

XMOS の XS1-G4,L1 であそぶ - その 7

テスト基板 XC-1,XC-2,XK-1,XC-1A などであそんで気がついたことのメモを書いています

XTAG2 アダプタで独自ファームウェアを実行する (2012/10)

XMOS 純正の XTAG2 デバッグアダプタは、自身ではデバッグ用のファームウェアを持たず、PC への USB 接続後にホストからファームウェアを転送する(たとえば xrun コマンドの実行)ことによって JTAG デバッガの機能を実現しています。



つまり PC に刺した段階では OTP からブートローダが起動した状態になっているということらしいです。

このブートローダへのアクセスですが、XTAG2 の標準ドライバは libusb を使っているなので、libusb を使用したブートローダを作成することで、独自コードを実行することができます。

Application Note: Dynamic loading of USB Binaries onto the XTAG2

<http://www.xmos.com/published/dynamic-code-loader-xtag-2?version=latest>

ほぼ上記のアプリケーションノートの通りに実行することで実現可能なのですが、ハマったポイントがあるのでメモ。

まず、ブートローダにたいしてあらかじめビルド済みの XTAG2 用オリジナルファームウェアを転送するための PC 側のコードが必要です。

これは上記のアプリケーションノートにあるように、github で proj_xtag2 から一式もってきた中の run_dyanmic_xe フォルダにあるコードがそれです。

Windows 環境の場合には、libusb からスタブ (.lib) と (.h) を取ってくれば、VisualStudio2012 Express(無料、登録要版)で Win32 コンソールアプリとしてすんなりビルド、実行が可能で、何の問題もありません。

(ヘッダファイル名 usb.h を変える必要があるかもしれません)

app_l1_hid.h という名前のファイル内で int burnData [] という配列になっているものがブートローダに与えるバイナリで、runDynamic にはじめから含まれるものをそのまま使用すれば、HID マウスとして動作すると思います。

さて、オリジナルの app_l1_hid.h を生成するためのスクリプト例に問題があります。

まず、od と awk が必要です。アプリケーションノートでは OS X での開発を推奨していますが、Windows であれば Cygwin などからの実行が必要です。

(このページを見られている方ならなんでもないことかと思えます。)

Quartus(Altera) のコマンドラインも Cygwin なので私はこれを使いました。

すんなりいくかと思いきや、

myfirmware.xe を上記アプリケーションノートにしたがって変換

```
%xobjdump --split --strip myfirmware.xe
%od -t x4 image_n0c0.bin | awk 'BEGIN { print " int burnData [] = {}" END { print";"}' {for (i =2;i
<= NF;i ++) { print " 0x" $i ",,"}}' > app_l1_hid.h
```

すると、何度実行しても転送するとデバイスマネージャから XTAG-2 が消えてしまいます。

うーんうーんとうなって、問題を見つけました。

どうも od に -v オプションが必要らしいです。

```
%od -t x4 -v image_n0c0.bin | awk 'BEGIN { print " int burnData [] = {}" END { print";"}' {for (i =2
;i <= NF;i ++) { print " 0x" $i ",,"}}' > app_l1_hid.h
```

これで動作しました。

< < XMOS の XS1-G4.L1 であそぶ - その 6 XMOS の XCORE であそぶ (2014/12) > >