Course Name: _EE332/493 Embedded Systems Hardware/Software Fall 2019__

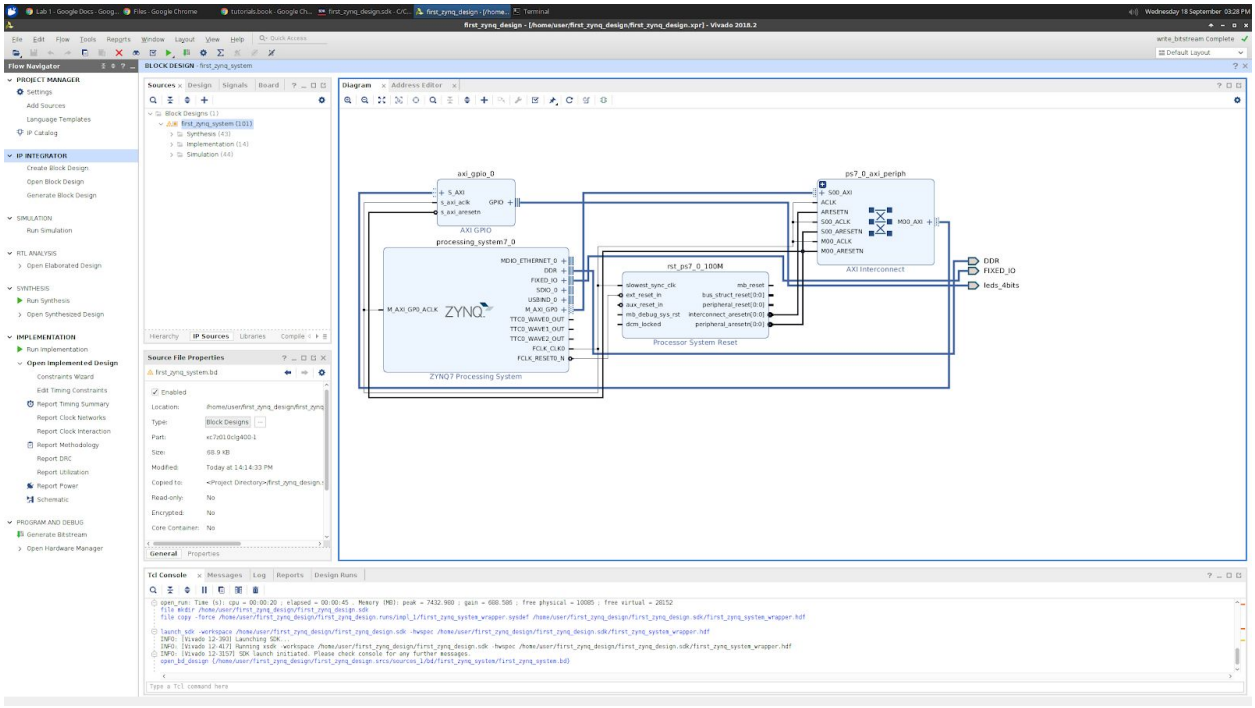
Experiment: [Lab#1]

Date Performed: 9/11/19

Date Submitted: 9/18/19

Submitted by: William Barron 172005966

- Purpose
    - The main purpose of this lab was to get us used to launching and using Vivado. This was an introductory lab that is meant to give us the experience to be able to start up the next lad much faster. This lab will show us the basics of making an embedded system and how to incorporate this system into hardware.
- Theory of operation
    - 1A
        - Here we simply mean to create a new project in Vivado. Throughout each step in this part there shouldn't be any confusion or problems. This part is just starting up and creating a project, following the instructions step by step. I had expected this part to be easy and straightforward.
    - 1B
        - Much like part A, part B is also straightforward and doesn't leave much to interpretation. Here I was able to follow the tutorial step by step and was able to create a simple Zynq embedded system. Theoretically this embedded system would implement a GPIO controller for the Zybo device.
    - 1C
        - Here we want to see our program actually work. This part is supposed to show us how everything we've done comes together and works.
- Block Diagram:



- Design
    - What follows is the given C code we were given in LED_test_tut_1C.c

```
/*
 * LED_test.c
```

```
 *
 *  Created on:          13 June 2013
 *      Author:    Ross Elliot
 *     Version:            1.2
 */

/*****************************************************************************************
* VERSION HISTORY
*****************************************************************************************
*         v1.2 - 13 February 2015
*                 Modified for Zybo Development Board ~ DN
*
*         v1.1 - 27 January 2014
*                 GPIO_DEVICE_ID definition updated to reflect new naming conventions
in Vivado 2013.3
*                 onwards.
*
*         v1.0 - 13 June 2013
*                 First version created.
*****************************************************************************************/

/*****************************************************************************************
 * This file contains an example of using the GPIO driver to provide communication
between
 * the Zynq Processing System (PS) and the AXI GPIO block implemented in the Zynq
Programmable
 * Logic (PL). The AXI GPIO is connected to the LEDs on the Zybo.
 *
 * The provided code demonstrates how to use the GPIO driver to write to the memory
mapped AXI
 * GPIO block, which in turn controls the LEDs.
 *****************************************************************************************/

/* Include Files */
#include "xparameters.h"
#include "xgpio.h"
#include "xstatus.h"
#include "xil_printf.h"

/* Definitions */
#define GPIO_DEVICE_ID  XPAR_AXI_GPIO_0_DEVICE_ID       /* GPIO device that
LEDs are connected to */
```

```
#define LED 0x9
/* Initial LED value - X00X */
#define LED_DELAY 10000000                                              /*
Software delay length */
#define LED_CHANNEL 1                                                   /*
GPIO port for LEDs */
#define printf xil_printf                                       /* smaller, optimised
printf */

XGpio Gpio;
/* GPIO Device driver instance */

int LEDOutputExample(void)
{

        volatile int Delay;
        int Status;
        int led = LED; /* Hold current LED value. Initialise to LED definition */

                /* GPIO driver initialisation */
                Status = XGpio_Initialize(&Gpio, GPIO_DEVICE_ID);
                if (Status != XST_SUCCESS) {
                        return XST_FAILURE;
                }

                /*Set the direction for the LEDs to output. */
                XGpio_SetDataDirection(&Gpio, LED_CHANNEL, 0x0);

                /* Loop forever blinking the LED. */
                        while (1) {
                                /* Write output to the LEDs. */
                                XGpio_DiscreteWrite(&Gpio, LED_CHANNEL, led);

                                /* Flip LEDs. */
                                led = ~led;

                                /* Wait a small amount of time so that the LED blinking is
visible. */
                                for (Delay = 0; Delay < LED_DELAY; Delay++);
                        }

                return XST_SUCCESS; /* Should be unreachable */
}
```

```
/* Main function. */
int main(void){

	int Status;

	/* Execute the LED output. */
	Status = LEDOutputExample();
	if (Status != XST_SUCCESS) {
		xil_printf("GPIO output to the LEDs failed!\r\n");
	}

	return 0;
}
```

- Test
    - Quite simply, I ran the program in the very last step and observed the flashing LED lights on the Zybo board. Clearly showing that the program was set up and worked correctly.
- Discussion
    - I had no prior experience working with Vivado or SDK and this lad gave me a good opportunity to learn how to get started with these programs. I am now sure of starting Vivado, creating a program, building a block diagram on Vivado as well as creating new projects on SDK and importing files and libraries into it.

Attached in Github are pictures of my block flashing back and forth between states.