

# MACGYVERQUE //

# #INTERNETTRAP

I'm pretty good with computers...

---

[WHERE IS USN SASEBO JAMES JONES?](#) [EXPERTISE IN SW DEVELOPMENT](#) [ABOUT ME](#) [HOME](#)

---

OCT 09 2013

LEAVE A COMMENT

UNCATEGORIZED

## XILINX SP606 FPGA + MICROBLAZE + LWIP // SPI FLASH BOOTLOADER GENERATION

### General Project Setup

One of the most important points here is that if you're attempting to create an embedded system is that your linker script is setup properly. When using LWIP, I had to increase the heap and stack size of my main application to 0x0A00 as seen below.

### Linker Script: lscript.ld

A linker script is used to control where different sections of an executable are placed in memory. In this page, you can define new memory regions, and change the assignment of sections to memory regions.

#### Available Memory Regions

| Name                   | Base Address | Size       |
|------------------------|--------------|------------|
| ilmb_cntlr_dlm_b_cntlr | 0x00000050   | 0x00007FB0 |
| FLASH_MEM0_BASEADDR    | 0x86000000   | 0x02000000 |
| MCB_DDR3_MPMC_BASEADDR | 0x88000000   | 0x08000000 |

#### Stack and Heap Sizes

Stack Size

Heap Size

#### Section to Memory Region Mapping

| Section Name | Memory Region          |
|--------------|------------------------|
| .text        | MCB_DDR3_MPMC_BASEADDR |
| .init        | MCB_DDR3_MPMC_BASEADDR |
| .fini        | MCB_DDR3_MPMC_BASEADDR |
| .ctors       | MCB_DDR3_MPMC_BASEADDR |
| .dtors       | MCB_DDR3_MPMC_BASEADDR |
| .rodata      | MCB_DDR3_MPMC_BASEADDR |
| .sdata2      | MCB_DDR3_MPMC_BASEADDR |
| .sbss2       | MCB_DDR3_MPMC_BASEADDR |
| .data        | MCB_DDR3_MPMC_BASEADDR |
| .got         | MCB_DDR3_MPMC_BASEADDR |
| .got1        | MCB_DDR3_MPMC_BASEADDR |
| .got2        | MCB_DDR3_MPMC_BASEADDR |

Summary Source

### Bootloader Generation

From the SDK, create a default bootloader (srec\_bootloader)

Here's its linker script...

### Linker Script: lscript.ld

A linker script is used to control where different sections of an executable are placed in memory. In this page, you can define new memory regions, and change the assignment of sections to memory regions.

#### Available Memory Regions

| Name                   | Base Address | Size       |
|------------------------|--------------|------------|
| ilmb_cntlr_dlm_b_cntlr | 0x00000050   | 0x00007FB0 |
| FLASH_MEM0_BASEADDR    | 0x86000000   | 0x02000000 |
| MCB_DDR3_MPMC_BASEADDR | 0x88000000   | 0x000FA000 |
| DDR3_PULSE_TABLE       | 0x880FA000   | 0x7406000  |
| SCRATCH_MEMORY         | 0x8f500000   | 0xAFFFFF   |

#### Stack and Heap Sizes

Stack Size

Heap Size

#### Section to Memory Region Mapping

| Section Name | Memory Region          |
|--------------|------------------------|
| .text        | ilmb_cntlr_dlm_b_cntlr |
| .init        | ilmb_cntlr_dlm_b_cntlr |
| .fini        | ilmb_cntlr_dlm_b_cntlr |
| .ctors       | ilmb_cntlr_dlm_b_cntlr |
| .dtors       | ilmb_cntlr_dlm_b_cntlr |
| .rodata      | ilmb_cntlr_dlm_b_cntlr |
| .sdata2      | ilmb_cntlr_dlm_b_cntlr |
| .sbss2       | ilmb_cntlr_dlm_b_cntlr |
| .data        | ilmb_cntlr_dlm_b_cntlr |
| .got         | ilmb_cntlr_dlm_b_cntlr |

Ensure that in your blconfig.h file you have the following line

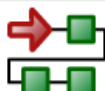
```
#define FLASH_IMAGE_BASEADDR 0x87200000
```

This tells the bootloader where to begin pulling the SREC data (your actual executable application)

Program the FPGA w/ the system bit file and the srec bootloader.

### Program FPGA

Specify the bitstream and the ELF files that reside in BRAM memory



#### Hardware Configuration

Hardware Specification: /home/ztswdev/workspace/XILINX/RLTS\_Hardware\_Platform/system.xml

Bitstream:

BMM File:

#### Software Configuration

| Processor    | ELF File to Initialize in Block RAM   |
|--------------|---|
| microblaze_0 | <input type="text" value="/workspace/XILINX/srec_bootloader_0/Debug/srec_bootloader_0.elf"/> <input type="button" value="v"/> |

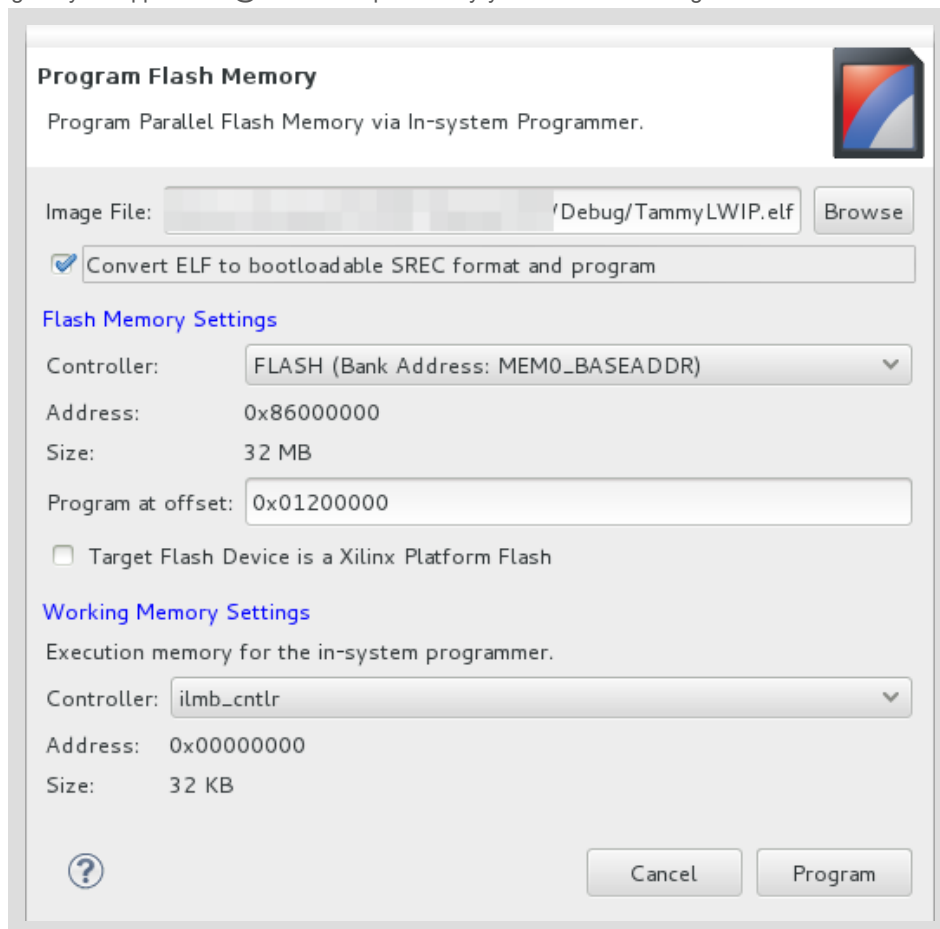
Take the download.bit file from the hardware platform

Run the following commands

```
impact -batch make_bpi_up.impact  
xmcsutil -accept_notice -18 pi outfile.hex -o bootloader.bin
```

(The last command above creates a bootable image w/ your FPGA bit file and the bootloader)

Program your application @ the offset specified by your bootloader using XSDK...



Next, program the bootloader @ 0x0 using XSDK

Note, when you're selecting "Program Flash" from XSDK, you'll need to browse to the bootloader.bin and select \* as opposed to ".bin;.elf;\*.srec" in the file type filter in the browser window so you'll be able to select the bootloader.bin file itself.

### Program Flash Memory

Program Parallel Flash Memory via In-system Programmer.

Image File:  [Browse](#)

☐ Convert ELF to bootloadable SREC format and program

#### Flash Memory Settings

Controller:

Address:

Size:

Program at offset:

☐ Target Flash Device is a Xilinx Platform Flash

#### Working Memory Settings

Execution memory for the in-system programmer.

Controller:

Address:

Size:

[?](#) [Cancel](#) [Program](#)

Let me know if you need me to clarify any of the steps in the comments below

The make\_bpi\_up.impact file consists of the following

```
setMode -pff
setSubmode -pffparallel
setPreference -pref StartupClock:Auto_Correction
addPromDevice -p 1 -size 32768
addDesign -version 0 -startaddress 000000
addDeviceChain -index 0
addDevice -p 1 -file download.bit
generate -format hex -fillvalue FF -output outfile
quit
```



## Leave a Reply

Your email address will not be published. Required fields are marked \*

Name \*

Email \*

Website

Comment

Post Comment

---

Proudly powered by [WordPress](#) Theme: [Chunk](#) by [Automattic](#).