The Definitive Guide to Hardware/Software Interfacing with the XUP-Virtex2P-Pro Development System

Philip Amberg and Andrew Giles
4/14/2007

Introduction

In this guide we will walk through setting up a simple hardware/software system to count to 8 on the Virtex board LED's.

Guide

Lets start with the easy part!

1. Create a new XPS project using base system builder.
2. Name the project LEDCOUNT
3. Enter the path to the board repository in the designated area.
4. Select "I would like to create a new design"
5. Select the XUP Virtex-2 Pro Development System, Board Revision C and press next
6. Select PowerPC and press next
7. Select a processor clock frequency of 300 MHz and a bus frequency of 100 MHz
8. Set instruction and data memories to 16 KB and press next
9. Deselect all peripherals except RS232 and press next
10. Make sure PLB_BRAM is set to 16 KB and press next
11. Disable the memory test and press next
12. Press generate
13. Press finish to create your project
14. In XPS Studio, select Hardware>Create or Import Peripheral
15. Press next
16. Press next
17. Press next
18. Name your peripheral ledlink then press next
19. Select Onchip peripheral bus (OPB) then press next
20. Deselect everything except User logic S/W register support and press next
21. Ensure number of software accesable registers is 1, data with is 32 bit, and enable post write behavior is selected then press next
22. Press next
23. Press next
24. Select generate stub logic in verilog instead of VHDL then press next
25. Press finish to generate the skeleton of the peripheral.

Now comes the fun part!

26. In XPS, press File>Open, and navigate to the folder for your LEDCOUNT project.
27. Click on pcores, then ledlink_v1_00_a, then hdl, then verilog, and open user_logic.v
28. Where the file says ADD USER PORTS BELOW THIS LINE, add the following code:
    ```
    led_out,
    ```
29. Scroll down to the next ADD USER PORTS BELOW THIS LINE section and add the following code:
    ```
    output [0:3] led_out;
    ```
30. Scroll down to the code section labeled "user net declarations here" (~line 109)
31. Add the following code:
    ```
    reg [0:3] led_out;
    ```
32. Scroll down to the code section labeled "implement slave model register(s)" (~line 147)
33. In the case statement, under case 1'b1, comment out the code already existing there and add the following code:
    ```
    led_out <= Bus2IP_Data[28:31]; //(note the little
    endianness)
    ```
34. Save user_logic.v and close it
35. In XPS, press File>Open, and navigate to pcores\ledlink_v1_00_a\hdl\vhdl and open ledlink.vhd
36. Scroll down to ~line 118 where it says ADD USER PORTS BELOW THIS LINE and add the following code:
    ```
    led_out        : out std_logic_vector(0 to 3);
    ```
37. Now scroll down to ~line 308 within component user_logic is.
38. Find where it says ADD USER PORTS BELOW THIS LINE and add the following code:
    ```
    led_out        : out std_logic_vector(0 to 3);
    ```
39. Now scroll down to ~line 428 within USER_LOGIC_I: component user_logic
40. Find where it says MAP USER PORTS BELOW THIS LINE and add the following code:
    ```
    led_out => led_out,
    ```
41. Save the file ledlink.vhd and close it

Now were going to get really tricky!

42. In XPS, press File>Open
43. Navigate to pcores\ledlink_v1_00_a\data and open ledlink_v2_1_0.mpd
44. Scroll to the bottom of the file and add the following code under ##Ports but before END:
    ```
    PORT led_out = "", DIR=O, VEC=[0:3]
    ```
45. Save the file ledlink_v2_1_0.mpd and close it

46. In XPS, click on the tab labeled IP catalog, expand project repository, right click on ledlink and click add IP.
47. In the system assembly view you will see ledlink_0 added to the list of system components.
48. Expand this entry and change bus connection to new connection.
49. Press File>Close Project, then File>Open Project and select your project. Yay xilinx tools!
50. When your project opens, go back to the system assembly view, click the ports button under filters at the top of the system assembly view.
51. Expand the ledlink entry. You will see the led_out signal you added before. Under net, change the connection to make external. If you now expand External Ports, you will see your ledlink port listed in here.
52. Now select addresses from the filters and click generate addresses.
53. In XPS, press File>Open and navigate to the data folder in your project and open the file ledcount.ucf.
54. Scroll to the end of the file and add the following code to define pinouts
```
Net ledlink_0_led_out_pin<0> LOC=AC4;
Net ledlink_0_led_out_pin<0> IOSTANDARD=LVTTL;
Net ledlink_0_led_out_pin<0> SLEW=SLOW;
Net ledlink_0_led_out_pin<0> DRIVE=12;

Net ledlink_0_led_out_pin<1> LOC=AC3;
Net ledlink_0_led_out_pin<1> IOSTANDARD=LVTTL;
Net ledlink_0_led_out_pin<1> SLEW=SLOW;
Net ledlink_0_led_out_pin<1> DRIVE=12;

Net ledlink_0_led_out_pin<2> LOC=AA6;
Net ledlink_0_led_out_pin<2> IOSTANDARD=LVTTL;
Net ledlink_0_led_out_pin<2> SLEW=SLOW;
Net ledlink_0_led_out_pin<2> DRIVE=12;

Net ledlink_0_led_out_pin<3> LOC=AA5;
Net ledlink_0_led_out_pin<3> IOSTANDARD=LVTTL;
Net ledlink_0_led_out_pin<3> SLEW=SLOW;
Net ledlink_0_led_out_pin<3> DRIVE=12;
```

55. Save the file ledcount.ucf and close it

Now for some software!

56. In XPS, find the Applications tab, click it, then click Add Software Application Project.
57. Name the project ledcode and select processor ppc405_0.
58. Under your newly created project, right click on sources and click add new file. Name it ledcode_main.c
59. Paste the following the code into the file:
```
//include files
```

```
//define io devices
//#include "stdio.h"

#include "xparameters.h"

#include "stdlib.h"

//define general purpose io function for
accessing peripherals
//include testy.h for the test device:
#include "testy.h"

//define printf
#define printf xil_printf

int main(void)
{
     //do stuff here.
     int x;
     x = 0;
     int y;
     while(1)
     {
          x= x+ 0x00000001;
          if (x == 0x0000000F)
               x = 0;

     TESTY_mWriteSlaveReg0(XPAR_TESTY_0_BASEADDR,
x);
          y = 0;
          while(y<25000000)
          {
               y++;
          }
     }
}
```

60. Save the file
61. Right click on the Project:ledcode header and select mark to initialize BRAMs
62. Click Softare>Generate Linker Script then click Generate
63. Click Device Configuration>Download Bitstream

Interfacing with Coregen Components

Interfacing with

Guide

1. Follow steps 1-25 of the above guide. Name your project something other than LEDCOUNT. Call the peripheral blockmem.
2. Open Xilinx Project Navigator.
3. In project navigator, click File>Open Project then navigate to your saved XPS project. Once there, navigate to \pcores\"your peripheral name and version number"\devl\projnav then open the .ise file.
4. When the project opens, click Project>New Source, select IP Coregen, call the module blk_mem and press next.
5. Navigate to Memories and Storage Elements, RAMS and ROMS, and select dual port block memory, then press next twice.
6. On the block memory configuration screen that now appears, ensure the Width is set to 32, and the depth is set to 2. This creates a block memory with 2, 32 bit words. Set port A to write only and port B to read only.
7. Press next, press next, then click generate.
8. Open the userlogic.v file.
9. First we must add a clock signal, in the module (~line 55) add the following line of code:
   ```
   clk,
   ```
10. Around line 86, where the code says ADD USER PORTS BELOW THIS LINE add the following code:
    ```
    input clk;
    ```
11. Around line 110, where the code says USER nets declarations added here, add the following code:
    ```
    reg [0:31] data_in;
    wire [0:31] data_out;
    ```
12. In the case statement, under case 1'b1 (~line 162), comment out the code already existing there and add the following code:
    ```
    begin
        data_in<=Bus2IP_Data;
    end
    ```

13. Around line 172, right outside that particular always block, instantiate the block memory with the following code:
    ```
    blk_mem data_store(1'b1, 1'b1, clk, clk, data_in, data_out, 1'b1);
    ```
14. Around line 189 where it says assign IP2Bus_Data = slv_ip2bus_data;, change slv_ip2bus_data to data_out.
15. Save userlogic.v,
16. Press File>Open and open blkmem.vhd in the vhdl folder.
17. Around line 120 where the code says USER ports added here, add the following code:
    ```
    clk:in std_logic
    ```

18. Around line 310 where the code says USER ports added here, add the following code:

    ```
    clk:in std_logic
    ```
19. Around line 430 where the code says USER ports mapped here, add the following code

    ```
    clk=>clk,
    ```
20. Save blkmem.vhd
21. close project navigator and go back to XPS.
22. In XPS, ensure your project is open, then click Hardware>Create or Import Peripheral.
23. On the window that pops up, select import existing peripheral then click next.
24. Select to an XPS project then click next
25. In enter name of the top VHDL entry or Verilog module of your peripheral, enter blkmem. Select use version and use 1.00.b, then press next
26. Select HDL source files and netlist files then press next
27. Under HDL language used to implement your peripheral, select mixed
28. Then select use an XST project file, click browse, navigate to blkmem_v1.00a\devl\synthesis and open blkmem_xst.prj then press next
29. On this next window you should see a list of all the source files in your project, verify userlogic.v and blkmem.vhd, are there
30. Click add files, navigate to blkmem_v1.00a\devl\projnav and add blk_mem.v This adds blk_mem.v to the bottom of the list. Select it and press move up and move it above blkmem.vhd. Click next
31. On the bus interfaces window, select OPB Slave (SOPB). Click next
32. Click next
33. Click next
34. Deselect select and configure interrupts. Click next
35. Click next
36. On the netlist window, press select files, navigate to projnav folder, open blk_mem.edn
37. Click next, and then click finish to generate the peripherial.
38. In XPS, in the IP Catalog under Project Repository, blkmem 1.00b should be there. Right click on it and select add IP.
39. In the System Assembly View, find blkmem_0, expand it, and connect it to the OPB by selecing opb under bus connection.
40. Then in the system assembly view, go to addresses and click generate addresses. If it fails, manually assign the base address for blkmem_0 to 0x70000000 and click generate addresses again.
41. Now go to ports, expand blkmem_0, select clk and under net change it to sys_clk_s.
42. Now in XPS, go to the applications tab and click add software application project and give your project a name.
43. Now click on the project tab and open the MSS file BlkMem.mss, scroll down to the bottom and check the parameter DRIVER_VER for blkmem. If it is 1.00.b, change it to 1.00.a and save the file

44. Click on the applications tab and make a new c source file. Copy paste the following code into this file:

```
//BLKMEMTEST
//5.11.07
//Andrew Giles and Phil Amberg

//#include files
#include "xparameters.h"
#include "xgpio.h"
#include "blkmem.h"

#define printf xil_printf

int main(void)
{
   int data;
   int readdata;

   data = 2;
   while(1)
   {

   //write data to the block memory
   BLKMEM_mWriteSlaveReg0(XPAR_BLKMEM_0_BASEADDR,
data);

   readdata =
BLKMEM_mReadSlaveReg0(XPAR_BLKMEM_0_BASEADDR);
   printf("data Read Value = %X \r\n", readdata);

   //count the number of ones
   data++;
   //

   for(i = 0; i < 30000000; i++)
      {
      }
   }

}
```

45. Save the file, generate a linker script and download it to the board.
46. That is it! Congratulations on your Hardware/Software system.