COEN 317: MICROPROCESSOR-BASED SYSTEM

Section UN-X

LAB REPORT # 3: Using the AXI Hardware Timer

INSTRUCTOR:
Dr. Fadi El-Hassan

Lab Coordinator:
Ted Obuchowicz

Submitted by:
Jaskirat Kaur
40138320

“I certify that this submission is my original work and meets the Faculty's Expectations of Originality”

Date Written: Monday, March 18, 2024

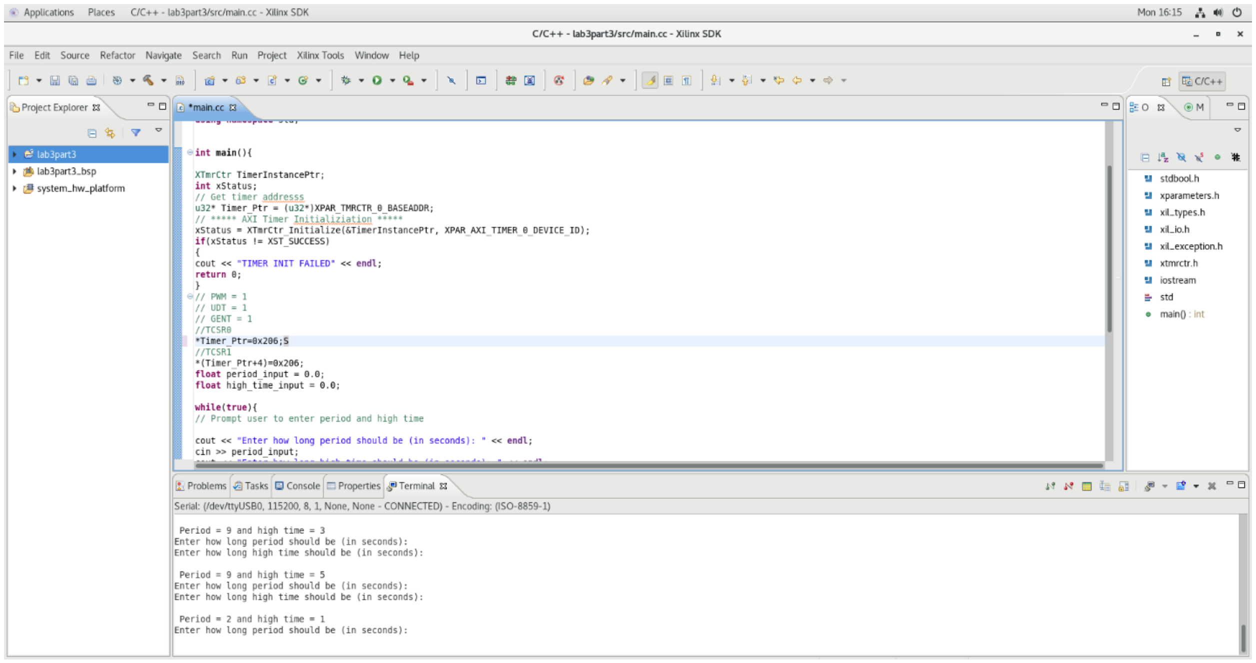
Objectives

The objectives of the lab are to become familiar with the AXI timer on the Xilinx ZC702 development board. Additionally, it is also to gain experience with the PlanAhead tools, the Xilinx Platform Studio, and the SDK.

Introduction

In this part, we are asked to implement the PWM mode for the AXI Timer. Following the small changes mentioned in the lab, I made it to the SDK where I wrote the program for the main. To write the program, I referred to the pseudo code given in part 2 for the order of the process, the equations to calculate the period and the high time given in the lab manual, and the pseudo code given in part 3 for what the procedure should be. Additionally, I referred to the AXI Timer DS764 manual to find which bits I need to set and which registers to use.

In the end, I manage



```
int main(){

XTmrCtr TimerInstancePtr;
int xStatus;
// Get timer addresss
u32* Timer_Ptr = (u32*)XPAR_TMRCTR_0_BASEADDR;
// ***** AXI Timer Initializiation *****
xStatus = XTmrCtr_Initialize(&TimerInstancePtr, XPAR_AXI_TIMER_0_DEVICE_ID);
if(xStatus != XST_SUCCESS)
{
cout << "TIMER INIT FAILED" << endl;
return 0;
}
// PWM = 1
// UDT = 1
// GENT = 1
//TCSR0
*Timer_Ptr=0x206;S
//TCSR1
*(Timer_Ptr+4)=0x206;
float period_input = 0.0;
float high_time_input = 0.0;

while(true){
// Prompt user to enter period and high time

cout << "Enter how long period should be (in seconds): " << endl;
cin >> period_input;
```

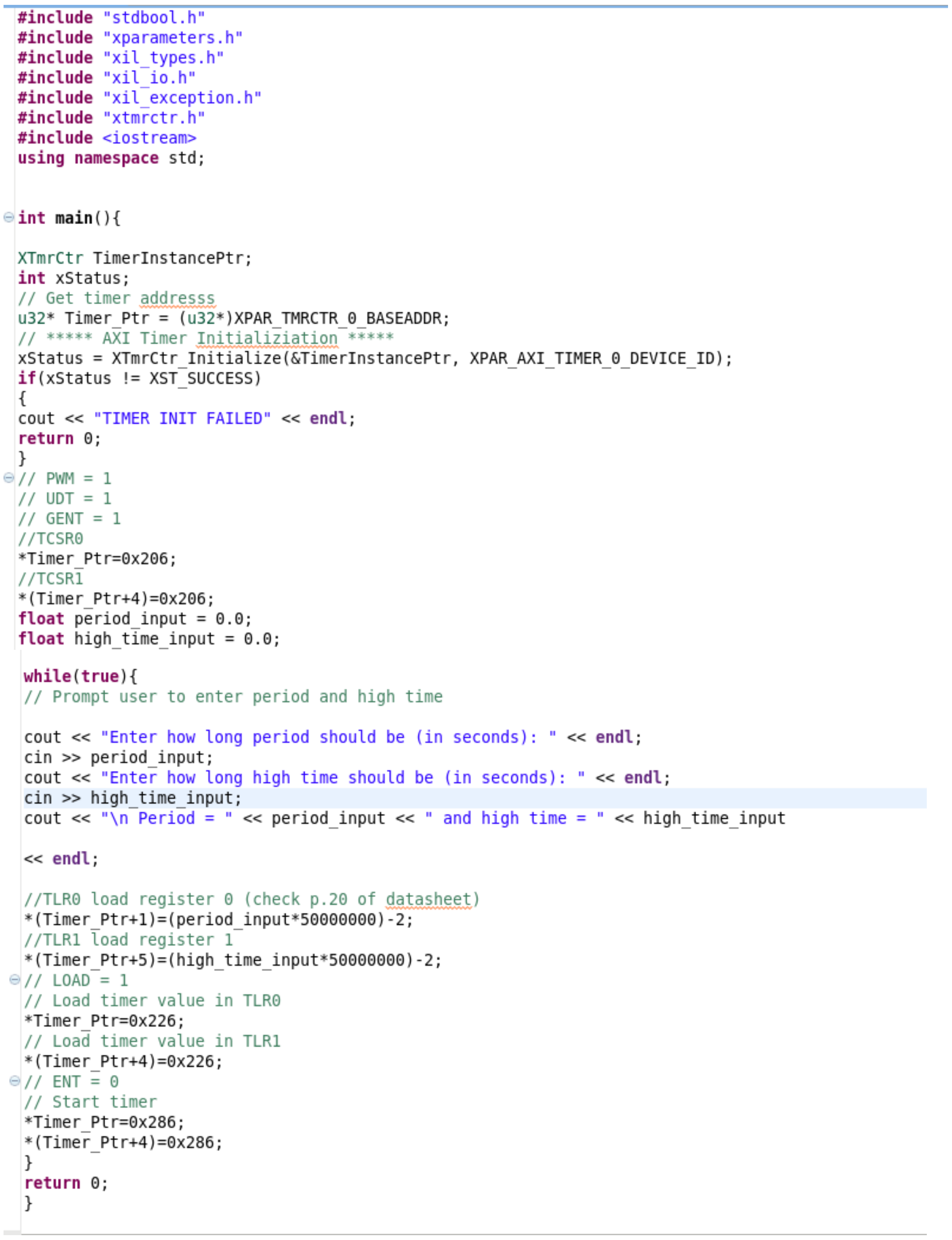
Serial: (/dev/ttyUSB0, 115200, 8, 1, None, None - CONNECTED) - Encoding: (ISO-8859-1)

```
 Period = 9 and high time = 3
Enter how long period should be (in seconds):
Enter how long high time should be (in seconds):

 Period = 9 and high time = 5
Enter how long period should be (in seconds):
Enter how long high time should be (in seconds):

 Period = 2 and high time = 1
Enter how long period should be (in seconds):
```

Conclusion:

To sum up, the lab's goals were successfully achieved. Although I encountered some technical hurdles, such as the SDK's limitations on inputting values and IMPACT hindering the completion of the Capture mode, I was nonetheless able to gain a comprehensive understanding of the AXI Timer's operation, familiarizing myself with its registers and various control/status options.

Code:



```
#include "stdbool.h"
#include "xparameters.h"
#include "xil_types.h"
#include "xil_io.h"
#include "xil_exception.h"
#include "xtmrctr.h"
#include <iostream>
using namespace std;


int main(){

XTmrCtr TimerInstancePtr;
int xStatus;
// Get timer addresss
u32* Timer_Ptr = (u32*)XPAR_TMRCTR_0_BASEADDR;
// ***** AXI Timer Initializiation *****
xStatus = XTmrCtr_Initialize(&TimerInstancePtr, XPAR_AXI_TIMER_0_DEVICE_ID);
if(xStatus != XST_SUCCESS)
{
cout << "TIMER INIT FAILED" << endl;
return 0;
}
// PWM = 1
// UDT = 1
// GENT = 1
//TCSR0
*Timer_Ptr=0x206;
//TCSR1
*(Timer_Ptr+4)=0x206;
float period_input = 0.0;
float high_time_input = 0.0;

while(true){
// Prompt user to enter period and high time

cout << "Enter how long period should be (in seconds): " << endl;
cin >> period_input;
cout << "Enter how long high time should be (in seconds): " << endl;
cin >> high_time_input;
cout << "\n Period = " << period_input << " and high time = " << high_time_input

<< endl;

//TLR0 load register 0 (check p.20 of datasheet)
*(Timer_Ptr+1)=(period_input*50000000)-2;
//TLR1 load register 1
*(Timer_Ptr+5)=(high_time_input*50000000)-2;
// LOAD = 1
// Load timer value in TLR0
*Timer_Ptr=0x226;
// Load timer value in TLR1
*(Timer_Ptr+4)=0x226;
// ENT = 0
// Start timer
*Timer_Ptr=0x286;
*(Timer_Ptr+4)=0x286;
}
return 0;
}
```