ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

A Digital Systems Design of Simple Traffic Light Controller

V.Suresh
Assistant Professor,
Department of ECE,
A.I.E.T, Thagarapuvalasa,
Visakhapatnam.

P.Sateesh Kumar
Associate Professor,
Department of ECE,
A.I.E.T, Thagarapuvalasa,
Visakhapatnam.

M.Kesab Chandrasen
Assistant Professor,
Department of ECE,
A.I.E.T, Thagarapuvalasa,
Visakhapatnam.

D.Lakshmi Narayana
Associate Professor,
Department of ECE,
A.I.E.T, Narisipatnam,
Visakhapatnam.

chandrasenmk@gmail.com & suresh.eceavanthi@gmail.com

Abstract:

In the past few years, laboratory exercises for our Digital Systems course were typically self contained. The lack of integration prevented students from fully realizing a complete digital system during the course of the class. The simple traffic light controller design was introduced to alleviate this shortcoming and to ensure students gain experience in solving implementation and interfacing problems of a modern digital system. Students implement a fully functional traffic signal controller for a four-way intersection. The intersection is complete with sensors to detect the presence of vehicles waiting at or approaching the intersection. This paper incorporates many concepts and components that are discussed in detail throughout the course. These include FPGAs, VHDL for modeling and synthesis, finite state machines, embedded microprocessors, memory interfaces, serial communication, and signal synchronization. A bottom-up with a partially specified design methodology is used to encourage students to use their breadth of knowledge and creativity. By the end of the project, students will have gained a better understanding of digital system design methodologies through hands-on experience.

Index Terms:

Education, digital systems, traffic light controller, finite state machine, VHDL, FPGA.

I. INTRODUCTION:

THIS paper describes a simple traffic light controller design project. This project was developed because there was a need for laboratory exercises that incorporated VHDL modeling, simulation, microprocessors, memory interfaces, serial communications and a variety of related topics into a complete digital system.The design project requires students to develop a state machine based controller for traffic signals at a four-way inter-section.

The intersection has two travel lanes in each direction; north, south, east and west. In addition, each direction has a dedicated left turn lane. Each lane contains a sensor to indicate if a car is waiting at a red light. Travel lanes also contain additional sensors to indicate if cars are approaching the intersection. This project stresses the difference of writing VHDL for modeling and synthesis and that VHDL should not be thought of as a programming language. It teaches proper design of combinational and sequential circuits. Often students unin-tentionally infer latches when they are trying to generate a purely combinational circuit. It requires proper definition of pin constraints for interfacing peripherals external to the FPGA. These are some typical student pitfalls that should be avoided when learning to design a digital system [1].In past sections of the course, a series of loosely related lab-oratory exercises were used.

The system components studied did not coalesce into a complete digital design. Students lacked hands-on experience of the interfacing between different components. Another drawback is that the study of components often occurred only in simulation. This led to confusion on how to write synthesizable VHDL.The simple traffic light controller design project is structured in a way that allows the student to study a particular digital system component and then integrate it into the design. A Xilinx based FPGA development board is used for all hardware components of the controller. This allows the inclusion of state machines, microprocessors, memories, UARTs, etc. under a single development environment. For simulation, students use ModelSim PE Student Edition. The Synthesis, and place and route processes are performed using Xilinx's ISE Web PACK. Both of these tools are available as a free download to students.The following sections describe the traffic signal phase requirements, the system architecture, and hardware test bench. Then, the progression of the signal controller design project is presented as a series of six stages, each building upon the previous.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

II.TRAFFIC SIGNAL PHASE SEQUENCES:

The traffic light controller must handle a four-phase signal intersection. Students have the option to implement two 4-phase sequence options. In the first option, through traffic for the east and west directions proceed at the same time; then north and south through traffic, followed by the north and south turning lanes. Finally, the east and west turning lanes can proceed. In the second option, through traffic and the turning lane for a single travel direction access the intersection at the same time. Students may implement an alternate phase sequence, but each phase must be as described. See figure 1 for phase details.

A six-phase signal sequence can also be implemented. The six-phase intersection is commonly referred to as a split green intersection. It is a combination of leading and lagging left turns [2]. Left turns are implemented at the beginning and the end of each north-south and east-west sequence. For example, in a North-South sequence, the North bound travel and turn lanes would be green for 15 seconds. Then, the turn lane will sequence to red and the South bound travel lanes will turn.

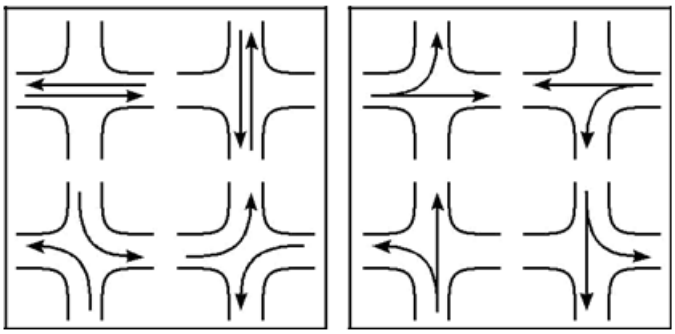

Figure 1. 4-Phase options: (left) option I, (right) option II.

green. After 30 seconds, the North-bound travel lanes will sequence to red and the South-bound left turn will go green for 15 seconds. Finally, all South-bound lanes sequence to red. The East-West sequence can then begin.

III. SYSTEM ARCHITECTURE:

The complete architecture, shown in figure 2, consists of the traffic light controller FSM, an embedded processor, embedded program memory, multiplexers for processor input and output, a bidirectional UART, digital switch debouncers, and capture synchronizers.

Lane sensor inputs are directly monitored by the FSM. Approach signals are first debounced and synchronized. The light controller state information is passed to the processor for monitoring and reporting purposes. The processor reports the FSM state in a human readable format to a PC terminal through the embedded UART. Commands from the terminal can also be received and processed to generate control signals to the FSM. For example, the ASCII command "AR" may instruct the processor to set the traffic light controller to an, all-red, failsafe mode. Students can also use the state information to implement additional features.

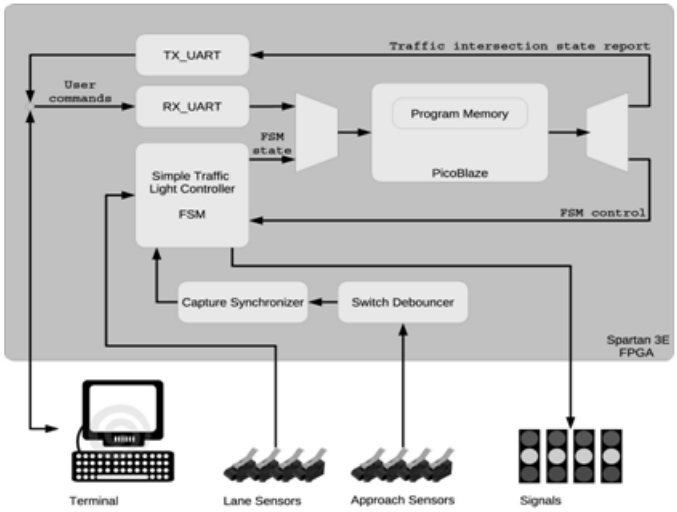

Figure 2. System Architecture

Implementation of the traffic light controller is targeted to the Digilent Nexys 2 FPGA development board [3]. This board contains a Xilinx Spartan 3E FPGA, flash memory, and various user IO devices and interfaces. Using an FPGA allows students to add hardware without having to worry about the physical connections between them. Instead, they can concentrate on the operation and interfaces of the various devices used. At the beginning of the project, a Moore finite state machine is the only device on the FPGA. Later, the signal controller will monitor sensors to detect whether a car is present or approaching the intersection. Sampling the lane sensors is fairly trivial and does not require the use of any additional hardware. Things become a slightly more complicated when approach sensors are introduced. These asynchronous signals must be debounced and captured. To implement this feature, a digital switch debouncer is implemented using a sample and count technique [4]. A simple capture synchronizer, shown in figure 3, is then used to synchronize the asynchronous approach sensor signals to the FSM clock.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

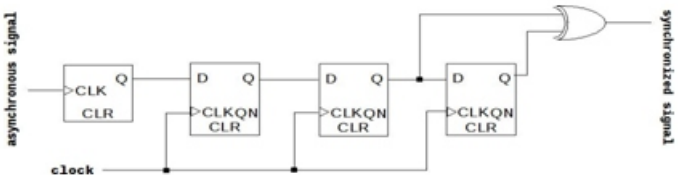

Figure 3. Capture Synchronizer.

To facilitate communication and control of the traffic light FSM from a computer terminal, a bidirectional UART [5] and an embedded processor with required memories is used. Xilinx's PicoBlaze microprocessor [6] was chosen because of its simple IO interface and low FPGA resource consumption. This processor is used to send information about the signal's state in a human readable form to a PC terminal. Commands from the PC terminal can also be received to allow altering the state of the signal controller. Xilinx only provides the KCPSM3 assembler to generate the PicoBlaze's program memory, forcing developers to use assembly language.

Students are already exposed to assembly language programming in the micro-computer course that runs concurrently to the digital design course. To alleviate redundancy between the two classes and provide faster software development, programming of the the PicoBlaze is accomplished using the C language. Unfortunately, an official PicoBlaze C compiler does not exist. Francesco Poderico has written the freely available PicoBlaze C Compiler [7]. This compiler does not seem to be under active development and generates less than optimal assembly, but it supports most of the required language features for this project. In-line assembly can be used if memory constraints or performance become an issue to the student design.

IV. INTERSECTION MODEL:

In addition to the FPGA board, a model of a four-way intersection was constructed. It contains two travel lanes and a left turn lane for each direction. To simulate sensors, momentary switches are used to detect cars waiting at a red light and cars approaching the intersection. Because of IO constraints on the Nexys 2 board, the turn lane sensor is tied to the through traffic lane sensor. This sensor can be untied and connected via the Nexys' FX2 connector if the student design requires independent monitoring of the left turn sensor signal.

Figure 4. Intersection configuration.

SPST-NO switches are embedded into the base of the model intersection at the locations marked in figure 4. These are connected to FPGA IO pins via the Peripheral Module (PMOD) connectors on the Nexys 2 board. Pull-down resistors are enabled for these pins. Traffic can be simulated simply by rolling Matchbox cars over the sensor switches.The signal lights are LEDs mounted on a PCB in the shape of a light post with three signals. The posts were fabricated in house using VCU's PCB milling facilities. This reduced the cost of manufacturing the model test bench and allowed for the complex shape of the PCB. LEDs with wavelengths of 630 nm, 590 nm, and 528 nm for the red, yellow, and green signals where used because they closely match the actual colors used on a real traffic signal light.

Because the FPGA IO pins cannot source sufficient current to light an LED, NPN transistors are used as simple LED drivers. Each signal post is fastened to the model base using an IDC connector. Three pins on this connector are used to control the through traffic signals. Another three are used for the left turn signal. Power and ground are provided with an additional two pins. These pins along with the lane and approach sensor signals are connected to a PMOD connector on the Nexys board. Each of the four available PMOD connectors have enough IO pins to service a single travel direction.The base of the test bench is a 1.5 inch thick Styrofoam sheet covered with poster board. The road markings were created by adhering a large piece of paper with the plot of a four-way intersection. A wood frame surrounds the model to provide structural support and protect the edges. All materials for the base can be inexpensively purchased at any home improvement center.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

V. PROJECT STAGES:

Before beginning the design project, students are familiarized with the development environment through the com-pletion of introductory and tutorial exercises. These include exercises on using ModelSim, Xilinx ISE, and the Nexys 2.

Figure 5. Completed intersection model test bench.

development board. Students, grouped in pairs, implement the traffic signal controller in a series of six stages. Each milestone builds on the previous by adding functionality. Because of this, it is important for students to successfully complete each step on time. A bottom-up with a partially specified design method-ology [8] is used. This encourages students to draw from their own knowledge in order to complete each milestone. This is particularly true for the last development stage. The six stages or milestones are outlined in the following subsections.

A. Stage 1:

In the first stage, students implement the basic light controller finite state machine. At this stage of development, the controller cycles through the proper sequence to regulate travel in each direction; north-south and east-west. The green light period is set to 30 seconds and yellow light period to 7 seconds. The main purpose of this stage is to make sure that students understand the basic operation of the FSM and are capable of simulating the design. Prior to beginning this stage students are introduced to the Memory/Next-state/Output Process model for state machines. As an example, the state machine for the stage 1 basic controller is presented in its entirety during lecture. This ensures that all students have a solid foundation on which to begin the project. The complete code listing is provided, so all the student has to do is simulate and use the hardware test bench to verify proper operation. This is a crucial first step and it is important that every student is successful in its timely completion.

B. Stage 2:

In stage 2, left turn lane signals are added. Students have the option to implement either of the two 4-phase sequences or the split green 6-phase sequence. In all implementations, the green-light period is 30 seconds and the yellow-light period is 7 seconds for the travel lanes. Turn lanes have a green-light period of 15 seconds and the yellow-light period of 7 seconds.

C. Stage 3:

In this stage, lane sensors are used for the first time. The simple controller cycles through the proper sequence as above. However, if there is a car at the intersection in the direction that is red, the green-light period is shortened by 10 seconds.This should occur whenever a car appears at the intersection in the lane with a red light at any time before the light changes, i.e., if a car appears at the intersection at the red light any time after 20 seconds, the sequence to change to a green light for the stopped car should begin immediately.

D. Stage 4:

The controller cycles through the controlled sequence of stage 3. However, if a car is detected passing through the approach sensors when the light in that direction is green, the sequence is delayed (extended) for 10 seconds to allow the car to pass through the intersection. This delay only occurs if no cars are waiting at the intersection at a red light. The sensor switch must be debounced and synchronized to avoid undesirable effects.

E. Stage 5:

At this stage controller cycles through the controlled se-quence of stage 4. In addition, the PicoBlaze processor moni-tors the state of the controller and provides an ASCII, human readable output of the current controller state via the RS-232 port on Nexys 2 board. Also, the PicoBlaze can accept an ASCII command via the RS-232 port to reset the controller back to the safe, all-red state. Prior to beginning this stage, students complete a tutorial laboratory to introduce to the Xilinx PicoBlaze processor IO and the C compiler. Students are also required complete a tutorial laboratory on how to use the embedded UART to communicate with a PC terminal.

F. Stage 6:

In this stage students can add some unique, innovative functionality of their own design to the controller.

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Additions made at this stage are judged by how original and complex the added functionality is and how well it performs.

VI. RESULTS:

As expected, students that performed well in homework assignments completed all milestones of the project on time. Students that fell behind were given the opportunity to seek additional help. This ensured that all students were able to complete all the design stages. One major cause of delayed milestone completion was the lack of simulation prior to synthesis and testing on the hardware test bench. Students had the initial expectation that just because their VHDL description synthesized without errors, it would function flawlessly. They quickly found the value of simulating their signal controller. In fact, not providing a board to every team was done intentionally, so students would learn to rely on the simulation environment rather then testing every modification on the hardware test bench.For milestone six, students added many interesting features to their designs. Several teams added 'red-light' cameras to their controller design. One team used a buzzer to indicate that a car ran a red light. Another flashed a light to simulate the flash of a camera. The best implementation of red light camera used a camera to take a picture of the red-light running car.

The camera was controlled using its remote trigger feature. To detect that a car entered the intersection on a red light, all teams monitored the lane sensor for a high to low transition when the signal was red. Teams also added the capability for the signal sequence to be altered by an emergency vehicle to allow safe and quick access to the intersection. Some teams used buttons on the Nexys board to simulate a sensor input. A better implementation used an infrared emitter on the vehicle and a infrared detector on the signal post. Other notable additions include the addition of cross-walk signals, GUI interface to the signal controller, night-time (blinking red/yellow) sequence, and extending the green light time for through lanes that are backed up. One team implemented totally pointless, but fun to watch mode. This mode, termed 'Napster', was intended to clog the intersection with wrecked vehicles by quickly cycling through a four-phase sequence.A major limitation of the hardware test bench was the availability of GPIO pins. All of the PMOD connector IO pins were used for the signal control and sensors leaving no pins available for student designs.

Some were able to avoid this limitation by using the Nexys 2's buttons, switches and LEDs. If a design required IO external to the board, the FX2 connector had to be used to gain access to addition IO pins. Future versions of the test bench interface will use the Hirose FX2 connector for the light control and sensor signals and leave PMOD connectors for student additions.

VII. CONCLUSIONS;

The Simple Traffic Light Controller digital design project turned out to be a great success. A high rate of successful completion was observed. This can be attributed to the im-plementation of the controller as a series of milestones, the use of a realistic hardware test bench and allowing students to innovate the simple controller by adding functionality of their own design. More importantly, the project provided hands-on experience with most of the course's learning objectives. These include the use of VHDL to model, simulate and synthesize digital systems, understanding the function, design and implementation of components of modern digital systems, understanding the fundamental concepts and applications of FPGAs, and to develop expertise in microprocessor interfaces.

REFERENCES:

[1]R. Duckworth, "Embedded system design with fpga using hdl (lessons learned and pitfalls to be avoided)," in Microelectronic Systems Education, 2005. (MSE '05). Proceedings. 2005 IEEE International Conference on, June 2005, pp. 35–36.
[2]R. L. Gordon and W. Tighe, "Traffic control systems handbook," Federal Highway Administration, Washington, DC, Tech. Rep. FHWA-HOP-06-006, Oct. 2005.
[3]Digilent Nexys2 Board Reference Manual, Digilent Inc.
[4]A. Greensted. (2009, Aug.) Switch debouncing. [Online].Available: http://www.labbookpages.co.uk/electronics/debounce.html
[5]K. Chapman, 200 MHz UART with Internal 16-Byte Buffer, Xilinx, Apr. 2008.
[6]PicoBlaze 8-bit Embedded Microcontroller User Guide, Xilinx, Nov. 2009.
[7]F. Poderico, PicoBlaze C Compiler, Jul. 2005.
[8]J. Cerda Boluda, M. A. Martinez Peiro, M. A. Larrea Torres, R. Gadea Girones, and R. J. Colom Palero, "An Active Methodology for Teaching Electronic Systems Design," IEEE Transactions on Education, vol. 49, pp. 355–359, Aug. 2006.