北京理工大學
Beijing Institute of Technology

信息与电子学院
SCHOOL OF INFORMATION AND ELECTRONICS

# FOUNDATION OF FPGA AND SoPC DESIGN (0501009)

## FINAL REPORT

# TOPIC: DESIGN AND IMPLEMENTATION OF A SIMPLE ZYNQ BASED TRAFFIC LIGHT CONTROLLER


| NAME | STUDENT ID |
| --- | --- |
| GBENGA KUSADE | XXXXX |

**TEACHER: PROF. ZHENYU LIU**


**2020/06/12**

**ABSTRACT**

Traffic jam is a constant predicament owing from the ever increasing rate of automobile on our roads. In order to ensure smooth and efficient coordination of the automobile, there is need for a traffic light controller system. This project demonstrates the working principle of a traffic light system which is design based on Xilinx Zynq FPGA The methods used in this project are hardware development design, software development design, and integration/implementation on the hardware. In this project, XILINX design suite software was chosen to devise a schematic using block design feature, and create an (Hardware Description Language) wrapper, SDK to write the logic in c programming language and implements the circuit on Programmable Logic Device [PLD]. The system has been successfully tested and implemented in hardware using Zynq 7000 xc7z010clg400-1 FPGA.

## 1. INTRODUCTION

### Background

The need for traffic light control is paramount in our today's environment. Traffic jam is a critical predicament which has resulted into many setbacks all across the globe. Due to these problem, time, money and energy has been exhausted due to continual and increasing use of automobiles which has directly impacted our way of life from reduction in the ease of commuting, to productivity of commuters (workers, sellers, etc.), to price inflation and spoilage of foods. In order to solve this problem [1], there is need for smooth and efficient coordination of automobile on our roads, thereby the need for a traffic light controller.

In this project, a simple traffic light controller has been implemented using Zynq-7000 xc7z010clg400-1 field programmable gate array (FPGA) to demonstrate the working principle of a traffic light controller.

### Project Description

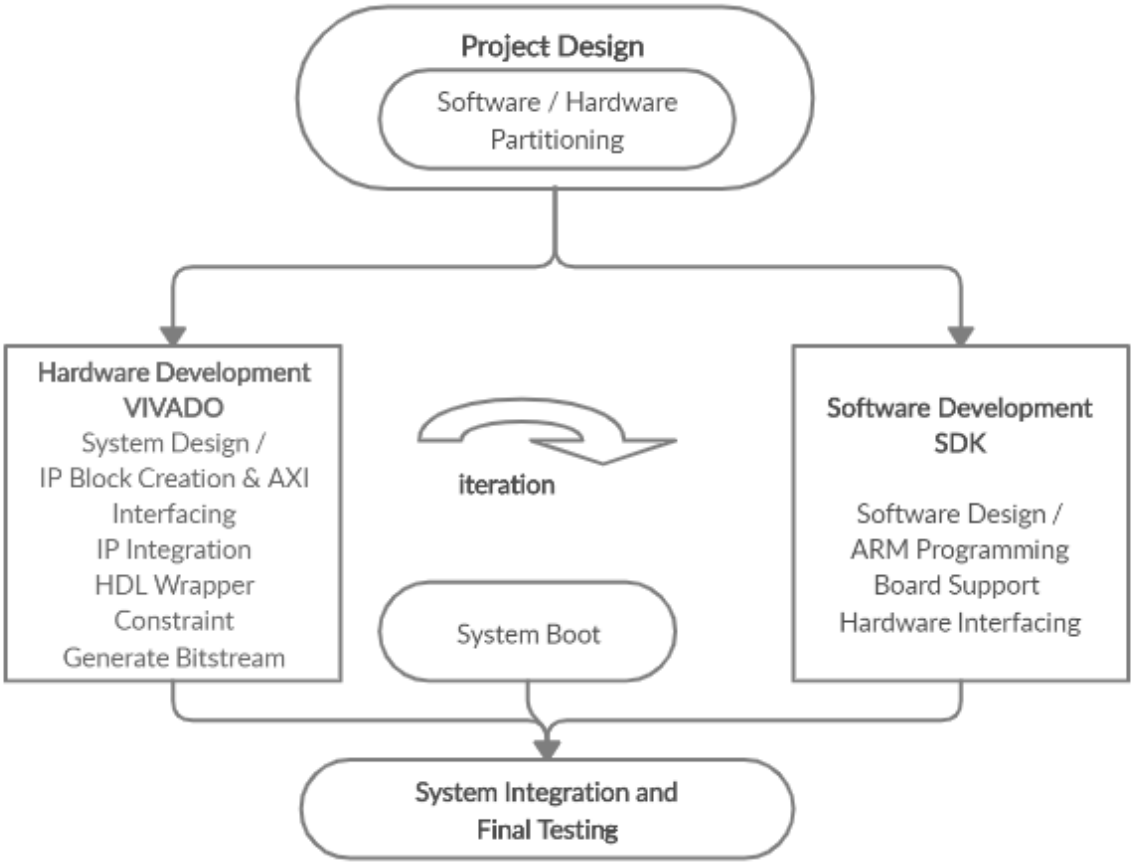
The objective of this project is to design a traffic control system with appropriate delay which can be implemented to show the working principle of a traffic light controllers. The simple traffic light controller design is based on Xilinx ZYNQ (Embedded ARM processor) and it's simply a switching implementation of 3 lights with a countdown clock. The state depicts the change of state of the traffic signal. The signal light values which are available as registers are triggered based on the state, whereas the state value is changed in accordance with a countdown clock with system clock frequency of 50MHz.

The traffic light sequence works on the specific switching of Red, Green and Yellow lights with stipulated time form. The normal function of traffic lights requires sophisticated control and coordination to ensure that traffic moves as smoothly and safely as possible [1]. However, this traffic light sequence is generated using a specific switching mechanism which will help to control the traffic light system in a specified sequence. The implementation involves 60 second for both red and green light and 5 second for yellow light.

The methods used in this project are systematic hardware development design, software development design, and integration/implement on the hardware. In this project, XILINX design suite software was chosen to devise a schematic using block design feature, and create an (Hardware Description Language) wrapper, SDK to write the logic in c programming language and implements the circuit on Programmable Logic Device [PLD]. The system has been successfully tested and implemented in hardware using Zynq 7000 xc7z010clg400-1 FPGA.

## 2. TASK ANALYSIS

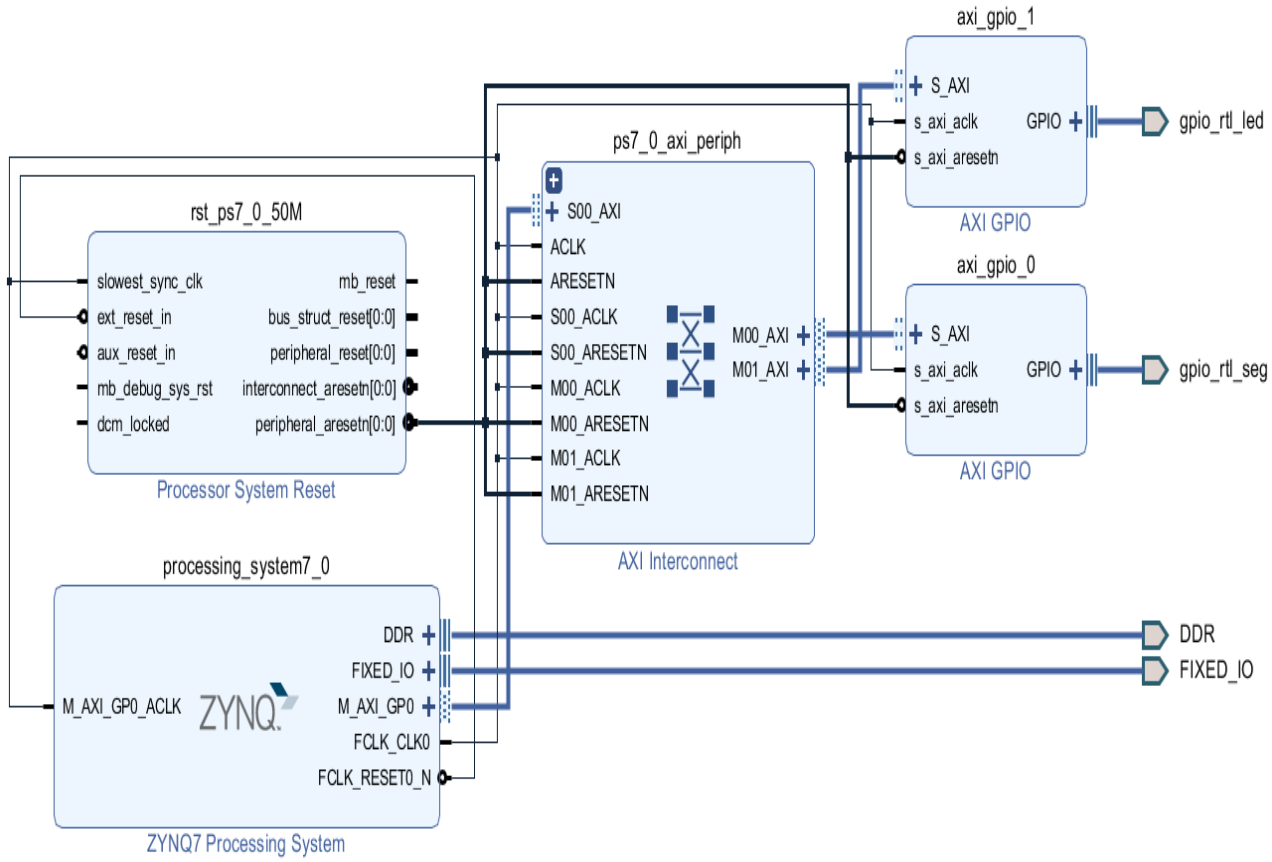
A simplified version of project design flow/ analysis is given in the following block diagram




Project design flow

The project task is partitioned into: Hardware and Software development. which are both integrated into the system hardware for final testing.

**Hardware Development:** Here, the schematic based entry was adopted using VIVADO block design feature to give more visibility into the hardware, the IP block was created with an AXI interface which interface the ZYNQ processor with two AXI GPIO where the LED and Seven Segment Display were both connected. HDL wrapper was created and a constraints added to automatically generate the HDL code. A bitstream (programming file) file was generated after validation of design schematic for the hardware after implementation. The figure showing the block design is shown below.




Schematic/Block design view

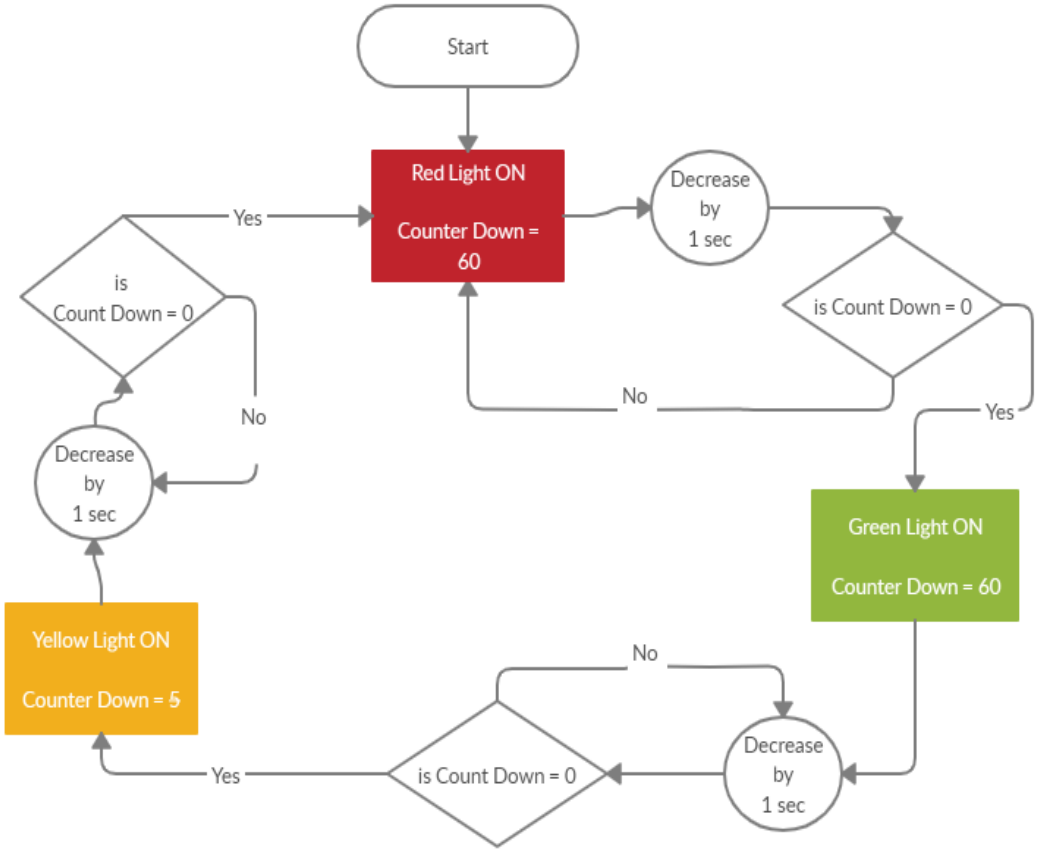
The generated bitstream which is used to interface with the system hardware is exported to the SDK folder for the software design.

**Software Development:** The logic (Software Design) behind the traffic light controller was done using the Software Development Kit (SDK) which gives adding features for debugging, hardware support and interfacing. It is also used to run the code on the ARM core.

**System Integration**: in order to integrate ARM and FPGA, the IP Block Design on Xilinx Vivado using HDL and ARM Programming on Xilinx SDK using C programming language were used on the Zynq 7000 xc7z010clg400-1 FPGA for occasional and final testing on the board.

**Design of Traffic Light Controller**

The logic behind the traffic light controller is that three LEDs present red, yellow, and green traffic light. 60 second for both red and green light and 5 second for yellow light. A count-down clock, two-bit 7-seg led, indicate the remaining time in each time period. The flow chart for the logic is shown below




Traffic light design flow chart

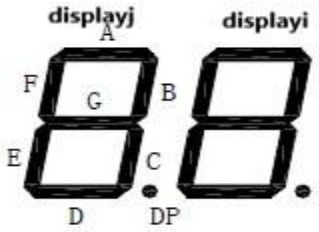
The red light indicates Stop, green light indicates to allow the traffic and yellow light indicates the caution that the traffic is going to be stopped in few seconds. The countdown clock, counts the remaining time in each period when the LED are ON, i.e. the delay for each LED.

**LEDs**

The light emitting Diodes are a type of a diode, which converts electrical energy into light [2]. Three LEDs on the board are used in this project to represents RED, YELLOW and GREEN in that order and connected to PIN R18, P18 and N16 respectively. The HEX value of the position of each LED is sent to the LED GPIO to turn it ON.

**SEVEN SEGMENT DISPLAY (SSD)**

The seven segment displays are used to display the countdown clock for each LED when ON. Here, two SSD display (displayi and displayj) was adopted with a common anode configuration to display the digits (ones and tens) of the countdown clock. The two digits are configured so that displayi displays the ones i.e. count%10, and displayj displays the tens i.e. count/10 respectively.




Seven segment display

To ensure the displays shows the correct value at every time, a select value was used to select the display to turn ON, the value to display on it. The configuration on the Segment GPIO was done as follows: S3, S2, S1, gfedcba

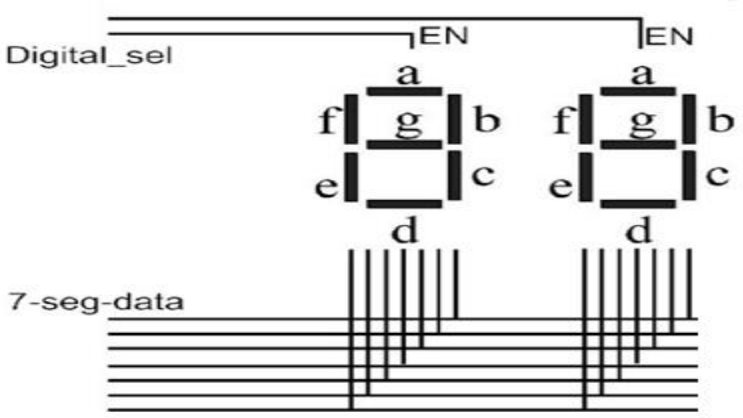
The figures below show the values and corresponding binary digits to be displayed on any of the two displays when selected.

Display values and corresponding binary values

| Number | SSD(gfedcba) |
|---|---|
| 0 | 1000000 |
| 1 | 1111001 |
| 2 | 0100100 |
| 3 | 0110000 |
| 4 | 0011001 |
| 5 | 0010010 |
| 6 | 0000010 |
| 7 | 1111000 |
| 8 | 0000000 |
| 9 | 0010000 |

Displayj is turned Low (0) while displayi is HIGH (1) in order to show the value on displayj. Similarly, displayi is turn low (0), and displayj is turn HIGH (1) to show the value for displayi. These two values are combined with count value to be display and the then written to the Segment GPIO to be displayed on the corresponding display ssd. For example, to write 9 on displayj, 1010010000 (S3, S2, S1, gfedcba) is placed on a register and then written to the GPIO. S3 at every point is high to keep it in OFF state.

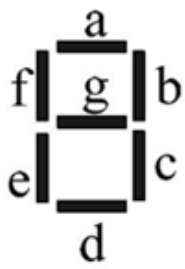
**Common Anode Configuration**




Common anode configuration

the figure above shows the common anode configuration for the two SSD, the LEDs for each display (abcdefg) are both connected inseries to a common port on the board, the below figure is used for the PORT configuration on the board.

| device name | Port | device name | Port |
|---|---|---|---|
| Switch0 | T17 | 7-Seg-a | W15 |
| Switch1 | N17 | 7-Seg-b | U15 |
| Switch2 | N15 | 7-Seg-c | U19 |
| Switch3 | L19 | 7-Seg-d | Y14 |
| Switch4 | M18 | 7-Seg-e | Y17 |
| Switch5 | L14 | 7-Seg-f | W20 |
| Switch6 | J20 | 7-Seg-g | Y19 |
| Switch7 | K19 | 7-Seg-unit | V20 |
| 7-Seg-ten | Y18 | 7-Seg-Hundred | T16 |



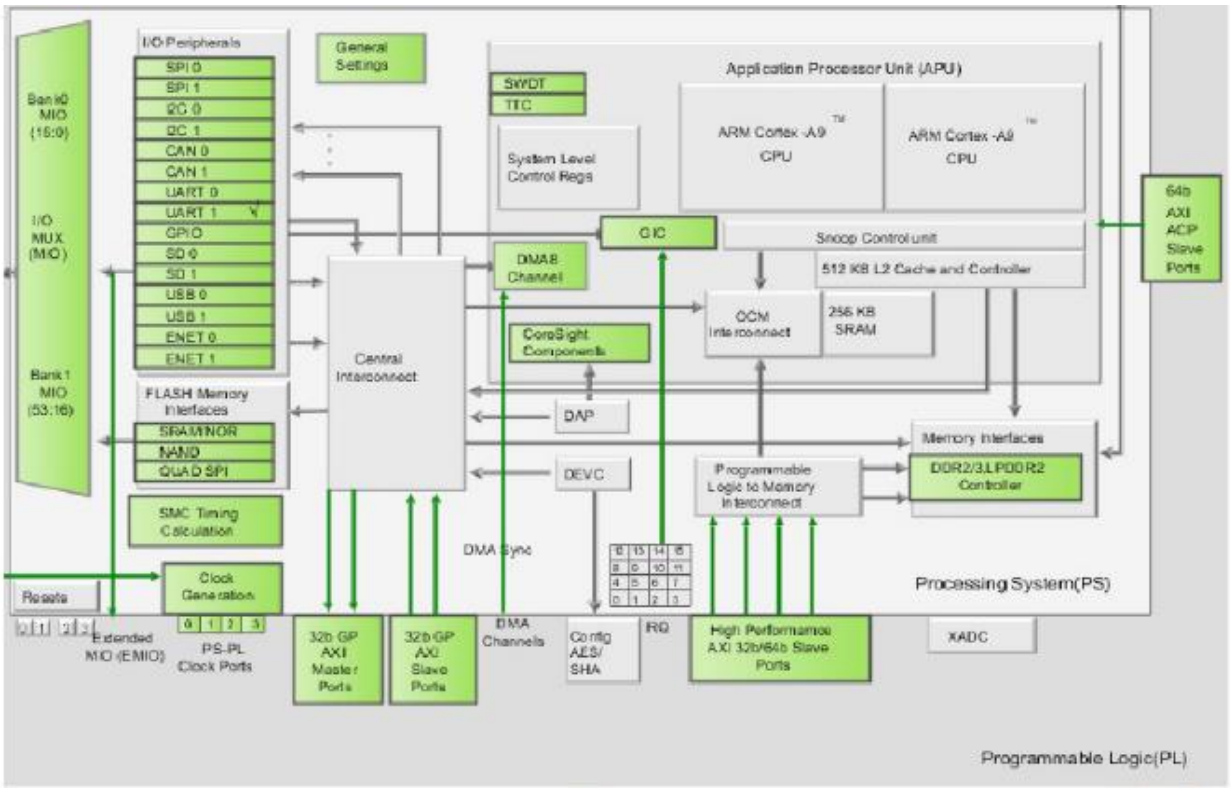

Port configuration for display

The digital_sel is used to select which SSD to display a count value.

## 3. ARCHITECTURE/MODULE DESCRIPTION

**Zynq 7000 xc7z010clg400-1 field programmable gate array (FPGA)**

This evaluation board is a series of the Zynq 7000 series. It integrates a feature-rich dual-core or single-core ARM Cortex-A9 based **processing system** (PS) and 28 nm Xilinx **programmable logic** (PL) in a single device. The ARM Cortex-A9 CPUs are the heart of the PS



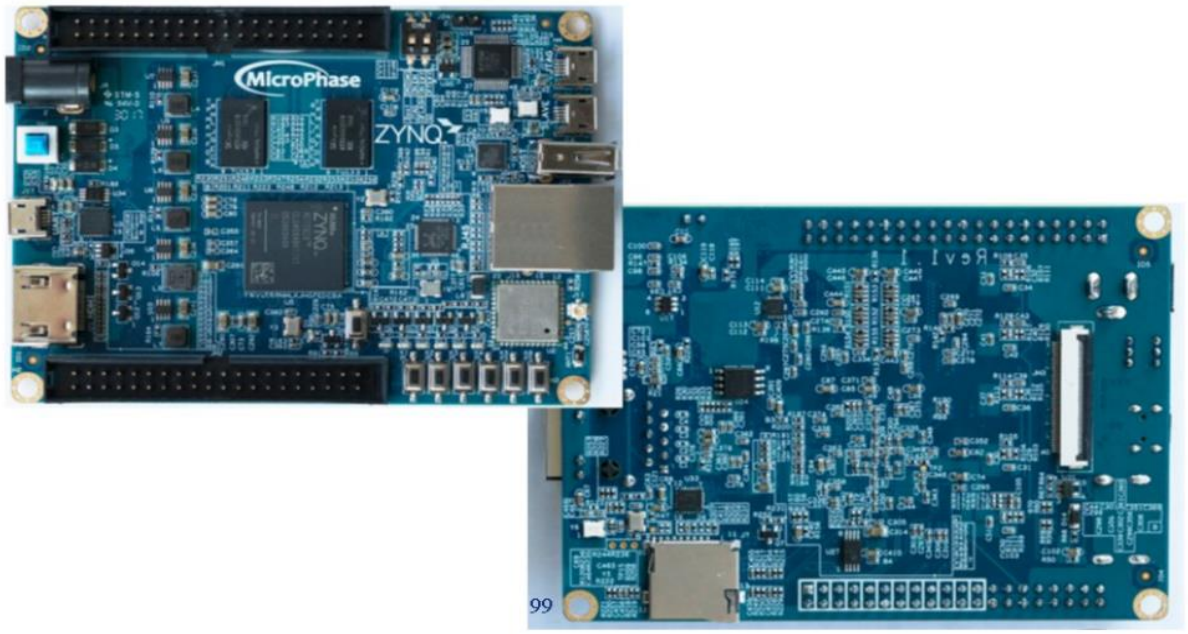

ZYNQ Architecture

**Processing system**

- ARM Cortex-A9 Based Application Processor Unit (APU)
- On-Chip Memory
- External Memory Interfaces
- 8-Channel DMA Controller
- I/O Peripherals and Interfaces

**Programmable Logic (PL)**

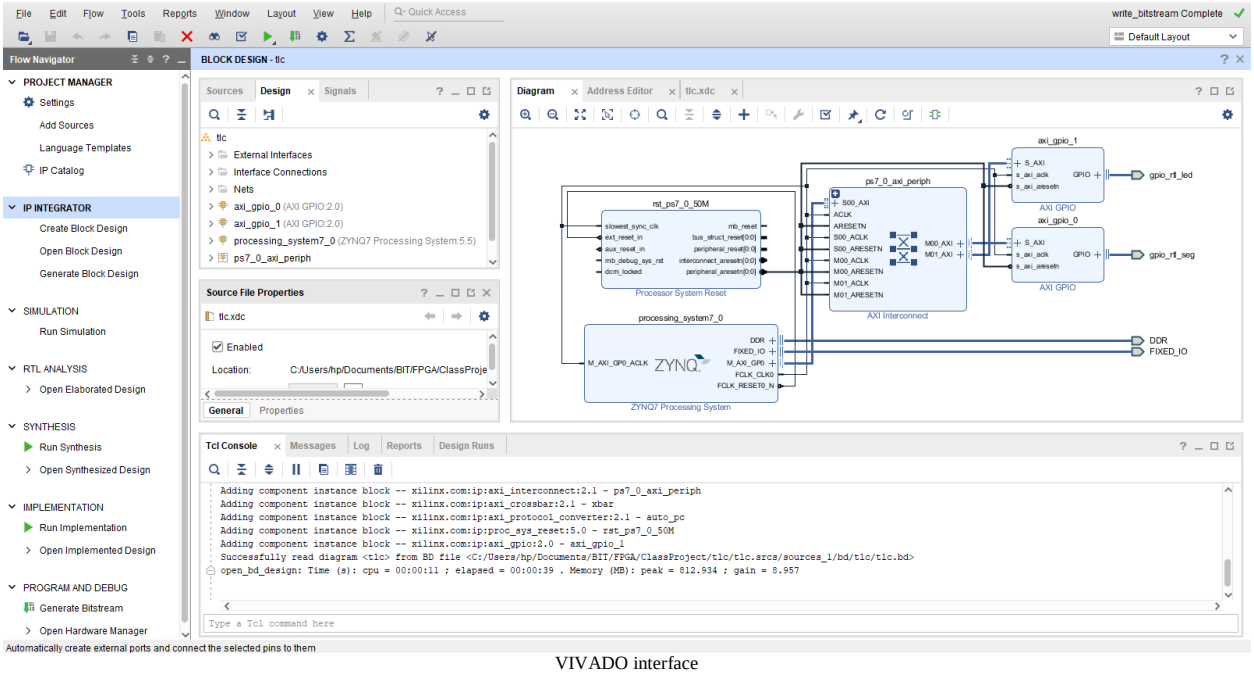
- Configurable Logic Blocks (CLB)
- 36KB Block RAM
- DSP Blocks
- Programmable I/O Blocks
- PCI Express Block
- Two 12-Bit Analog-to-Digital Converters




Zynq 7000 xc7z010clg400-1 FPGA Board

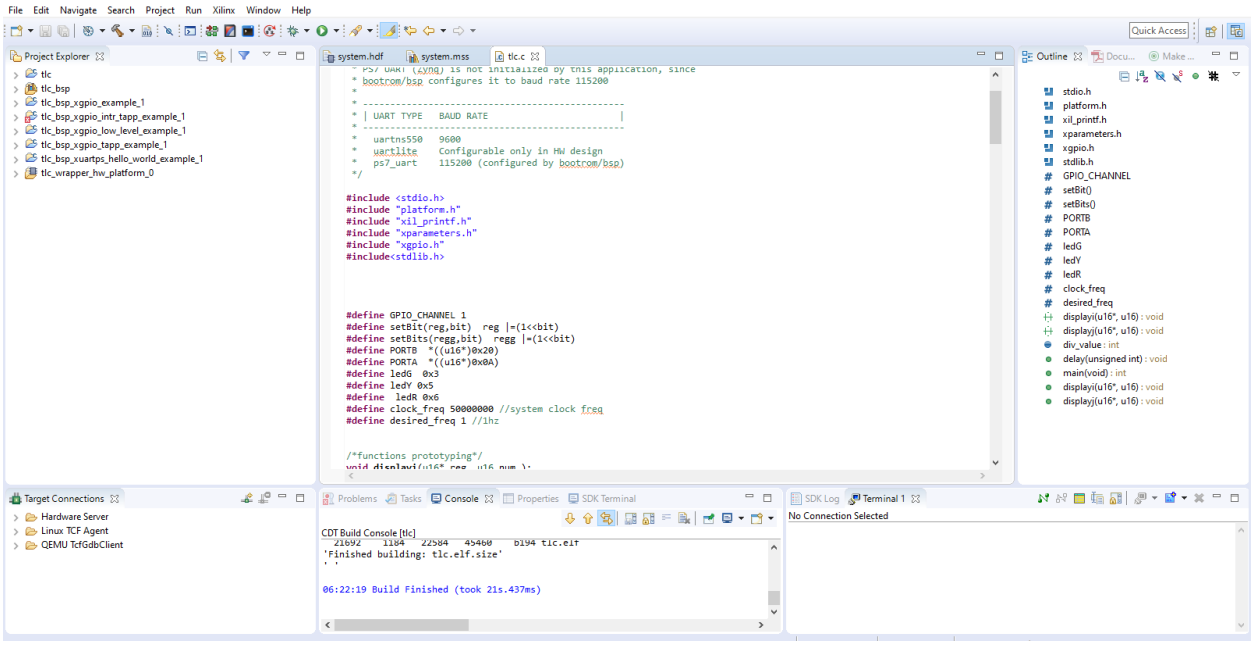
**The Xilinx design tools**
The design tools are used to cater for both hardware and software part of the project.

**Vivado** – This is a top level design environment for the hardware design. It was used to create the contents of the Programmable Logic, to create the embedded processor section of the design, and to configure the settings for the Zynq "Processing System" section of the design.




VIVADO interface

**SDK** – The Software Development Kit was used for the software design, where the C code for the traffic light controller was developed, also used to test the code using the debugger.

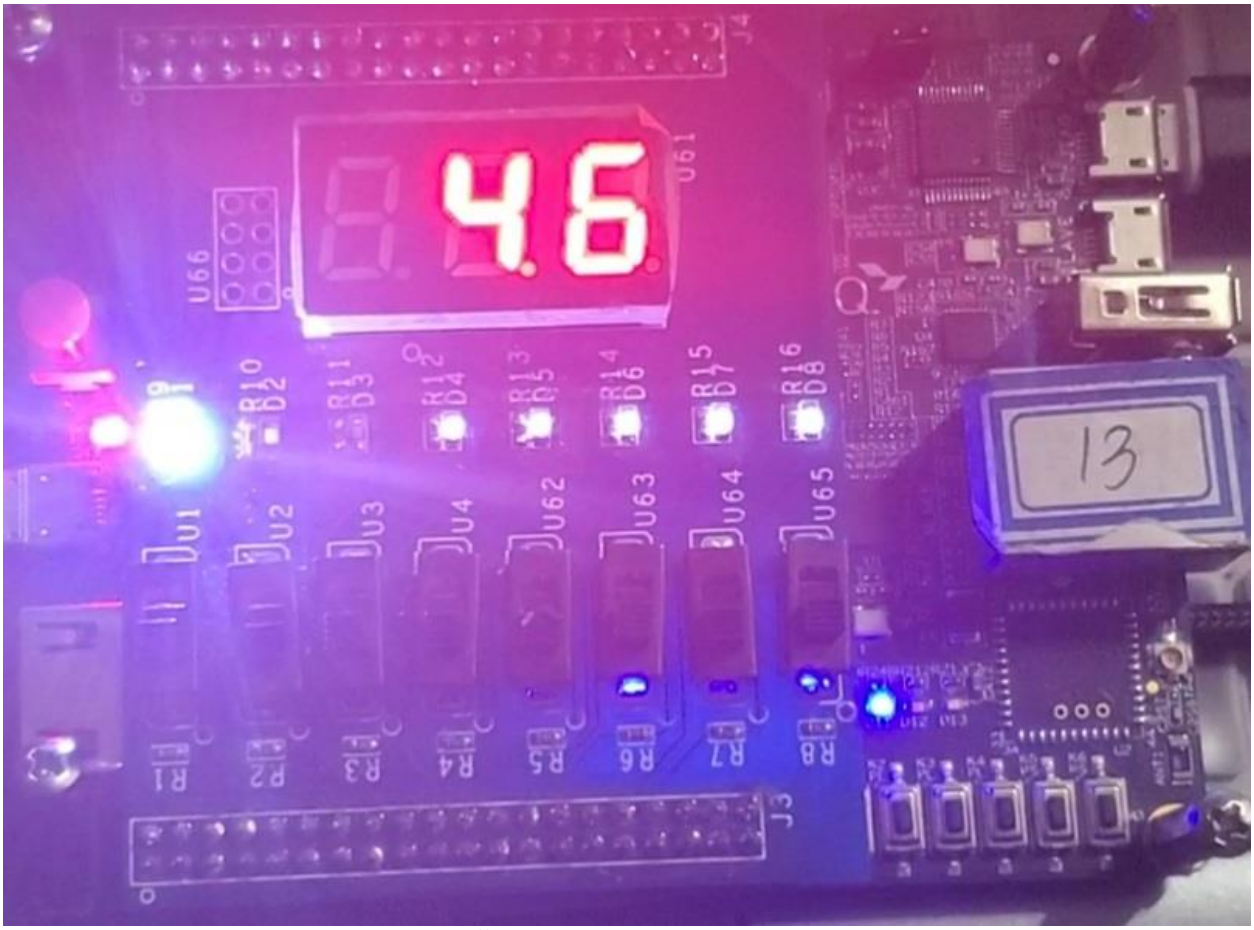



SDK interface

## 4. EXPERIMENTAL RESULTS

**Hardware Implementation**

The Traffic Light Controller was designed using the integration of ARM and FPGA. The output of the Traffic light controller is verified with Zynq 7000 xc7z010clg400-1 (FPGA). Here, three LEDs & two 7-segment display are used to represent the outputs. The left most three LEDS corresponds to Red, Yellow and Green and the right two 7- segment display signifies the countdown clock displays.




Traffic light controller output on the board

At start, the RED (stop) light turns ON and the countdown clock displays counts for 60 seconds, after it turns off and switch to the GREEN (Go) light, with the display counts down for 60 seconds, and switches to the YELLOW (warning) lights which count down for 5 seconds. The process iterates as long as the devices remains ON.

## 5. DISCUSSION AND CONCLUSION

The demonstration of traffic light controller system has been achieved. The LED lights turns ON in sequence for a specified time which is shown on the seven segment display as a countdown clock. A delay mechanism was adopted for both the LED and the clock. The goal is to generate a 1Hz pulse from the 50MHz clock signal. The equation (clock_freq/(2* desired_freq) – 1) was adopted to get the division value. However, the precision of the countdown clock differs from the real clocking system initially by large margin. So, the delay function was adjusted to a milliseconds delay which helps the seven segment display visually stable and the countdown clock precision more accurate. The countdown for a minute (60 seconds) counts up to 62538.08mS, 62538.10mS and 62538.13mS respectively for three counts according to the xtime_l header function. From the data, the precision value of the countdown clock varies between 0.02 and 0.05mS. This project shows a simple demonstration of a traffic control light system. It however can serve as basis for a more advanced intelligent traffic light control system.

## REFERENCES


1. Bhavana, D., Tej, D. R., Jain, P., Mounika, G., & Bhavana, R. M. (2017). traffic light controller using VHDL. *International Journal of Modern Trends in Engineering & Research*, *4*(4), 57–62. https://doi.org/10.21884/ijmter.2017.4118.c529
2. Shivani, D., & Shashidhar, S. (n.d.). *PROJECT REPORT ON Under the guidance of*. (012560340).
3. Ming-Chang YANG (2020), Lecture 07: Rapid Prototyping (I) – Integration of ARM and FPGA, retrieved from https://bit.ly/3f1TiVu