## ■ Embedded Linux Hands-on Tutorial -- ZedBoard

Linux + Zynq に慣れるべく，2013 年と 2 年前の資料だけど，Embedded Linux Hands-on Tutorial -- ZedBoard をやってみる．

ISE は 14.4.
14.7 でやろうとしたら, PS の IP コアのバージョンがあがっているためか, そのままでは BitStream が生成できなかった.
OS は CentOS 6.2 を用意.
今日び CnetOS 6.2 のリポジトリはそうそうないので,

```
baseurl=http://vault.centos.org/6.2/os/$basearch/
```

とかを使う．(cf. http://d.hatena.ne.jp/tmatsuu/20120324/1332578375)
x86_64 版でインストールするけど，CodeSaurcy の都合で 32bit 実行 / 開発環境も必要.

```
sudo yum install compat-libstdc++-33-3.2.3-69.el6.i686
sudo yum install glibc-devel.i686 glibc-devel
```

とかした上で ISE をインストールする必要があるので注意.

## ■準備

設計イメージのダウンロードと展開

```
wget http://www.digilentinc.com/Data/Products/ZEDBOARD/ZedBoard_Linux_Design.zip
unzip ZedBoard_Linux_Design.zip
```

XPS で開く

```
source /opt/Xilinx/14.4/ISE_DS/settings64.sh
xps ZedBoard_Linux_Design/hw/xps_proj/system.xmp &
```

## ■ HW の変更と合成

PS の GPIO コアから LED をはずす

1. I/O Peripherals をクリック
2. Zynq PS MIO Configurations ダイアログが開く → EMIO GPIO の幅を 60 から 52 に変更 (8 LED ピンの削除 )
3. Ports タブを開いてポートの設定
    1. processing_system7_0 の (IO_IF) GPIO_0 を選択
    2. ポートを No Connection に
    3. ポートを External Ports に．名前の prefix の _pin を削除 ( 既存の UCF とできるだけあわせるため )

Create and Import Peripheral Wizard で myled を作る

・ コア名は myled.

- AXI4-Lite
- software reset と include data phase timer はオフ
- レジスタの数は 1 つ
- user_logic は ( 私の場合 )VHDL のままでいい

myled の追加と編集

- add IP．パラメタはデフォルトのまま
- ソースコードを修正．Browse HDL Sources... で HDL コードに，View MPD で mpd ファイルに簡単にアクセスできる
  - user_logic.vhd の修正

```
101  LED : out std_logic_vector(7 downto 0);
148  LED <= slv_reg0(7 downto 0);
```

  - myled.vhd の修正

```
141  LED : out std_logic_vector(7 downto 0);
306  LED => LED,
```

  - myled_v2_1_0.mpd の修正

```
40 PORT LED = "", DIR = O, VEC = [7:0]
```

- Project -> Rescan User Repositries で IP コアをリスキャン
- Ports タブで myled_0 を開くと追加した LED な出力ポートができているので外部出力ピンに
- UCF でピン割り当てを変更
  - Project タブの UCF File: data/system.ucf からアクセスできる
  - 80 行目付近からの On-board LED's の processing_system7_0_GPIO<> を myled_0 _LED_pin<> に変更.
  - 以降のペリフェラルに対する processing_sytem7_0_GPIO<> のインデックスを 8 減らす

Bit ファイルの生成

- Hardware->Genearete Bitstream

## ■ U-BOOT 作る

U-Boot のコンパイルの準備

git で取得するか Branch のアーカイブかを利用する.

```
git clone https://github.com/Digilent/u-boot-digilent
```

ドキュメントに沿ったバージョンのものをダウンロードしてみた.

```
wget https://github.com/Digilent/u-boot-digilent/archive/v2012.04-digilent-13.01.zip
```

展開して，もぐる

```
unzip v2012.04-digilent-13.01.zip
cd u-boot-digilent-2012.04-digilent-13.01
```

U-Boot のコンパイルのための設定

・ IP アドレスの設定．include/configs/zynq_zed.h を編集.

/* Default environment */
#define CONFIG_IPADDR　10.0.0.1
#define CONFIG_SERVERIP 10.0.0.3

ビルド

次のようにしてビルド.

```
make CROSS_COMPILE=arm-xilinx-linux-gnueabi- zynq_zed_config
make CROSS_COMPILE=arm-xilinx-linux-gnueabi-
```

できあがったものをコピー.
cp u-boot ../ZedBoard_Linux_Design/boot_image/u-boot.elf

## ■ FSBL を作って BOOT.BIN を作る

BOOT.BIN を作る ( 準備 )

( 閉じていれば )xps で xmp を再び開く.

```
xps ZedBoard_Linux_Design/hw/xps_proj/system.xmp &
```

Project → Export Hardware Design to SDK で Export しつつ SDK 起動.
ワークスペースは,

```
ZedBoard_Linux_Design/hw/xps_proj/SDK/SDK_Export
```

にある.
File ⇒ New ⇒ Project... で Xilinx の Application Project の作成を開始.
キモは,

```
Project Name: FSBL
Hardware Plat: xps_proj_hw_platform
OS Platform: standalone
```

くらいか．Available Templates では，Zynq FSBL を選択.

main.c の修正

ZedBoard では，FSBL で，USB-Reset ピンのトグルによって USB PHY チップをリセットする必要があるらしい.
main.c の FsbHandOff() の呼び出しの前 (472 行目 ) に次のコードを追加.

```
472
473   /* Reset the USB */
474    {
475         fsbl_printf(DEBUG_GENERAL, "Reset USB...\r\n");
476
477         /* Set data dir */
478         *(unsigned int *)0xe000a284 = 0x00000001;
479
480         /* Set OEN */
481         *(unsigned int *)0xe000a288 = 0x00000001;
482         Xil_DCacheFlush();
483         /* For REVB Set data value low for reset, then back high */
484 #ifdef ZED_REV_A
485         *(unsigned int *)0xe000a048 = 0x00000001;
486         Xil_DCacheFlush();
487         *(unsigned int *)0xe000a048 = 0x00000000;
488         Xil_DCacheFlush();
489 #else
490         *(unsigned int *)0xe000a048 = 0x00000000;
491         Xil_DCacheFlush();
492         *(unsigned int *)0xe000a048 = 0x00000001;
493         Xil_DCacheFlush();
494 #endif
495    }
```

デフォルトで自動 Build だけど，念の為に Clean して Build.

BOOT.BIN を作る

Xilinx Tools ⇒ Create Zynq Boot Image でツールを起動.
FSLB elf には,

```
ZedBoard_Linux_Design/hw/xps_proj/SDK/SDK_Export/FSBL/Debug/FSBL.elf
```

を選択.
List of partitions int the boot image は，FSBL.elf，system.bit，u-boot.elf の順.

```
ZedBoard_Linux_Design/hw/xps_proj/SDK/SDK_Export/FSBL/Debug/FSBL.elf
ZedBoard_Linux_Design/hw/xps_proj/SDK/SDK_Export/xps_proj_hw_platform/system.bit
ZedBoard_Linux_Design/boot_image/u-boot.elf
```

出力先のフォルダは,

```
ZedBoard_Linux_Design/boot_image/
```

にする.
できあがった u-boot.bin を BOOT.BIN にリネーム.

## ■ Linux カーネルのコンパイル

Linux カーネルのコンパイルの準備

git のメインブランチかタグを切られたものを持ってくる.
u-boot のときと同じように，v3.6-digilent-13.01 とタグを切られたものを持ってくることにする.
https://github.com/Digilent/linux-digilent/archive/v3.6-digilent-13.01.zip
wget で取得したら拡張子がつかなかったので末尾に .zip をつけて，unzip で展開してもぐる.

コンフィグ

まずはデフォルトコンフィギュレーション

```
make ARCH=arm CROSS_COMPILE=arm-xilinx-linux-gnueabi- digilent_zed_defconfig
```

で，menuconfig

```
make ARCH=arm CROSS_COMPILE=arm-xilinx-linux-gnueabi- menuconfig
```

ちなみに，ncurses-devel がなかったので yum でインストール ...
ドキュメントでは PmodOLED1 をビルトイン・ドライバからローダブルモジュールに変更している
具体的には Device Driver → PMOD Support と辿って，PmodOLED1 の '*' を 'M' に変更.

ビルド

Exit をつづけて終了したらビルド

```
make ARCH=arm CROSS_COMPILE=arm-xilinx-linux-gnueabi-
```

ビルドが終わったら arch/arm/boot/zImage ができている

## ■デバイスツリーを作る

カーネルソースの下で,

```
./scripts/dtc/dtc -I dts -O dtb -o ../devicetree.dtb arch/arm/boot/dts/digilent-zed.dts
```

を実行

## ■ SD カードにセットアップ

SD カードを用意

先頭 1GB を vfat，残りを ext4 にする.

- fdisk でパーティション作る
- vfat と ext4 で，それぞれフォーマット．たとえば,

```
suudo mkfs -t vfat -n ZED_BOOT /dev/sdb1
sudo mkfs -t ext4 -L ROOT_FS /dev/sdb2
```

ルートファイルシステムの用意

- Linaro をダウンロード.

```
wget http://releases.linaro.org/12.09/ubuntu/precise-images/ubuntu-desktop/linaro-precise-ubuntu-desktop-20120923-436.tar.gz
```

- 展開

```
mkdir -p /tmp/linaro
sudo cp linaro-precise-ubuntu-desktop-20120923-436.tar.gz /tmp/linaro
cd /tmp/linaro
sudo tar zxf fs.tar.gz
```

- ext4 領域をマウント

```
mkdir -p /tmp/sd_ext4
sudo mount /dev/sdb2 /tmp/sd_ext4
```

- 展開した一式をコピー．ドキュメントでは rsync 使ってる

```
cd binary/boot/filesystem.dir/
sudo rsync -a ./ /tmp/sd_ext4
sudo sync; sudo sync; sudo sync; # 念のため
```

- unmount して取り出す

```
sudo umount /tmp/sd_ext4
```

BOOT.BIN，dtc，zImage をコピー

vfat 領域に

- BOOT.BIN
- devicetree.dtb
- zImage

をコピーする.

## ■ ZedBoard で起動してみる

HDMI ケーブルつなぐと GUI があがってくるのがわかる.

```
apt-get install openssh-server
```

とかすると sshd がインストールできる.
ssh でログインできるように root のパスワードを適当に設定.

## ■ myled 用のドライバを作る

準備

カーネルソースにアクセスしやすいようにシンボリックリンクを用意

```
ln -s linux-digilent-3.6-digilent-13.01 linux-digilent
```

作業用のディレクトリを用意して移動

```
mkdir drivers
cd drivers
```

必要なファイルを書いて make

Makefile を書く

```
obj-m := myled.o

all:
 make -C ../linux-digilent/ M=$(PWD) modules


clean:
 make -C ../linux-digilent/ M=$(PWD) clean
```

を書いて，make.

```
make ARCH=arm CROSS_COMPILE=arm-xilinx-linux-gnueabi-
```

## ■デバイスツリーを更新

myled のアドレスを xps で確認．今回は 0x7e400000-0x7e40ffff の 64KB の空間だった.
サンプルをコピー

```
cp ../linux-digilent/arch/arm/boot/dts/digilent-zed.dts .
```

一番最後にエントリを追加.

```
myled {
        compatible = "dglnt,myled-1.00.a";
        reg = <0x7e400000 0x10000>;
};
```

編集したらデバイスツリーを作りなおす

```
../linux-digilent/scripts/dtc/dtc -I dts -O dtb -o devicetree.dtb digilent-zed.dts
```

システムに反映

scp で myled.ko と devicetree.dtb をコピー.
ブートパーティションはマウントされてないので，マウントしてコピー

```
mount /dev/mmcblk0p1 /mnt/
cp /root/devicetree.dtb /mnt/
```

で，リブート．
再起動したら，/proc/myled ができている．

```
insmod myled.ko
echo 0x0F > /proc/myled
echo 0xF0 > /proc/myled
```

とかして楽しむ

## ■ユーザアプリを書く

作業用ディレクトリをつくって，もぐる

```
mkdir user_app
cd user_app
```

を書く．Makefile も用意．ドキュメントの Makefile は何か変な感じだったので注意．

また，クロスコンパイル環境とターゲットでライブラリがちぐはぐなので -static が必要だった．

```
CC = arm-xilinx-linux-gnueabi-gcc
CFLAGS = -g -static

all : led_blink

led_blink : led_blink.c
  ${CC} ${CFLAGS} -o $@ $^
clean :
  rm -rfv *.o
  rm -rfv led_blink

.PHONY : clean
```

Linaro の場合，ホストコンパイラもあるので，ソースをコンパイルしても OK.