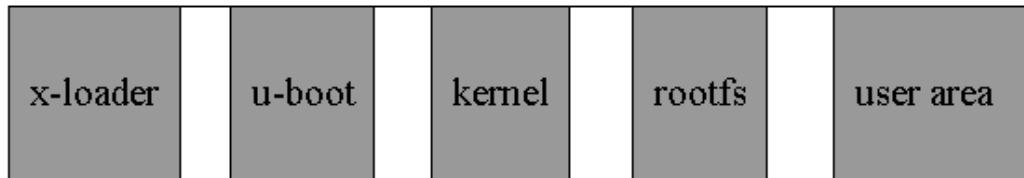
Pin	Signal	Function	Pin out
1	USER-KEY	User-defined key	
2	ON/OFF	System ON/OFF key	
3	RESET	System reset key	
4	BOOT-KEY	System boot configuration	

## 2.19 LED

Pin	Signal	Function	Pin out
1	LED33	3.3V Power led	
2	LED50	4.2V Power led	
3	LEDB	User LED	
4	LED1	User LED	
5	LED2	User LED	
6	LED3	User LED	

## 第三章 Linux の体験

### 3.1 初期時のシステム構成



x-loader------(x-load.bin.ift\_for\_NAND)

u-boot------(flash-uboot.bin)

2.6 kernel------(uImage)

rootfs------(ubi.img)

### 3.2 BSP の特性

Item		Note
BIOS	x-loader	NAND / ONENAND
		MMC/SD
		FAT
	u-boot	NAND / ONENAND
		MMC/SD
		FAT
		NET
Kernel	Linux-2.6.x	Supports ROM/CRAM/EXT2/EXT3/FAT/NFS/JFFS2/UBIFS and various file systems
Device Driver	serial	Series driver
	rtc	Hardware clock driver
	net	10/100M Ethernet card DM8000 driver
	flash	nand flash driver (supports nand boot)
	lcd	TFT LCD driver
	touch screen	Touch screen controller ads7846 driver
	mmc/sd	mmc/sd controller driver
	usb otg	usb otg 2.0 driver (can be configured as master/slave device)
	usb ehci	usb ehci driver
	dvi	Supports dvi-d signal output
	s-video	Supports s-video signal output
	Audio	Audio driver
	camera	Camera driver
	keypad	6x6 matrix keyboard driver
led	User led lamp driver	
GUI	Angstrom	release version for embedded devices' desktop environment
	Android	google android system

### 3.3 システムの起動

#### 3.3.1 PC 側でハイパーターミナル使用時の設定

- Baud rate: 115200
- Data bit: 8



- Parity check: no
- Stop bit: 1
- Flow control: no

### 3.3.2 Nand Flash から起動

ボードのデフォルト設定はNANDから起動する。NANDにシステムイメージファイルがなければ、SDカードから起動する。初期時のユーザー名は「root」で、パスワードは設定していない。

### 3.3.3 SD カードから起動

SD カードから起動したい場合はボードの「Boot key」を押しながら、起動する。初期時のユーザー名は「root」で、パスワードは設定していない。

## 3.4 液晶の設定

### 3.4.1 デフォルトの 4.3 インチ液晶

u-bootのパラメータの設定：電源を投入した後、3秒以内、パソコンのスペースキーを押すと、u-boot モードに入る。

#### ・ NAND 起動：

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:4.3inch_LCD
```

```
OMAP3 DevKit8000 # setenv bootcmd nand read.i 80300000 280000 300000¥;bootm 80300000
```

```
OMAP3 DevKit8000 # saveenv
```

#### ・ SD カード起動：

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram
initrd=0x81600000,40M video=omapfb:mode:4.3inch_LCD
```

```
OMAP3 DevKit8000 # setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc
0 81600000 ramdisk.gz;bootm 80300000'
```

```
OMAP3 DevKit8000 # saveenv
```

### 3.4.2 5.6 インチ液晶

#### ・ NAND 起動：

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:5.6inch_LCD
```

```
OMAP3 DevKit8000 # setenv bootcmd nand read.i 80300000 280000 300000¥;bootm 80300000
```

```
OMAP3 DevKit8000 # saveenv
```

#### ・ SD カード起動：



```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram
initrd=0x81600000,40M video=omapfb:mode:5.6inch_LCD
OMAP3 DevKit8000 # setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc
0 81600000 ramdisk.gz;bootm 80300000'
OMAP3 DevKit8000 # saveenv
```

### 3.4.3 7インチ液晶

#### ・ NAND 起動 :

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:7inch_LCD
```

```
OMAP3 DevKit8000 # setenv bootcmd nand read. i 80300000 280000 300000¥;bootm 80300000
```

```
OMAP3 DevKit8000 # saveenv
```

#### ・ SD カード起動 :

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram
initrd=0x81600000,40M video=omapfb:mode:7inch_LCD
```

```
OMAP3 DevKit8000 #setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc
0 81600000 ramdisk.gz;bootm 80300000'
```

```
OMAP3 DevKit8000 # saveenv
```

### 3.4.4 DVI

#### ・ NAND 起動 :

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:720p60
```

```
OMAP3 DevKit8000 # setenv bootcmd nand read. i 80300000 280000 300000¥;bootm 80300000
```

```
OMAP3 DevKit8000 # saveenv
```

#### ・ SD カード起動 :

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram
initrd=0x81600000,40M video=omapfb:mode:720p60
```

```
OMAP3 DevKit8000 #setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc
0 81600000 ramdisk.gz;bootm 80300000'
```

```
OMAP3 DevKit8000 # saveenv
```

### 3.4.5 VGA

#### ・ NAND 起動 :

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:VGA
```





```
OMAP3 DevKit8000 # setenv bootcmd nand read. i 80300000 280000 300000¥;bootm 80300000
```

```
OMAP3 DevKit8000 # saveenv
```

・SD カード起動：

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram  
initrd=0x81600000,40M video=omapfb:mode:VGA
```

```
OMAP3 DevKit8000 #setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc  
0 81600000 ramdisk.gz;bootm 80300000'
```

```
OMAP3 DevKit8000 # saveenv
```

### 3.5 テストプログラム

#### 3.5.1 LED テスト

ボード上のLED1, LED2, LED3はユーザーLEDですが、LED1は既にシステムtickとして、LED2はSDカードのアクセス指示用で利用されている。

下記はLED3のテスト(PCのハイパーターミナルで下記コマンドを入力する)：

点灯：

```
root@DevKit8000:~# echo -n 1 >/sys/class/leds/led3/brightness
```

消灯：

```
root@DevKit8000:~# echo -n 0 >/sys/class/leds/led3/brightness
```

#### 3.5.2 マトリクスキーボードテスト

本ボードは 6x6 マトリクスキーボードを拡張している。Evtest ツールで動作確認出来る。

マトリクスキーボードと本ボードを接続して、ハイパーターミナルから下記コマンドを入力する。

```
root@DevKit8000:~# evtest /dev/input/event0
```

キーボードから任意のキーを押す。例えば '1'、下記情報が出て来る：

```
Event: time 946684837.310027, type 1 (Key), code 2 (1), value 1
```

```
Event: time 946684837.402160, type 1 (Key), code 2 (1), value 0
```

“type 1 (Key), code 2 (1), value 1”、は何かイベントが発生し、key value が 2(keyboard の '1')、状態は押下( '0' は離す)。

※CONTROL+Cでテストプログラム終了。

#### 3.5.3 タッチパネルテスト

(1) 下記コマンドでタッチパネルを校正：

```
root@DevKit8000:~# ts_calibrate
```

画面上の提示通り“+”を5回クリックして完成する。

(2) 校正完了した後、下記コマンドでタッチパネルをテストできる：

```
root@DevKit8000:~# ts_test
```

※CONTROL+Cでテストプログラム終了。

### 3.5.4 RTC テスト

(1)RTC 時間の設定:

```
root@DevKit8000:~# date 080820002008
```

```
Fri Aug 8 20:00:00 UTC 2008
```

(2)システム時間をRTC に書き込む:

```
root@DevKit8000:~# hwclock -w
```

(3)RTC 時間を読む

```
root@DevKit8000:~# hwclock
```

(4)起動の時、RTC 時間をシステム時間に恢復します。

```
root@DevKit8000:~# hwclock -s
```

```
root@DevKit8000:~# date
```

```
Fri Aug 8 20:01:45 UTC 2008
```

### 3.5.5 MMC/SC カードテスト

MMC/SC カードを挿入すると、自動的に/media にマウントされデフォルトの名前は mmcblk0p1。

```
root@DevKit8000:~# cd /media/
```

```
root@DevKit8000:/media# ls
```

```
card hdd mmcblk0p1 ram union cf mmc1 net realroot
```

```
root@DevKit8000:/media# ls mmcblk0p1/
```

```
flash-uboot.bin  u-boot.bin  x-load.bin.ift_for_NAND  mlo  uImage  ramdisk.gz  ubi.img
```

```
root@DevKit8000:/media# umount /media/mmcblk0p1/
```

※アンマウントしなくて SD カードを拔出すと、SD カードを壊す恐れがある。

### 3.5.6 USB OTG テスト

(1)デバイスとして利用する場合:

ア USB miniB to USB Aケーブルでパソコンとボードを接続する(USB miniBをボード側に)。USBドライバは弊社サーバーのlinux¥toolsを参照ください。

イ USB ドライバをインストールした後、PC 側は仮想 LAN が一つ増えている。

#### LAN or High-Speed Internet

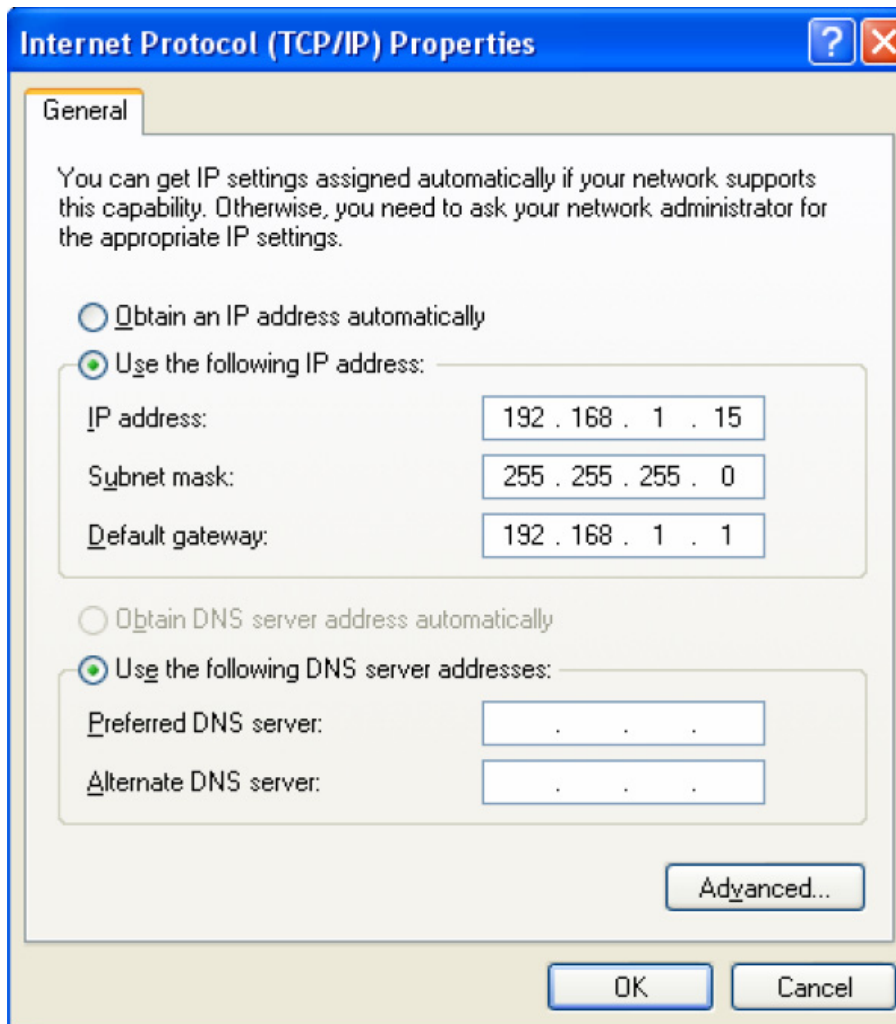


Local Area Connection 4



Local Area Connection

ウ この仮想 LAN の IP アドレスを次の画面のように設定する。



エ ボードの IP と usb 仮想 LAN の IP アドレスを同じネットワークセグメントに設定する。

```
root@DevKit8000:~# ifconfig usb0 192.168.1.115
```

```
root@DevKit8000:~# ifconfig
```

```
lo Link encap:Local Loopback
```

```
inet addr:127.0.0.1 Mask:255.0.0.0
```

```
UP LOOPBACK RUNNING MTU:16436 Metric:1
```

```
RX packets:26 errors:0 dropped:0 overruns:0 frame:0
```

```
TX packets:26 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:0 RX bytes:2316  
(2.2 KiB) TX bytes:2316 (2.2 KiB)
```

```
usb0 Link encap:Ethernet HWaddr 5E:C5:F6:D4:2B:91
```

```
inet addr:192.168.1.115 Bcast:192.168.1.255 Mask:255.255.255.0
```

```
UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
```

```
RX packets:253 errors:0 dropped:0 overruns:0 frame:0
```

```
TX packets:43 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX
```



```
bytes:35277 (34.4 KiB) TX bytes:10152 (9.9 KiB)
```

オ テスト:

```
root@DevKit8000:~# ping 192.168.1.15
```

```
PING 192.168.1.15 (192.168.1.15): 56 data bytes
```

```
64 bytes from 192.168.1.15: seq=0 ttl=128 time=0.885 ms
```

```
64 bytes from 192.168.1.15: seq=1 ttl=128 time=0.550 ms
```

※CONTROL+Cでテストプログラム終了。

(2)HOST として利用する場合:

USB miniA to USB A ケーブルでボードと USB デバイス(例 Udisc)を接続する(USB miniA をボード側に)。

ボードを起動すると、自動的に USB デバイスを/media にマウントする。

```
root@DevKit8000:~# cd /media/
```

```
root@DevKit8000:/media# ls
```

```
card hdd mmcblk0p1 ram sda1 cf mmc1 net realroot union
```

```
root@DevKit8000:/media# cd sda1
```

sda1 フォルダの下にファイルがあれば、正しくアクセス出来ている。

※Udisc の種類によって、sda に認識される場合がある。

### 3.5.7 AUDIO テスト

(1)Record

MIC を挿入して、下記コマンドで録音できる。

```
root@DevKit8000:~# arecord -t wav -c 2 -r 44100 -f S16_LE -v k
```

(2)Play

イヤホンを接続して、下記コマンドで録音した内容が聞こえる。

```
root@DevKit8000:~# aplay -t wav -c 2 -r 44100 -f S16_LE -v k
```

※CONTROL+Cでテストプログラム終了。

### 3.5.8 LAN テスト

本ボードは DM9000 搭載していて、ローカルネットワークに接続できる。

LAN ケーブルを接続して、下記コマンドを入力する。

```
root@DevKit8000:~# ifconfig eth0 192.192.192.200
```

```
eth0: link down
```

```
eth0: link up, 100Mbps, full-duplex, lpa 0x41E1
```

```
root@DevKit8000:~# ping 192.192.192.90
```

```
PING 192.192.192.90 (192.192.192.90): 56 data bytes
```



```
64 bytes from 192.192.192.90: seq=0 ttl=128 time=1.007 ms
```

```
64 bytes from 192.192.192.90: seq=1 ttl=128 time=0.306 ms
```

```
64 bytes from 192.192.192.90: seq=2 ttl=128 time=0.397 ms
```

```
64 bytes from 192.192.192.90: seq=3 ttl=128 time=0.367 ms
```

```
--- 192.192.192.90 ping statistics --- 4 packets transmitted, 4 packets received, 0% packet loss  
round-trip min/avg/max = 0.306/0.519/1.007 ms
```

### 3.5.9 CAMERA テスト

カメラモジュールを挿入して、下記コマンドを入力する。

```
root@DevKit8000:~# saMmapLoopback
```

液晶にカメラから撮取した画像が見える。

### 3.6 Demo

次の操作はパソコンのLinux 環境(ubuntu 9.04)です。

#### 3.6.1 Angstrom(GPE)

(1)SD カードに二つのパーティションを作る(付録をご参照ください)。SD カードを再マウントして、次のコマンドを入力する。

```
cp /media/cdrom/linux/demo/angstrom/MLO /media/LABEL1
```

```
cp /media/cdrom/linux/demo/angstrom/u-boot.bin /media/LABEL1
```

```
cp /media/cdrom/linux/demo/angstrom/uImage /media/LABEL1
```

```
rm -rf /media/LABEL2/*
```

```
sudo tar jxvf
```

```
linux/demo/angstrom/Angstrom-DevKit8000-demo-image-glibc-ipk-2008.1-test-20080111-DevKit8000.rootfs.tar.bz2 -C /media/LABEL2
```

```
sync
```

```
umount /media/LABEL1
```

```
umount /media/LABEL2
```

(2)SD カードをボードに挿入し、SDカードから起動する。デフォルトはDVI で出力する。

初起動時はイニシャルのため、時間がかかります。

もっと詳しい情報は下記URLをご参照ください。

<http://www.angstrom-distribution.org/>

<http://www.angstrom-distribution.org/demo/beagleboard/>

#### 3.6.2 Google android

(1)SD カードに二つのパーティションを作る(付録をご参照ください)。SD カードを再マウントして、

次のコマンドを入力する。

ア 4.3 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_4.3 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_4.3 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

イ 5.6 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_5.6 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_5.6 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

ウ 7 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/android/MLO /media/LABEL1
cp /media/cdrom/linux/demo/android/u-boot.bin_7 /media/LABEL1/u-boot.bin
cp /media/cdrom/linux/demo/android/uImage_7 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/ android/RFS.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

(2) SD カードを挿入、電源入れて、SD カードから起動すると、Android システムになる。

Android のソースコードと資料は下記 URL をご参照ください。

<http://www.embedinfo.com/english/download/linuxadroid.zip>

[http://labs.embinux.org/index.php/Android\\_Porting\\_Guide\\_to\\_Beagle\\_Board](http://labs.embinux.org/index.php/Android_Porting_Guide_to_Beagle_Board)

### 3.6.3 DVSDK

(1) SD カードに二つのパーティションを作る(付録をご参照ください)。SD カードを再マウントして、次のコマンドを入力する。



① 4.3 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/dv sdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/uImage_4.3 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dv sdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

② 5.6 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/dv sdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/uImage_5.6 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dv sdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

③ 7 インチ LCD の場合：

```
cp /media/cdrom/linux/demo/dv sdk/MLO /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/u-boot.bin /media/LABEL1
cp /media/cdrom/linux/demo/dv sdk/uImage_7 /media/LABEL1/uImage
rm -rf /media/LABEL2/*
sudo tar jxvf linux/demo/dv sdk/DVSDK.tar.bz2 -C /media/LABEL2
sync
umount /media/LABEL1
umount /media/LABEL2
```

(2) SD カードを挿入、電源入れて、SD カードから起動すると、DVSDK システムになる。



## 第四章 Linux の開発

### 4.1 開発環境

ホストLinux はubuntu7.10 をお勧めします。ほかのLinux も使える。

付属DVDをパソコンに挿入する。UbuntuはDVDを/media/cdromにマウントする。

クロスツールは/media/cdrom/linux/tools にある。

#### 4.1.1 クロスコンパイルツールをインストールする

```
cd /media/cdrom/linux/tools tar xvjf arm-2007q3-51-arm-none-linux-gnueabi-i686.tar.bz2 -C /home/embest
```

///home/embest はユーザーフォルダ

#### 4.1.2 他のツールのインストール

```
mkdir /home/embest/tools
```

```
cp /media/cdrom/linux/tools/mkimage /home/embest/tools
```

```
cp /media/cdrom/linux/tools/signGP /home/embest/tools
```

```
cp /media/cdrom/linux/tools/mkfs.ubifs /home/embest/tools
```

```
cp /media/cdrom/linux/tools/ubinize /home/embest/tools
```

```
cp /media/cdrom/linux/tools/ubinize.cfg /home/embest/tools
```

#### 4.1.3 環境変数の追加

```
export PATH=/home/embest/arm-2007q3/bin:/home/embest/tools:$PATH
```

### 4.2 システムのコンパイル

#### 4.2.1 コンパイルファイルの準備

ソースコードは付属 DVD の linux/source で、使う前に linux 環境に解凍する必要。

```
mkdir /home/embest/work
```

```
cd /home/embest/work
```

```
tar xvf /media/cdrom/linux/source/x-load-1.41.tar.bz2
```

```
tar xvf /media/cdrom/linux/source/u-boot-1.3.3.tar.bz2
```

```
tar xvf /media/cdrom/linux/source/linux-2.6.28-omap.tar.bz2
```

```
sudo tar xvf /media/cdrom/linux/source/rootfs.tar.bz2
```

上記操作完了後、カレントフォルダに linux-2.6.22-omap、u-boot-1.3.3、x-load-1.41、rootfs 四つのフォルダが生成される。

#### 4.2.2 x-loader イメージファイルのコンパイル

MMC/SD カード起動と NAND 起動によって二つの x-loader イメージファイルになっている。それぞれ





れ生成方法も違う。

(1) SD カード起動用の x-loader イメージファイル MLO の生成

```
cd x-load-1.41
make distclean
make omap3devkit8000_config
make
signGP x-load.bin
mv x-load.bin.ift MLO
```

上記操作後、カレントフォルダに MLO ファイルが生成される。

(2) NAND 起動用の x-load.bin.ift\_for\_NAND の生成

①□ x-loader-1.4.1/include/configs/omap3devkit8000.h ファイルを修正する。

```
vi x-loader-1.4.1/include/configs/omap3devkit8000.h
```

下記の行をコメントアウトする。

```
//#define CFG_CMD_MMC 1
```

②クロスコンパイル

```
cd x-load-1.41
make distclean
make omap3devkit8000_config
make
signGP x-load.bin
mv x-load.bin.ift x-load.bin.ift_for_NAND
```

上記操作後、カレントフォルダに x-load.bin.ift\_for\_NAND ファイルが生成される。

4.2.3 u-boot.bin ファイルのコンパイル

```
cd u-boot-1.3.3
make distclean
make omap3devkit8000_config
make
```

上記操作後、カレントフォルダに u-boot.bin ファイルが生成される。

4.2.4 uImage カーネルファイルのコンパイル

```
cd linux-2.6.28-omap
make distclean
make omap3_devkit8000_defconfig
make uImage
```

上記操作後、arch/arm/boot フォルダに uImage ファイルが生成される。



#### 4.2.5 ファイルシステムファイルのコンパイル

```
cd /home/embest/work
```

```
sudo /home/embest/tools/mkfs.ubifs -r rootfs -m 2048 -e 129024 -c 812 -o ubifs.img
```

```
sudo /home/embest/tools/ubinize -o ubi.img -m 2048 -p 128KiB -s 512  
/home/embest/tools/ubinize.cfg
```

上記操作後、カレントフォルダに ubi.img ファイルが生成される。

#### 4.3 システムのコンフィグ

Linux のカーネルは色んなカーネルコンフィグオプションがありますが、デフォルトのコンフィグを基  
づいて必要に応じて追加、削除出来る。

##### 4.3.1 カーネルコンフィグの変更

デフォルトのコンフィグファイルは下記フォルダ：

```
linux-2.6.28-omap/arch/arm/configs/omap3_devkit8000_defconfig
```

下記コマンドでコンフィグを変更する：

```
cd linux-2.6.28-omap
```

```
cp arch/arm/configs/omap3_devkit8000_defconfig .config
```

```
make menuconfig
```



usb gadget で usb mass storage device をシミュレーションする例で、コンフィグ変更を説明する。

① Device drivers を選択する。

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

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

② USB support を選択する。



```
[ ] ISDN support --->
  Input device support --->
  Character devices --->
<*) I2C support --->
[*] SPI support --->
-* GPIO Support --->
<> Dallas's 1-wire support --->
<> Power supply class support --->
<> Hardware Monitoring support --->
<> Generic Thermal sysfs driver --->
[ ] Watchdog Timer Support --->
  Sonics Silicon Backplane --->
  Multifunction device drivers --->
  Multimedia devices --->
  Graphics support --->
<*) Sound card support --->
[*] HID Devices --->
[*] USB support --->
<*) MMC/SD/SDIO card support --->
<> Sony MemoryStick card support (EXPERIMENTAL) --->
[ ] Accessibility support --->
[*] LED Support --->
<*) Real Time Clock --->
[ ] DMA Engine support --->
[ ] Voltage and Current Regulator Support --->
<> Userspace I/O drivers --->
  CBUS support --->
```

**<Select>**   < Exit >   < Help >

- ③ USB Gadget Support を選択する。



```
<> EMI 6|2m USB Audio interface support
<> EMI 2|6 USB Audio interface support
<> ADU devices from Ontrak Control Systems
<> USB 7-Segment LED Display
<> USB Diamond Rio500 support
<> USB Lego Infrared Tower support
<> USB LCD driver support
<> USB BlackBerry recharge support
<> USB LED driver support
<> Cypress CY7C63xxx USB driver support
<> Cypress USB thermometer driver support
<> USB Phidgets drivers
<> Siemens ID USB Mouse Fingerprint sensor support
<> Elan PCMCIA CardBus Adapter USB Client
<> Apple Cinema Display support
<> USB 2.0 SVGA dongle support (Net2280/SiS315)
<> USB LD driver
<> PlayStation 2 Trance Vibrator driver support
<> IO Warrior driver support
<> USB testing driver
<> iSight firmware loading support
<> USB VST driver
<*) USB Gadget Support --->
    *** OTG and related infrastructure ***
<> GPIO based peripheral-only VBUS sensing 'transceiver'
<> Philips ISP1301 with OMAP OTG
<*) TWL4030 USB Transceiver Driver
```

&lt;Select&gt;

&lt; Exit &gt;

&lt; Help &gt;

## ④ USB Gadget Support を変更する。

```
--- USB Gadget Support
[ ] Debugging messages (DEVELOPMENT)
[ ] Debugging information files (DEVELOPMENT)
(2) Maximum VBUS Power usage (2-500 mA)
USB Peripheral Controller (Inventra HRC USB Peripheral (TI, ADI, ...)) --->
<M> USB Gadget Drivers
<> Gadget Zero (DEVELOPMENT)
<> Ethernet Gadget (with CDC Ethernet support)
<> Gadget Filesystem (EXPERIMENTAL)
<M> File-backed Storage Gadget
[ ] File-backed Storage Gadget testing version (NEW)
<> Serial Gadget (with CDC ACM and CDC OBEX support)
<> MIDI Gadget (EXPERIMENTAL)
<> Printer Gadget
<> CDC Composite Device (Ethernet and ACM)

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

## 4.3.2 コンパイルする

コンフィグを保存して、下記コマンドでカーネルをコンパイルする。

```
make
```

```
make uImage
```

上記操作後、arch/arm/boot フォルダに新しいカーネルファイル uImage が生成され、drivers/usb/gadget フォルダに g\_file\_storage.ko ファイルが生成される。

## 4.3.3 テスト

SD カード上のuImageファイルを更新し、g\_file\_storage.koファイルもSDカードにコピーする。SDカードから起動して、次のコマンドでSDカードをUSBメモリデバイスに仮想して、パソコンからアクセスする。

```
root@DevKit8000:~# cd /media/mmcblk0p1/
```

```
root@DevKit8000:/media/mmcblk0p1# insmod g_file_storage.ko file=/dev/mmcblk0p1 stall=0 removable=1
```

```
g_file_storage gadget: File-backed Storage Gadget, version: 7 August 2007 g_file_storage gadget:
```



Number of LUNs=1

g\_file\_storage gadget-lun0: ro=0, file: /dev/mmcblk0p1

musb\_hdrc musb\_hdrc: MUSB HDRC host driver

musb\_hdrc musb\_hdrc: new USB bus registered, assigned bus number 2

usb usb2: configuration #1 chosen from 1 choice

hub 2-0:1.0: USB hub found

hub 2-0:1.0: 1 port detected

※USB miniB to USB A 変換ケーブルでボードと PC を繋ぐと、PC 側で usb mass storage device を発見した提示が出て来る。

## 第五章 Linux システムイメージの書き込み

MMC/SD 或いは NAND 起動によって、書き込み方法が異なる。

### 5.1 SD カード起動時システムイメージの更新

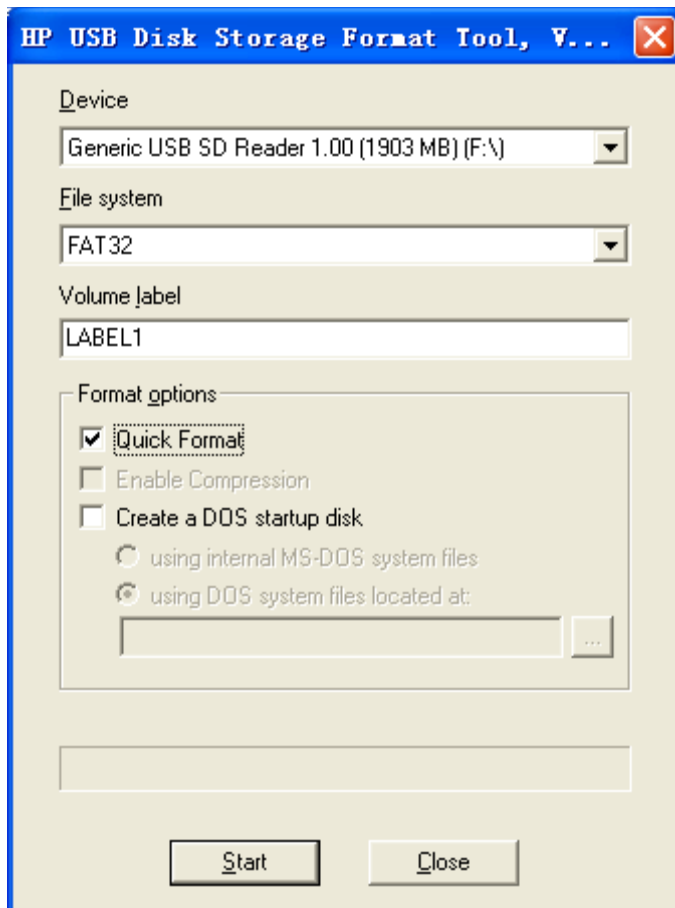
#### 5.1.1 SD カードの準備

推薦するフォーマットツールは HP USB Disk Storage Format Tool 2.0.6:

<http://selfdestruct.net/misc/usbboot/SP27213.exe>

或いは付属 DVD の ¥Linux¥tools¥formatSD.zip をご参照ください。

- ① MMC/SD を PC のカードリーダーに差し込む。
- ② HP USB Disk Storage Format Tool ツールを開く:



- ③ "FAT32"ファイルシステムを選択する。
- ④ "Start"をクリックする。
- ⑤ フォーマット終了後、"OK"をクリックする。

※複数の Volume 持つ SD カードをフォーマットする場合、Volume 情報まで初期化される。

#### 5.1.2 イメージファイルの更新

PC で生成されたイメージファイルを SD カードに直接コピーする。

※元のファイルはバックアップするのをお勧めします。





### 5.1.3 u-boot パラメータ設定

電源を投入した後、3 秒以内、PCのスペースキーを押して、u-bootモードに入る。

u-boot モードで次のコマンドでパラメータを設定できる。

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 root=/dev/ram  
initrd=0x81600000,40M video=omapfb:mode:4.3inch_LCD
```

```
OMAP3 DevKit8000 # setenv bootcmd 'mmcinit;fatload mmc 0 80300000 uImage;fatload mmc 0  
81600000 ramdisk.gz;bootm 80300000'
```

```
OMAP3 DevKit8000 # saveenv
```

### 5.2 NAND Flash のシステムイメージの更新

u-boot モードで行う。U-boot モードで MMC/SD 或いは TFTP 経由の二つの方法がありますが、下記は MMC/SD 経由で説明する。

x-load.bin.ift\_for\_NAND、flash-uboot.bin、uImage、ubi.img ファイルを SD カードにコピーし、ボードに差し込む。電源入れて 3 秒以内、任意キーを押して、u-boot モードに入る。

#### 5.2.1 x-load の更新

```
OMAP3 DevKit8000 # mmcinit
```

```
OMAP3 DevKit8000 # fatload mmc 0:1 80000000 x-load.bin.ift_for_NAND
```

```
reading x-load.bin.ift_for_NAND
```

```
9664 bytes read
```

```
OMAP3 DevKit8000 # nand unlock
```

```
device 0 whole chip
```

```
nand_unlock: start: 00000000, length: 134217728!
```

```
NAND flash successfully unlocked
```

```
OMAP3 DevKit8000 # nand ecc hw
```

```
OMAP3 DevKit8000 # nand erase 0 80000
```

```
NAND erase: device 0 offset 0x0, size 0x80000
```

```
Erasing at 0x60000 -- 100% complete.
```

```
OK
```

```
OMAP3 DevKit8000 # nand write.i 80000000 0 $(filesize)
```

```
NAND write: device 0 offset 0x0, size 0x80000
```

```
Writing data at 0x7f800 -- 100% complete.
```

```
524288 bytes written: OK
```

#### 5.2.2 u-boot.bin ファイルの更新

```
OMAP3 DevKit8000 # mmcinit
```



```
OMAP3 DevKit8000 # fatload mmc 0:1 80000000 flash-uboot.bin
reading flash-uboot.bin
1085536 bytes read
OMAP3 DevKit8000 # nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 134217728!
NAND flash successfully unlocked
OMAP3 DevKit8000 # nand ecc sw
OMAP3 DevKit8000 # nand erase 80000 160000
NAND erase: device 0 offset 0x80000, size 0x160000
Erasing at 0x1c0000 -- 100% complete.
OK
OMAP3 DevKit8000 # nand write.i 80000000 80000 $(filesize)
NAND write: device 0 offset 0x80000, size 0x160000
Writing data at 0x1df800 -- 100% complete.
1441792 bytes written: OK
```

### 5.2.3 カーネルの更新

```
OMAP3 DevKit8000 # mmcinit
OMAP3 DevKit8000 # fatload mmc 0:1 80000000 uImage
reading uImage
1991900 bytes read
OMAP3 DevKit8000 # nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 268435456!
NAND flash successfully unlocked
OMAP3 DevKit8000 # nand ecc sw
OMAP3 DevKit8000 # nand erase 280000 300000
NAND erase: device 0 offset 0x280000, size 0x200000
Erasing at 0x460000 -- 100% complete.
OK
OMAP3 DevKit8000 # nand write.i 80000000 280000 $(filesize)
NAND write: device 0 offset 0x280000, size 0x200000
Writing data at 0x47f800 -- 100% complete.
2097152 bytes written: OK
```



#### 5.2.4 ファイルシステムの更新

```
OMAP3 DevKit8000 # mmcinit
OMAP3 DevKit8000 # fatload mmc 0:1 81000000 ubi.img
reading ubi.img
12845056 bytes read
OMAP3 DevKit8000 # nand unlock
device 0 whole chip
nand_unlock: start: 00000000, length: 268435456!
NAND flash successfully unlocked
OMAP3 DevKit8000 # nand ecc sw
OMAP3 DevKit8000 # nand erase 680000
NAND erase: device 0 offset 0x680000, size 0x7980000
Erasing at 0x7fe0000 -- 100% complete.
OK
OMAP3 DevKit8000 # nand write.i 81000000 680000 $(filesize)
NAND write: device 0 offset 0x680000, size 0xc40000
Writing data at 0x12bf800 -- 100% complete.
12845056 bytes written: OK
```

#### 5.2.5 u-boot パラメータの設定

```
OMAP3 DevKit8000 # setenv bootargs console=ttyS2,115200n8 ubi.mtd=4 root=ubi0:rootfs
rootfstype=ubifs video=omapfb:mode:4.3inch_LCD
OMAP3 DevKit8000 # setenv bootcmd nand read. i 80300000 280000 300000; bootm 80300000
OMAP3 DevKit8000 # saveenv
```

## 第六章 Linux のアプリの開発

### 6.1 サンプルLED 点灯

#### 6.1.1 coding

led\_acc.c ソースコード、三つの LED をアキュムレータ方式で点灯させる。

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/ioctl.h>
#include <fcntl.h>
#define LED1 "/sys/class/leds/led1/brightness"
#define LED2 "/sys/class/leds/led2/brightness"
#define LED3 "/sys/class/leds/led3/brightness"
int main(int argc, char *argv[])
{
    int f_led1, f_led2, f_led3;
    unsigned char i = 0;
    unsigned char dat1, dat2, dat3;
    if((f_led1 = open(LED1, O_RDWR)) < 0)
    {
        printf("error in open %s",LED1); return -1;
    }
    if((f_led2 = open(LED2, O_RDWR)) < 0)
    {
        printf("error in open %s",LED2); return -1;
    }
    if((f_led3 = open(LED3, O_RDWR)) < 0)
    {
        printf("error in open %s",LED3); return -1;
    }
    for(;;){
        i++;
        dat1 = i&0x1 ? '1':'0';
        dat2 = (i&0x2)>>1 ? '1':'0';
        dat3 = (i&0x4)>>2 ? '1':'0';
```



```
        write(f_led1, &dat1, sizeof(dat1));  
        write(f_led2, &dat2, sizeof(dat2));  
        write(f_led3, &dat3, sizeof(dat3));  
        usleep(300000);  
    }  
}
```

### 6.1.2 クロスコンパイル

```
arm-none-linux-gnueabi-gcc led_acc.c -o led_acc
```

### 6.1.3 ダウンロードして実行する

SDカードまたはUSBメモリまたはLANでボードにダウンロードする。次のコマンドで実行させる。

```
./led_acc
```

## 第七章 WindowsCE 6.0 概要

本ボードのソフトウェアシステムは下記内容で構成されている: プレーコンパイルしたイメージファイル、アプリのイメージファイル、静的ライブラリ、動的ライブラリ、ソースコード、コンパイルツール、その他開発ツールなど。コンパイルツールについては Microsoft 社の Web から入手できる(付録に各 URL を紹介している)。イメージファイル、ソースコードなどは付属している SD カードと DVD にある。

### ① 付属 SD カードにある内容:

- X-Loader イメージファイル (MLO)
- Ethernet Bootloader (EBOOT) イメージファイル (EBOOTSD.nb0)
- Windows Embedded CE 6.0 サンプル OS イメージファイル (NK.bin)
- テストプログラム (ADevKit8000.exe)

### ② 付属 DVD にある内容:

- OMAP35X の Windows Embedded CE 6.0 DevKit8000 Board Support Package (BSP) ソースコード
- この BSP の Windows Embedded CE 6.0 参考プロジェクト
- アプリサンプルのソースコード
- 開発ツール

### 7.1 初期時イメージの紹介

Windows Embedded CE 6.0 はマルチメディア、PDA、携帯、Micro カーネルなど色んなフォーマットを提供していますが、初期時のイメージは Mobile Handheld を基づいて下記特徴がある:



Image	Feature
X-Loader	To boot EBOOT
EBOOT	To boot the operating system from the network (network card or RNDIS)
	To boot the operating system with SD card
	To boot the operating system from the NAND Flash
Demonstrated operating system	Windows Explorer
	Console Window
	CAB File Installer/Uninstaller
	Internet Explorer 6.0
	ActiveSync
	Power Management (Full)
	.NET Compact Framework 3.5
	Hive-based Registry
	RAM and ROM File System
	Device Drivers



## 7.2 BSP の特徴

Module	Feature
X-Loader module	NAND
	ONENAND
	SD
EBOOT module	NAND
	ONENAND
	SD
OAL module	ILT
	REBOOT
	Watchdog
	RTC
KITL module	RNDIS KITL
Driver module	NLED driver
	GPIO/I2C/SPI/MCBSP driver
	Series port driver
	6X6 keyboard driver
	Audio driver
	NAND(K9F1G08)driver
	Display driver(LCD/DVI. S end/TV)/ TOUCH driver
	SD/MMC/SDIO driver
	DM9000 network card driver
	USB OTG driver
	USB EHCI driver
	VRFB driver
	DSPLINKK/CMEMK driver
	GPIO keyboard driver
	PWM(TPS65930)driver
	ADC(TPS65930)driver
ONENAND driver	
SMSC911X network card driver	
Power management module	Backlight driver
	Battery driver
	Sleep / wake-up button driver
	Expansion of power management
Application module	Flash Plug-in and Flash player
	MP3/MPEG4/H264 DSP Hardware decoder
	BSPINFO(control panel)
	CETK



### 7.3 WinCE システムの起動

#### 7.3.1 操作中、PC 側のハイパーターミナルの設定

- Baud rate: 115200
- Data bit: 8
- Parity check: no
- Stop bit: 1
- Flow control: no

#### 7.3.2 NAND Flash 起動

ボードのデフォルト設定はNANDから起動する。NANDにシステムイメージファイルがなければ、SDカードから起動する。

#### 7.3.3 SD カードから起動

SD カードから起動したい場合はボードの「Boot key」を押しながら、起動する。

#### 7.3.4 テストプログラム

※ボードのシリアルポートをPCと接続する。またボードにLCD、キーボード、AUDIO IN/OUT設備を接続する必要で、足りない場合はテスト結果に影響ある恐れがあります。

- ①  PCのハイパーターミナルを7.3.1と同じように設定する。
- ② PCのファイアウォールを閉じて、下記の様に設定する：  
IPアドレス: 192.168.1.2  
サブネットマスク: 255.255.255.0
- ③ SDカードを挿入し、シリアルケーブルでPCと繋ぐ。そして、キーボードとAUDIO設備を接続する。
- ④ 電源をいれる。ハイパーターミナル画面で情報が出て来る。初期化終了後、Windows Embedded CE 6.0システム開始する。
- ⑤ ADevKit8000.exe[¥Storage Card]を実行する。
- ⑥ Startをクリックしてテストを始める。



## 第八章 WinCE6.0 の開発

### 8.1 開発環境の構築

WinCE6.0 の開発は次のソフトを順番でインストールする必要。

アプリケーションの開発：

- Visual Studio 2005
- Visual Studio 2005 SP1
- Visual Studio 2005 SP1 Update for Vista (if applicable)
- ActiveSync 4.5

Windows Embedded CE 6.0 の開発：

- Visual Studio 2005
- Visual Studio 2005 SP1
- Visual Studio 2005 SP1 Update for Vista (if applicable)
- Windows Embedded CE 6.0 Platform Builder
- Windows Embedded CE 6.0 SP1
- Windows Embedded CE 6.0 R2
- Windows Embedded CE 6.0 Product Update Rollup 12/31/2008

※既に古いバージョンの CE 環境を持っている場合は、アンインストールしてから上記ソフトの順で再インストールください。

また依頼関係のため、順番は変更しないでください。

場所もデフォルトのままインストールください。

### 8.2 コンパイル

#### 8.2.1 コンパイルファイルの準備

¥wince\_6¥bsp¥DevKit8000.zip ファイルを解凍して DevKit8000 フォルダになる。

¥DevKit8000¥bsp¥フォルダにあるDevKit8000フォルダをC:¥WINCE600¥PLATFORMIにコピーする。

¥DevKit8000¥bsp\_prj¥フォルダにあるDevKit8000フォルダをC:¥WINCE600¥OSDesigns(作成する必要)にコピーする。

#### ① 4.3インチLCDの場合

platform/DevKit8000/src/drivers/lcd/vga/lcd\_vga.cを下記の様に修正する。

```
#define LCD_4_3_INCH 1  
  
//#define LCD_5_6_INCH 1  
  
//#define LCD_7_INCH 1
```

#### ② 5.6インチLCDの場合

platform/DevKit8000/src/drivers/lcd/vga/lcd\_vga.cを下記の様に修正する。



```
//#define LCD_4.3_INCH 1
```

```
#define LCD_5.6_INCH 1
```

```
//#define LCD_7_INCH 1
```

### ③ 7インチLCDの場合

platform/DevKit8000/src/drivers/lcd/vga/lcd\_vga.cを下記の様に修正する。

```
//#define LCD_4.3_INCH 1
```

```
//#define LCD_5.6_INCH 1
```

```
#define LCD_7_INCH 1
```

### ④ VGAの場合

C:\¥WINCE600¥DevKit8000¥BSP¥DevKit8000¥DevKit8000.batを下記の様に修正する。

```
set BSP_DVI_1024W_768H=1
```

## 8.2.2 システムのコンパイル

1、C:\¥WINCE600¥OSDesigns¥DevKit8000フォルダにあるプロジェクトファイルDevKit8000.slnを開く。或いは次の手順で新プロジェクトを作る。

- ・Visual Studio 2005 を開く
- ・メニューFile[New→Project]を選択する
- ・Platform Builder for CE 6.0 のテンプレートを選択する
- ・Windows Embedded CE 6.0 OS Design Wizard を開く
- ・Embext DevKit8000 BSP を選択する。
- ・Wizardより操作を完成する。

2、メニューからBuild→ Global Build Settings

- ・Copy Files to Release Directory After Build
- ・Make Run-Time Image After build

3、KITL必要な場合は、Build Optionsの[Project→ Properties]タブのEnable Kernel DebuggerとEnable KITLにチェック入れる。

4、Build→ Build Solution選択してBSPをコンパイルする。

5、コンパイル終了後NK.bin、EBOOTSD.nb0とMLOファイルが生成される。

C:\¥WINCE600¥OSDesigns¥DevKit8000¥DevKit8000¥RelDir¥DevKit8000\_ARMV4I\_Releaseフォルダにある上記ファイルをSDカードにコピーし、ボードに挿入して、SDカードから起動すればテスト出来る。



## 8.2.3 システムのコンフィグ

Component	Path
CAB File Installer/Uninstaller	Core OS->CEBASE->Application – End User
.NET Compact Framework 3.5	Core OS->CEBASE->Applications and Services Development->.NET Compact Framework 3.5
OS Dependencies for .NET Compact Framework 3.5	Core OS->CEBASE->Applications and Services Development->.NET Compact Framework 3.5-> OS Dependencies for .NET Compact Framework 3.5
Point-to-Point Protocol over Ethernet (PPPoE)	Core OS->CEBASE->Communication Services and Networking->Networking – Wide Area Network (WAN)
USB Function Driver	Core OS->CEBASE->Core OS Services->USB Host Support
USB Host Support	Core OS->CEBASE->Core OS Services->USB Host Support
USB Human Input Device (HID) Class Driver	Core OS->CEBASE->Core OS Services->USB Host Support
USB HID Keyboard and Mouse	Core OS->CEBASE->Core OS Services->USB Host Support-> USB Human Input Device (HID) Class Driver
USB Storage Class Driver	Core OS->CEBASE->Core OS Services->USB Host Support
RAM and ROM File System	Core OS->CEBASE->File Systems and Data Store->File System – Internal (Choose 1)
Hive-based Registry	Core OS->CEBASE->File Systems and Data Store->Registry Storage – Internal (Choose 1)
exFAT File System	Core OS->CEBASE->File Systems and Data Store->Storage Manager
FAT File System	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Storage Manager Control Panel Applet	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Transaction-Safe FAT File System (TFAT)	Core OS->CEBASE->File Systems and Data Store->Storage Manager
Video/Image Compression Manager	Core OS->CEBASE->Graphics and Multimedia Technologies->Media->Video Codecs and Renderers
Console Window	Core OS->CEBASE->Shell and User Interface->Shell->Command Shell
SD Memory	Device Drivers->SDIO->SDIO Memory
serial	Device Drivers->USB Function->USB Function Clients
Windows Embedded CE Test Kit	Device Drivers

Windows Embedded CE 6.0のコンフィグはVisual Studio 2005(VS2005)環境のCatalog Items Viewで追加或いは削減する事が出来る。

## 第九章 WinCE システムの更新

### 9.1 SD カードシステムイメージの更新

#### 9.1.1 SD カードの準備

ツール HP Disk Storage Format Tool で SD カードを FAT 或いは FAT32 ファイルシステムにフォーマットする。

#### 9.1.2 イメージの更新

付属の DVD の¥wince.6¥image の適用な LCD のフォルダにある MLO、EBOOTSD.nb0、NK.bin と ADevKit8000.exe ファイルを SD カードにコピーする。

### 9.2 NAND Flash システムイメージの更新

#### 9.2.1 更新ファイルの準備

1、ツール HP Disk Storage Format Tool で SD カードを FAT 或いは FAT32 ファイルシステムにフォーマットする。

2、¥wince.6¥imageの適用なLCDのフォルダにあるMLO、EBOOTNAND.nb0、NK.bin、XLDRNAND.nb0とADevKit8000.exeファイルをSDカードにコピーする。

また、EBOOTNAND.nb0をEBOOTSD.nb0にネーム変更する。

#### 9.2.2 イメージの更新

1、BOOTキーを押しながら、SDカード挿入して電源を入れる。SDカードから起動する。ハイパーターミナル画面に起動情報表示した後、[SPACE]キー押してEBOOTメニューに入る。

2、[5]を選択して、Flash管理メニューに入る。

3、[a]、[b]、[c]の順に選択して、XLDR、EBOOTとNKイメージを書き込む。

4、[0]を選択して、メインメニューに戻る。[2]、[4]、[7]と[y]の順に選択して、起動デバイスを変更する。

5、SDカードを抜き出して、システムを再起動する。NAND Flashから起動する様になる。



## 第十章 WinCE アプリの開発

開発する前にWindows Mobile 6 Professional SDK のインストールが必要。下記URLからダウンロードできる。

<http://www.microsoft.com/downloads/details.aspx?familyid=06111A3A-A651-4745-88EF-3D48091A390B&displaylang=en>

### 10.1 アプリのインタフェースと使用例

アプリの開発はWindows Embedded CE 6.0 の標準APIを使用している。GPIOのAPIだけは標準API基に拡張している。

#### 10.1.1 GPIO インタフェースの定義

GPIO デバイス名は L"GPIO:"、DeviceIoControl の API を拡張し、IOCTL は下記の表：

IOCTL Code	Description
IOCTL_GPIO_SETBIT	Set GPIO pin as 1
IOCTL_GPIO_CLRBIT	Set GPIO pin as 0
IOCTL_GPIO_GETBIT	Read GPIO pin
IOCTL_GPIO_SETMODE	Set the working mode of GPIO pin
IOCTL_GPIO_GETMODE	Read the working mode of GPIO pin
IOCTL_GPIO_GETIRQ	Read the corresponding IRQ of GPIO pin

#### 10.1.2 使用例

##### 1、GPIO デバイスを開く

```
HANDLE hFile = CreateFile(T"GPIO:", (GENERIC_READ|GENERIC_WRITE), (FILE_SHARE_READ|FILE_SHARE_WRITE), 0,
OPEN_EXISTING, 0, 0);
```

##### 2、GPIO動作モードの設定とリード

```
DWORD id = 0, mode = 0;
```

GPIO動作モードの設定：

```
DWORD pInBuffer[2];
```

```
pInBuffer[0] = id;
```

```
pInBuffer[1] = mode;
```

```
DeviceIoControl(hFile, IOCTL_GPIO_SETMODE, pInBuffer, sizeof(pInBuffer), NULL, 0, NULL, NULL);
```

GPIO動作モードのリード：

```
DeviceIoControl(hFile, IOCTL_GPIO_GETMODE, &id, sizeof(DWORD), &mode, sizeof(DWORD), NULL, NULL);
```

“id”はGPIOピン番号で、“mode”はGPIO動作モード。下記の様に定義している：



Mode definition	Description
GPIO_DIR_OUTPUT	Output mode
GPIO_DIR_INPUT	Input mode
GPIO_INT_LOW_HIGH	Rising edge trigger mode
GPIO_INT_HIGH_LOW	Falling edge trigger mode
GPIO_INT_LOW	low level trigger mode
GPIO_INT_HIGH	high level trigger mode
GPIO_DEBOUNCE_ENABLE	Jumping trigger enable

### 3、GPIOピン操作

DWORD id = 0, pin = 0;

Output high level:

DeviceIoControl(hFile, IOCTL\_GPIO\_SETBIT, &id, sizeof(DWORD), NULL, 0, NULL, NULL);

Output low level:

DeviceIoControl(hFile, IOCTL\_GPIO\_CLRBIT, &id, sizeof(DWORD), NULL, 0, NULL, NULL);

Read the pin state

DeviceIoControl(hFile, IOCTL\_GPIO\_GETBIT, &id, sizeof(DWORD), &pin, sizeof(DWORD), NULL, NULL);

“id”はGPIOのピン番号で，“pin”でピンのステータスを返却する。

### 4、その他操作

GPIOピン対応しているIRQ番号を取得する。

DWORD id = 0, irq = 0;

DeviceIoControl(hFile, IOCTL\_GPIO\_GETIRQ, &id, sizeof(DWORD), &irq, sizeof(DWORD), NULL, NULL);

“id”はGPIOのピン番号で，“irq”でIRQ番号を返却する。

### 5、GPIOデバイスをクローズする。

CloseHandle(hFile);

※ GPIOのピン番号定義:0~191 MPU Bank1~6 GPIO, 192~209 TPS65930 GPIO 0~17。

※ IOCTLまたモードの定義は付属DVDのwince\_6¥inc¥gpio.hにある。使用时このファイルをインクルードしてください。

## 10.2 インタフェースアプリ開発例

### 10.2.1 新しいプロジェクトの作成

1、Visual Studio 2005を開く

2、メニューFileから[New->Project]を選択する

3、MFC Smart Device Applicationテンプレート[Visual C++->Smart Device]を選択する

4、MFC Smart Device Application Wizardを開く

5、PlatformsページでWindows Mobile 6 Professional SDKのみチェックする。

6、Application TypeページでDialog basedを選択する。



7、Wizard より完成する。

## 10.2.2 ソースコード

付属 DVD の¥wince\_6¥app フォルダにある ADevKit8000 を参照する事。

## 10.2.3 コンパイルと実行

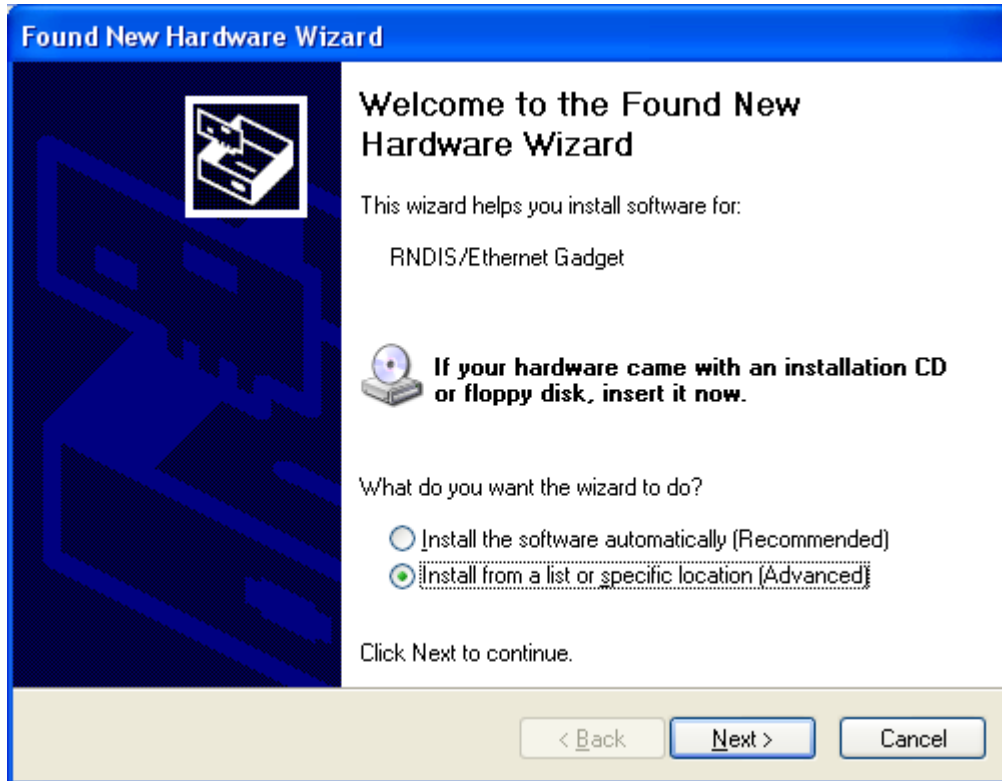
- 1、メニューの[Build→ Build Solution]選択して、実行ファイルADevKit8000.exeが生成される。
- 2、生成したADevKit8000.exeファイルをSDカードにコピーし、ボードに挿入して実行する。



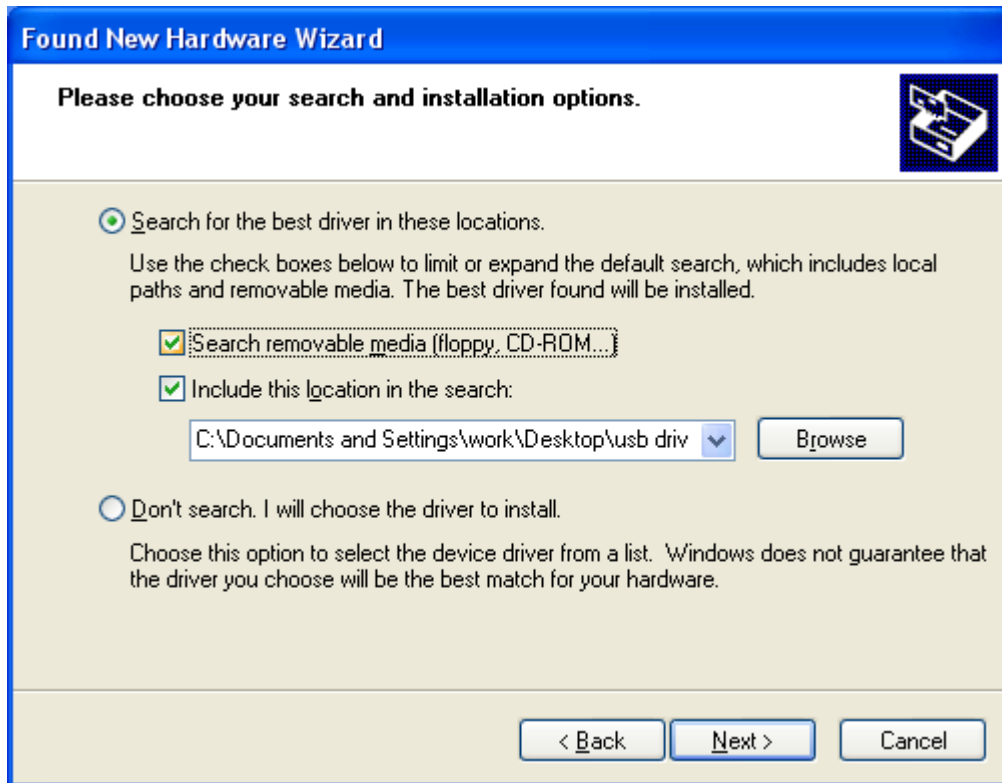
## 付録

### 付録一 Linux USB Ethernet/RNDIS Gadget ドライバのインストール

1、まだPC側にインストールしていない場合は、図のような提示画面が出て来る。



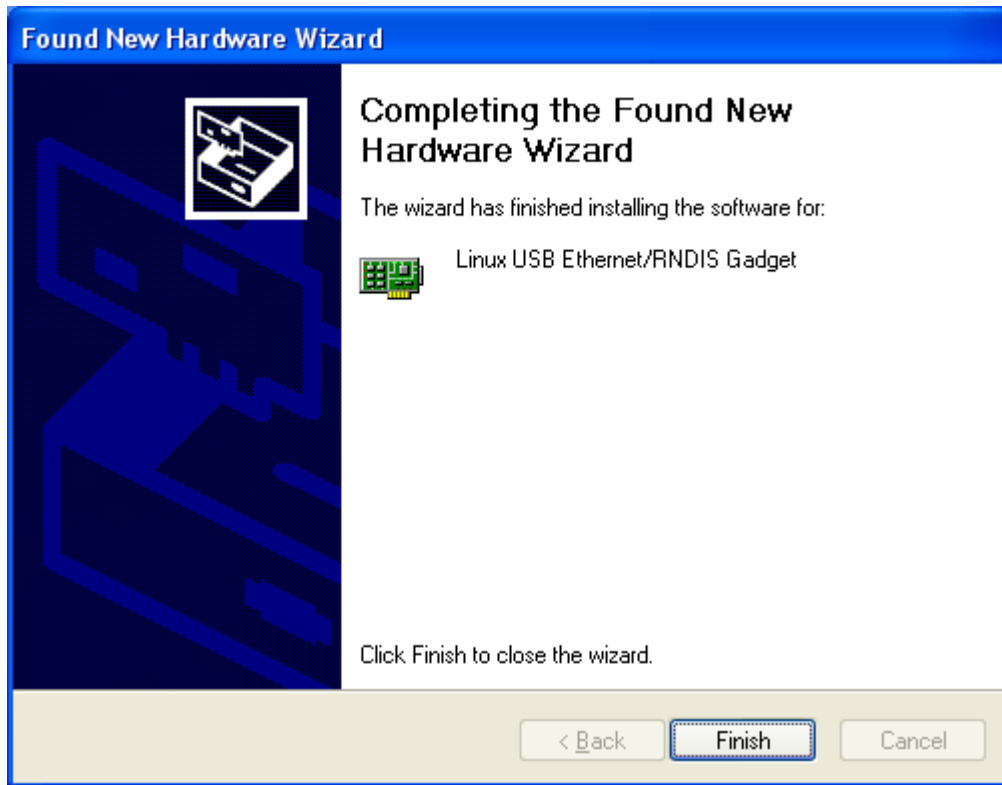
2、USBドライバのフォルダを付属DVDの¥linux¥toolsに指定して“Next”をクリックする。



3、下記の様な提示が出ても、インストールを継続する。



4、完了まで待つ。





## 付録二 Linux Boot Disk Format

How to create a dual-partition card for DevKit8000 to boot Linux from first partition and have root file system at second partition.

### 一、 Introduction

This guide is meant for those looking to create a **dual-partition** card, booting from a FAT partition that can be read by the OMAP3 ROM bootloader and Linux/Windows, then utilizing an ext3 partition for the Linux root file system.

### 二、 Details

Text marked with `[]` shows user input.

#### 1、 Determine which device the SD Card Reader is on your system

Plug the SD Card into the SD Card Reader and then plug the SD Card Reader into your system. After doing that, do the following to determine which device it is on your system.

```
$ [dmesg | tail] ...  
[ 6854.215650] sd 7:0:0:0: [sdc] Mode Sense: 0b 00 00 08  
[ 6854.215653] sd 7:0:0:0: [sdc] Assuming drive cache: write through [ 6854.215659] sdc:  
sdc1  
[ 6854.218079] sd 7:0:0:0: [sdc] Attached SCSI removable disk  
[ 6854.218135] sd 7:0:0:0: Attached scsi generic sg2 type 0  
...
```

In this case, it shows up as `/dev/sdc` (note `sdc` inside the square brackets above).

#### 2、 Check to see if the automounter has mounted the SD Card

Note there may be more than one partition (only one shown in the example below).

```
$ [df -h]  
Filesystem Size Used Avail Use% Mounted on  
...  
/dev/sdc1 400M 94M 307M 24% /media/disk  
...
```

Note the "Mounted on" field in the above and use that name in the `umount` commands below.

#### 3、 If so, unmount it

```
$ [umount /media/disk]
```

#### 4、 Start fdisk Be sure to choose the whole device (`/dev/sdc`), not a single partition (`/dev/sdc1`).

```
$ [sudo fdisk /dev/sdc]
```

#### 5、 Print the partition record

So you know your starting point. **Make sure to write down the number of bytes on the**



card (in this example, 2021654528).

Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes

255 heads, 63 sectors/track, 245 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sdc1 \* 1 246 1974240+ c W95 FAT32 (LBA)

Partition 1 has different physical/logical endings:

phys=(244, 254, 63) logical=(245, 200, 19)

#### 6、 Delete any partitions that are there already

Command (m for help): [d]

Selected partition 1

#### 7、 Set the Geometry of the SD Card

If the print out above does not show 255 heads, 63 sectors/track, then do the following expert mode steps to redo the SD Card:

##### 1)、 Go into expert mode.

Command (m for help): [x]

##### 2)、 Set the number of heads to 255.

Expert Command (m for help): [h]

Number of heads (1-256, default xxx): [255]

##### 3) Set the number of sectors to 63.

Expert Command (m for help): [s]

Number of sectors (1-63, default xxx): [63]

##### 4) Now Calculate the number of Cylinders for your SD Card.

#cylinders = FLOOR (the number of Bytes on the SD Card (from above) / 255 / 63 / 512 )

So for this example:  $2021654528 / 255 / 63 / 512 = 245.79$ . So we use 245 (i.e. truncate, don't round).

##### 5) Set the number of cylinders to the number calculated.

Expert Command (m for help): [c]

Number of cylinders (1-256, default xxx): [enter the number you calculated]

##### 6) Return to Normal mode.

Expert Command (m for help): [r]

#### 8、 Print the partition record to check your work

Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes

255 heads, 63 sectors/track, 245 cylinders



Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

### 9、 Create the FAT32 partition for booting and transferring files from Windows

Command (m for help): [n]

Command action

e extended

p primary partition (1-4)

[p]

Partition number (1-4): [1]

First cylinder (1-245, default 1): [(press Enter)]

Using default value 1

Last cylinder or +size or +sizeM or +sizeK (1-61, default 61): [+5]

Command (m for help): [t]

Selected partition 1

Hex code (type L to list codes): [c]

Changed system type of partition 1 to c (W95 FAT32 (LBA))

### 10、 Mark it as bootable

Command (m for help): [a]

Partition number (1-4): [1]

### 11、 Create the Linux partition for the root file system

Command (m for help): [n]

Command action

e extended

p primary partition (1-4) [p]

Partition number (1-4): [2]

First cylinder (7-61, default 7): [(press Enter)]

Using default value 52

Last cylinder or +size or +sizeM or +sizeK (7-61, default 61): [(press Enter)]

Using default value 245

### 12、 Print to Check Your Work

Command (m for help): [p]

Disk /dev/sdc: 2021 MB, 2021654528 bytes

255 heads, 63 sectors/track, 245 cylinders

Units = cylinders of 16065 \* 512 = 8225280 bytes

Device Boot Start End Blocks Id System

/dev/sdc1 \* 1 6 409626 c W95 FAT32 (LBA)

```
/dev/sdc2 7 61 1558305 83 Linux
```

### 13、 Save the new partition records on the SD Card

This is an important step. All the work up to now has been temporary.

```
Command (m for help): [w]
```

```
The partition table has been altered!
```

```
Calling ioctl() to re-read partition table.
```

```
WARNING: Re-reading the partition table failed with error 16: Device or resource busy.
```

```
The kernel still uses the old table.
```

```
The new table will be used at the next reboot.
```

```
WARNING: If you have created or modified any DOS 6.x partitions, please see the fdisk manual page for additional information.
```

```
Syncing disks.
```

### 14、 Format the partitions

The two partitions are given the volume names LABEL1 and LABEL2 by these commands.

You can substitute your own volume labels.

```
$ [sudo mkfs.msdos -F 32 /dev/sdc1 -n LABEL1]
```

```
mkfs.msdos 2.11 (12 Mar 2005)
```

```
$ [sudo mkfs.ext3 -L LABEL2 /dev/sdc2]
```

```
mke2fs 1.40-WIP (14-Nov-2006)
```

```
Filesystem label=
```

```
OS type: Linux
```

```
Block size=4096 (log=2)
```

```
Fragment size=4096 (log=2)
```

```
195072 inodes, 389576 blocks
```

```
19478 blocks (5.00%) reserved for the super user
```

```
First data block=0
```

```
Maximum filesystem blocks=402653184
```

```
12 block groups
```

```
32768 blocks per group, 32768 fragments per group
```

```
16256 inodes per group
```

```
Superblock backups stored on blocks:
```

```
32768, 98304, 163840, 229376, 294912
```

```
Writing inode tables: done
```

```
Creating journal (8192 blocks): done
```

```
Writing superblocks and filesystem accounting information:
```

## 付録三 TFTP サーバーの構築

### 1、 client側のインストール

```
$>sudo apt-get install tftp-hpa
```

```
$>sudo apt-get install tftpd-hpa
```

### 2、 inetのインストール

```
$>sudo apt-get install xinetd
```

```
$>sudo apt-get install netkit-inetd
```

### 3、 サーバーの設定

まずは、ルートフォルダでtftpbootを作成して、属性を設定する:

```
$>cd /
```

```
$>sudo mkdir tftpboot
```

```
$>sudo chmod 777 tftpboot
```

/etc/inetd.confに内容を追加する:

```
$>sudo vi /etc/inetd.conf //copy the follow word to this file
```

```
tftpd dgram udp wait root /usr/sbin/in.tftpd /usr/sbin/in.tftpd -s /tftpboot
```

そして、inetdを再ロードする:

```
$>sudo /etc/init.d/inetd reload
```

最後は/etc/xinetd.d/フォルダにtftpファイルを作って、内容を設定する。

```
$>cd /etc/xinetd.d/
```

```
$>sudo touch tftp
```

```
$>sudo vi tftp ////copy the follow word to tftp file
```

```
service tftp
```

```
{
```

```
    disable = no
```

```
    socket_type = dgram
```

```
    protocol = udp
```

```
    wait = yes
```

```
    user = root
```

```
    server = /usr/sbin/in.tftpd
```

```
    server_args = -s /tftpboot -c
```

```
    per_source = 11
```

```
    cps = 100 2
```

```
}
```

### 4、 再起動する

```
$>sudo /etc/init.d/xinetd restart
```

```
$>sudo in.tftpd -l /tftpboot
```





## 5、 テストする

```
$>touch abc
```

```
$>tftp 192.168.1.15 (192.168.1.15はボードのIP)
```

```
$>tftp> get abc
```

ダウンロードできればインストール成功。



## 付録四 WinCE 関連リソースの URL

(1) Windows Embedded CE 6.0 Platform Builder Service Pack 1

<http://www.microsoft.com/downloads/details.aspx?familyid=BF0DC0E3-8575-4860-A8E3-290ADF242678&displaylang=en>

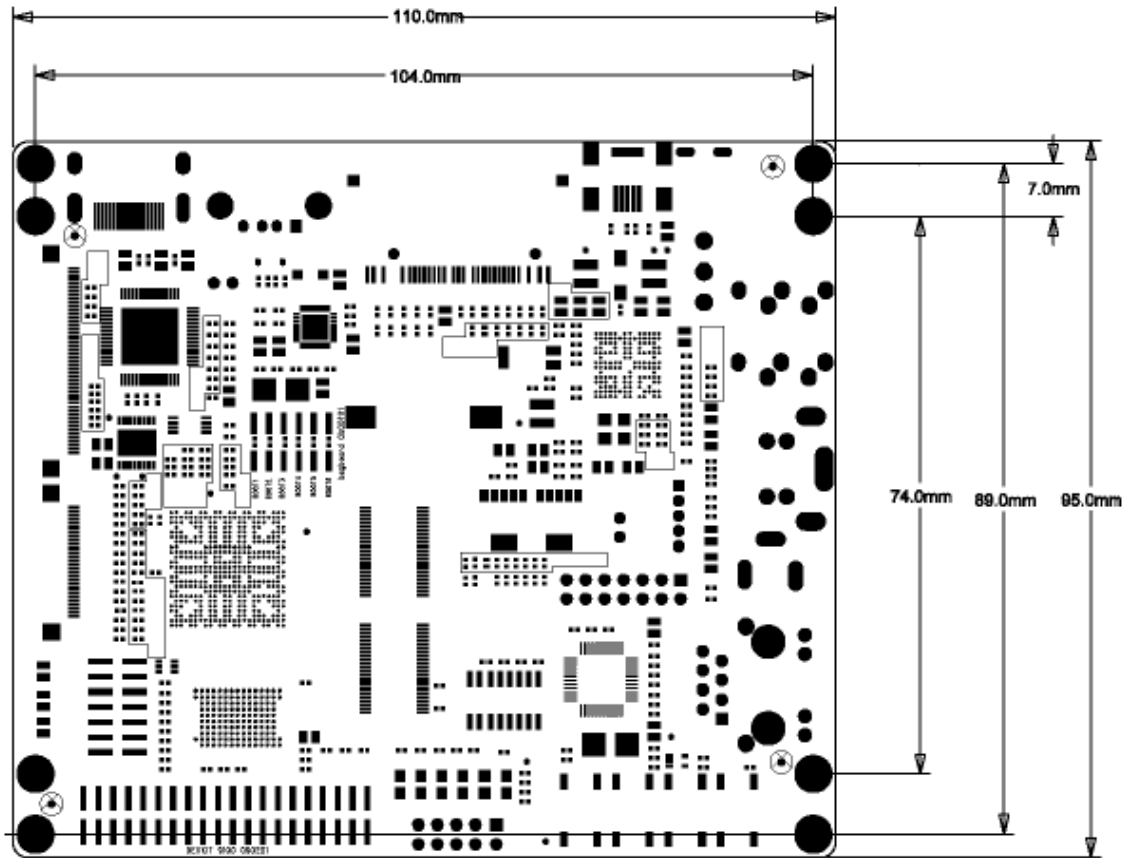
(2) Windows Embedded CE 6.0 R2

<http://www.microsoft.com/downloads/details.aspx?FamilyID=f41fc7c1-f0f4-4fd6-9366-b61e0ab59565&displaylang=en>

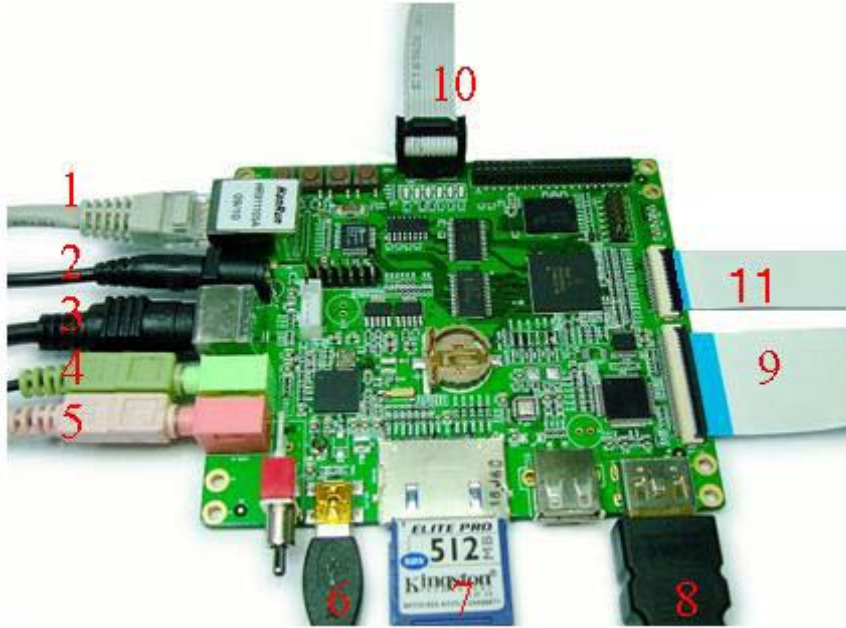
(3) Embedded CE 6.0 Platform Builder – Cumulative Product Update Rollup Package (through 12/31/2008)

<http://www.microsoft.com/downloads/details.aspx?FamilyID=b478949e-d020-465e-b451-73127b30b79f&DisplayLang=en>

## 付録五 ボードの寸法図



## 付録六 各デバイスの接続 I/F



- 1、 LAN
- 2、 電源
- 3、 S-Video
- 4、 Audio\_OUT
- 5、 Mic
- 6、 USB OTG
- 7、 SDカード
- 8、 HDMI
- 9、 LCD
- 10、 シリアルポート
- 11、 Cameraモジュール