Hardware/Software Codesign Lab 5

Student Name: Jose Sotelo

Student ID: 013969681

1. Follow the Lab 5 manual to finish Lab 5 and perform the following two demonstrations to your instructor:
   1. Program FPGA and download software application to the board to verify operations on hardware.
   2. Demonstrate step 4: Launch Debugger and debug
2. Copy and paste the following information to the end of this document:
3. Lab5.c

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88 | #include "xparameters.h"  #include "xgpio.h"  #include "led\_ip.h"  *// Include xscutimer header file*  #include "xscutimer.h"  *//====================================================*  XScuTimer Timer; */\* Cortex A9 SCU Private Timer Instance \*/*  #define ONE\_TENTH 32500000 *// half of the CPU clock speed/10*  **int** main (**void**)  {  XGpio dip, push;  **int** psb\_check, dip\_check, dip\_check\_prev, count, Status;  *// PS Timer related definitions*  XScuTimer\_Config \*ConfigPtr;  XScuTimer \*TimerInstancePtr = &Timer;  xil\_printf("-- Start of the Program --\r\n");    XGpio\_Initialize(&dip, XPAR\_SWITCHES\_DEVICE\_ID);  XGpio\_SetDataDirection(&dip, 1, 0xffffffff);    XGpio\_Initialize(&push, XPAR\_BUTTONS\_DEVICE\_ID);  XGpio\_SetDataDirection(&push, 1, 0xffffffff);  count = 0;    *// Initialize the timer*  ConfigPtr = XScuTimer\_LookupConfig(XPAR\_PS7\_SCUTIMER\_0\_DEVICE\_ID);  Status = XScuTimer\_CfgInitialize(TimerInstancePtr, ConfigPtr, ConfigPtr->BaseAddr);  **if**(Status != XST\_SUCCESS)  {  xil\_printf("Timer init() failed\r\n");  **return** XST\_FAILURE;  }  *// Read dip switch values*  dip\_check\_prev = XGpio\_DiscreteRead(&dip, 1);  *// Load timer with delay in multiple of ONE\_TENTH*  XScuTimer\_LoadTimer(TimerInstancePtr, ONE\_TENTH\*dip\_check\_prev);  *// Set AutoLoad mode*  XScuTimer\_EnableAutoReload(TimerInstancePtr);  *// Start the timer*  XScuTimer\_Start(TimerInstancePtr);  **while** (1)  {  *// Read push buttons and break the loop if Center button pressed*  psb\_check = XGpio\_DiscreteRead(&push, 1);  **if**(psb\_check > 0)  {  xil\_printf("Push button pressed: Exiting\r\n");  XScuTimer\_Stop(TimerInstancePtr);  **break**;  }  dip\_check = XGpio\_DiscreteRead(&dip, 1);  **if** (dip\_check != dip\_check\_prev)  {  xil\_printf("DIP Switch Status %x, %x\r\n", dip\_check\_prev, dip\_check);  dip\_check\_prev = dip\_check;  *// load timer with the new switch settings*  XScuTimer\_LoadTimer(TimerInstancePtr, ONE\_TENTH\*dip\_check);  count = 0;  }  **if**(XScuTimer\_IsExpired(TimerInstancePtr))  {  *// clear status bit*  XScuTimer\_ClearInterruptStatus(TimerInstancePtr);  *// output the count to LED and increment the count*  LED\_IP\_mWriteReg(XPAR\_LED\_IP\_S\_AXI\_BASEADDR, 0, count);  count++;  }  }  **return** 0;  } |

1. Answer the following questions:
   1. What is the prescale value of the private timer used in this lab?

**#define** XSCUTIMER\_CONTROL\_PRESCALER\_MASK 0x0000FF00U

* 1. What is the minimum time interval and maximum time interval controlled by the dip switch in this lab? Please show your calculation.

**#define** XPAR\_PS7\_CORTEXA9\_0\_CPU\_CLK\_FREQ\_HZ 666666687

Minimum = 1\*((0.5\*666666687)/10)

Maximum = 15\*((0.5\*666666687)/10)

* 1. List timer driver calling sequence.

1. Add the include file “xscutimer.h”

2. Add PS timer related definitions.

3. Initialize the timer using the XScuTimer\_LookUpConfig and XScuTimer\_CfgInitialize function

4. Load timer with delay using the XScuTimer\_LoadTimer function

5. Set AutoLoad mode using the XScuTime\_EnableAutoReload function

6. Start the timer using the XScuTimer\_Start function

7. Load timer with a new setting depending on switch or dip switch using the XScuTimer\_LoadTimer function

8. Check for when the timer has expired using the XScuTimer\_IsExpired function

9. Clear the status bit using the XScuTimer\_ClearInterruptStatus function