To The Graduate School:

The members of the Committee approve the project of Austin T. Griffith presented on June 14, 2006.

Dr. Steven Barrett, Chairman

Dr. Suresh Muknahallipatna

Dr. William Spears

APPROVED:

Dr. Mark Balas, Head, Department of Electrical and Computer Engineering

Don A. Roth, Dean, The Graduate School

Griffith, Austin T., HDL Verilog Embedded Design Feasibility Study, M.S., Department of Electrical and Computer Engineering, August, 2006.

The current HDL Verilog course, EE4490, is based solely on software simulations and not actual hardware implementations. This is a disadvantage to the educational process because students are limited to software results instead of real world, physical interaction. On the other hand, we currently have a separate microprocessors course that is using robots to autonomously navigate a maze and avoid walls. If possible, we would like to have the HDL students working with the same robots. In doing this, we not only accomplish the goal of physical results, but we also create a great basis for a third course which requires both HDL Verilog and microprocessors concepts. To accomplish this task, I used a Xilinx XCR Plus development board to control the robot. I wrote a real time operating system in HDL Verilog that processes the robot's vision and, in turn, controls its movement. I then wrote a step-by-step guide to walk through the process of programing and controlling each subsystem of the robot. Finally, I compiled a list of implementation suggestions on integrating this educational program into the current curriculum.

HDL VERILOG EMBEDDED DESIGN FEASIBILITY STUDY

by

Austin T. Griffith, B.S. Computer Engineering

A project submitted to the Department of Electrical and Computer Engineering
and The Graduate School of The University of Wyoming
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE
in
ELECTRICAL ENGINEERING

Laramie, Wyoming
August, 2006

To my parents.

I’m finally done spending all your money.

Contents

List of Figures

Acknowledgments

First and foremost, I would like to thank Dr. Steven Barrett and Dr. Mark Balas for their constant battle for funding. Your contributions make a huge impact on the lives of many students funded by the Electrical and Computer Engineering Department. Second I'd like to thank my committee members, Dr. Suresh Muknahallipatna and Dr. William Spears for working with my ever-changing schedule and helping me to finish up in time. I'd also like to thank Xilinx for providing the development boards free of charge and Dr. Cameron Wright for the LaTeX template. Finally, I'd like to thank Dr. Barrett and my parents for all your help keeping me on track, in line, and motivated.

Austin T. Griffith

The University of Wyoming
August 2006

Chapter 1

Introduction

1.1 The Challenge

Our current Verilog HDL class teaches students how to design complex embedded controller systems and real time operating systems. Currently all of their results remain in software simulations. It is difficult to get a range of students from Computer Science, Computer Engineering, and Electrical Engineering actually implementing software designs into hardware environments without allowing them to practice an end-to-end design. To integrate hardware into the curriculum will require some overhead costs. Many programmable applications can range up to one hundred dollars per board or, in this case, per student. Also, once we have a hardware device, there must be work done to determine how to interface it properly with the readily available maze navigating robot platforms already used within our microcontroller-based laboratory program. Therefore, our challenge is to find a low cost hardware solution to fit the needs of the students, get it operating correctly with current hardware applications, and create curriculum to support its inclusion in the Verilog HDL class.

1.2 Overview

This report outlines all the material I have prepared to launch a laboratory section for the EE 4490 Hardware Descriptive Language course. It includes a full set of guides to take basic hardware and Verilog HDL knowledge and control a robot that can navigate a maze in real time. From there, the report follows up with suggestions as to where this program should be taken in the future.

Chapter 2

Background

2.1 Motivation

As previously mentioned, Verilog HDL students have the knowledge and ability to design complex devices, but no hardware solutions to actually apply this knowledge. On top of that, the Department of Electrical and Computer Engineering has a large amount of educational robot platforms available that could be interfaced with and then placed in the maze that has already been built. The robots navigating autonomously in a maze has proven to be a powerful educational tool. Therefore, a logical answer would be to find a hardware board that HDL students could program and then interface with the available robots.

2.2 EE4490 Hardware Descriptive Language (HDL) Course

The EE4490 HDL course covers a semester's worth of basic HDL curriculum. Verilog HDL is a hardware design language that is meant to interface with and emulate real world digital hardware. It is also a very popular course attended by Computer Science, Computer Engineering, and Electrical Engineering students. Because HDL is meant to be applied to hardware applications, the current curriculum could be enhanced because it only offers software simulations of what real hardware would do.

2.3 Xilinx

Xilinx, the Programmable Logic Company, has offered us samples of a variety of their Complex Programmable Logic Devices (CPLD) and Field Programmable Gate Array (FPGA) boards that can be programmed in Verilog HDL. Many of these development boards could be used to interface with the robot, but to determine which would be best, it comes down to the per unit cost and ease of student use.

2.4 Concepts

To find out which board would be the best to use, we should first explore the concepts that will be needed to operate the robot platforms. As noted above, these robot platforms were originally developed in-house using readily available, off-the-shelf components. The robots are used extensively in the EE4590/5590 Real Time Embedded Systems course to teach complex concepts.

2.4.1 Robot Platform

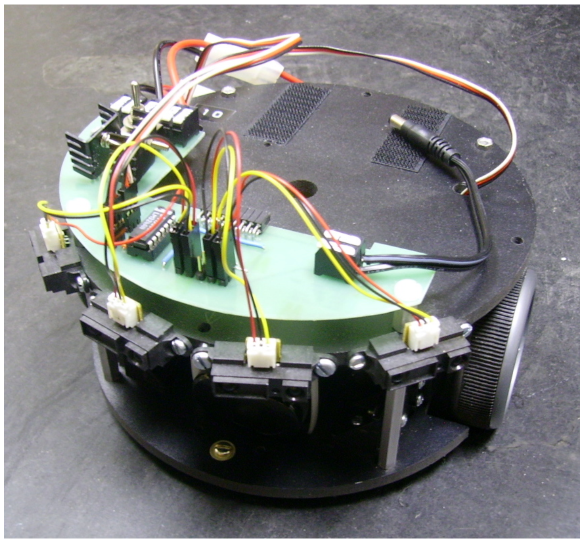

Figure 2.1: Bare Robot Platform

The bare robot platform's heart is the printed circuit board (PCB) designed and fabricated in-house in the Electrical and Computer Engineering Department. The PCB interfaces all of the robot's subsystems including; vision, motors, power, and the micro-controller.

2.4.2 Power

The power system has the ability to switch between a remote wall jack power supply and on-board rechargeable batteries. From there, it is routed through a voltage regulator and then sent out to all of the necessary subsystems including power for the micro-controller.

2.4.3 Motor Control

The robot's servo motors are controlled by pulse width modulation. PWM means sending a signal at a certain frequency that consists of a repeating square wave. The on time of this square wave is then changed to convey specific information. The stepper motors on the robot require a signal with a frequency of approximately 50 Hz, meaning a wave will repeat itself every 20 ms. During this 20 ms, the signal will be a logical high for a certain period of time and then it will be logical low for the remaining time. For instance, to make the motor spin forward, we must send it a 1 ms wide pulse at a logical high and then 19 ms of logical low. To make the motor spin in the opposite direction, we send a 2 ms pulse of logical high and then an 18 ms pulse of logical low. This signal is then repeated over and over to create a 50 Hz continuous signal. Duty Cycle is the waveform specification that relates the on time of the signal to its total period. It is usually expressed as a percentage from 0% to 100%.

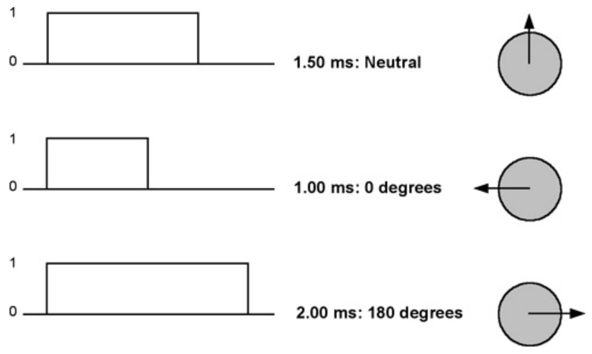

Figure 2.2: Duration of Pulse That Dictates Rotation Direction. [1]

2.4.4 Vision

The robot's vision comes from the 4 infrared range detection modules mounted along the front of the robot. For this project Sharp GP2D12 general purpose type distance measuring sensors were used. These modules emit IR light and, using triangulation, based on the angle of what is reflected back, one can judge an object's range. The sensor outputs a voltage based on the range of the object determined by the magnitude of reflected light. For example, a wall at a distance of 10 cm will result in an outputted voltage of 2.2V but a wall at a distance of 30 cm will cause the module to output a voltage of 1V. Therefore, we have an analog range of voltages coming from the sensors that needs to be converted to a digital quantity to be understood by the Xilinx chip. In many applications an analog to digital conversion (ADC) would be implemented. This would take a single analog output and then output a multiple bit binary digital output. It should be noted that the output from the sensor module is not linearly related to range. A characteristic sensor profile is provided in Figure 2.3.

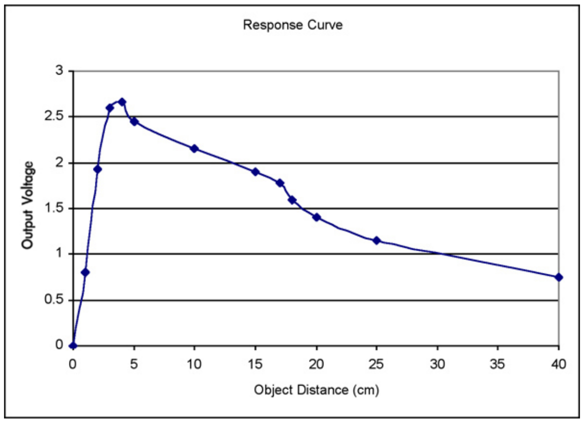

Figure 2.3: Voltage versus Distance to Reflected Object Curve. [1]

2.4.5 Micro-Controller Interface

All of the above systems are then routed to a header on the PCB, which can be used to interface the robot with a micro-controller. Below is a picture of the Xilinx XCR Plus mounted on the robot and wired to the header. As you see, this evaluation board can be easily mounted to the existing robot without modification.

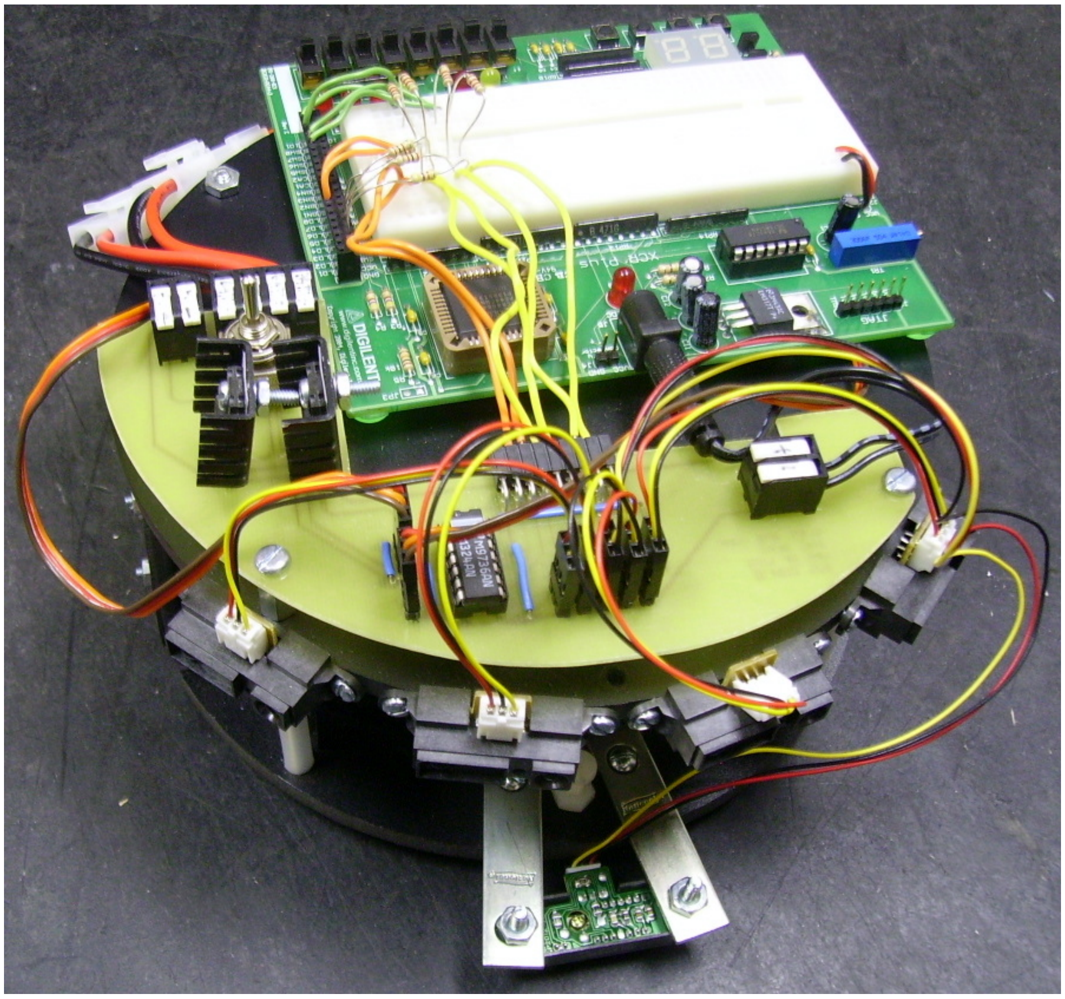

Figure 2.4: Xilinx XCR Plus mounted and wired to the robot platform.

Chapter 3

Solution

3.1 The Working Robot

In order to get the robot running in the maze and avoiding walls, many different systems must be designed. The first step I took was to test that the board worked and HDL Verilog code could be downloaded to it. Next, I programmed the robot to do simple movement. Once I had it moving, it was time to get its vision system working. Finally, I brought it all together to create a real time operating system that would navigate the robot through the maze. I will review each of these steps in this chapter.

3.1.1 Xilinx Board Choice

I worked with numerous Xilinx boards before finally settling on the XCR Plus evaluation board. It is an inexpensive (US $49.00) and simple board that is still powerful enough to control the robot. The XCRP features the Xilinx CoolRunner XC3064 CPLD. This evaluation board contains numerous features that cater to the needs of students. First off, the board can be programmed through the Joint Test Action Group (JTAG) interface via the parallel port. The board also features many types of on-board interface including eight light emitting diodes (LEDs), eight switches, four debounced buttons, and two 7-segment displays. For further information, see Appendix C, the ET-XCRP Reference Manual.

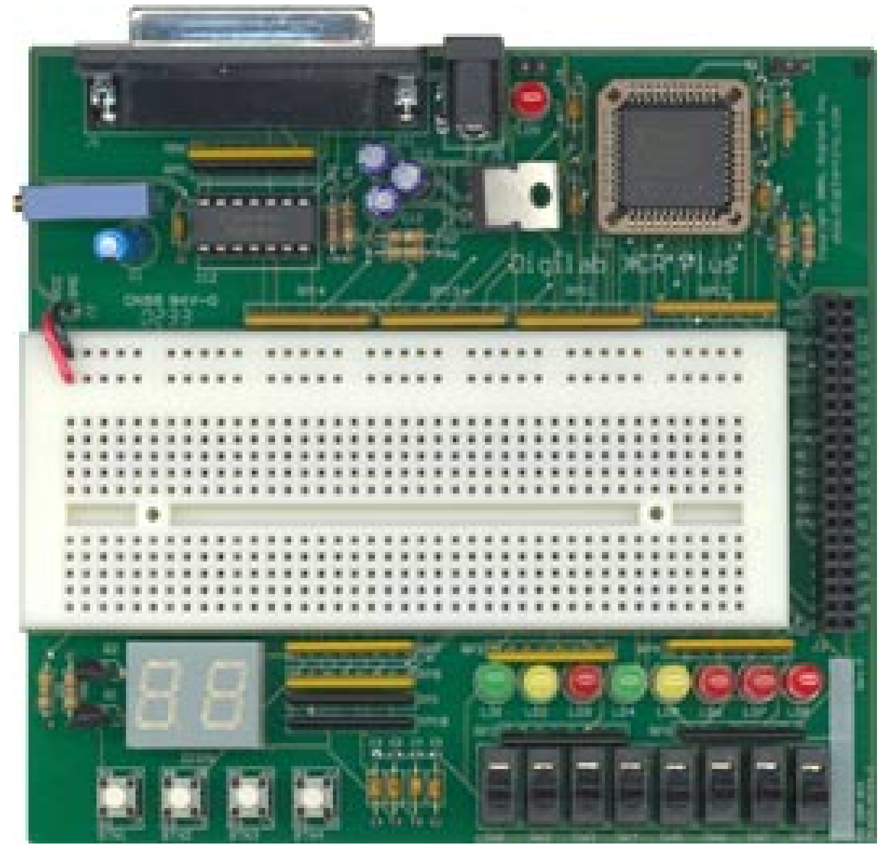

Figure 3.1: Digilent XCR Plus. [2]

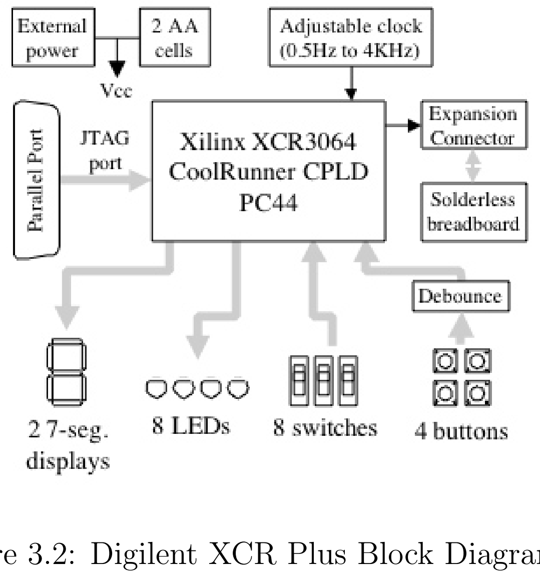

Figure 3.2: Digilent XCR Plus Block Diagram. [2]

3.1.2 Xilinx Board Test

The biggest challenge when working with a new evaluation board is getting a simple piece of code downloaded and working. Once that process is mastered, the possibilities are endless. To test the board, I wrote a small program that would test the input from the switches and the output to the light emitting diodes.

```
assign LED1=SW1;
assign LED2=SW2;
assign LED3 = SW1 & SW2;
```

After the code was successfully downloaded to the chip, asserting the switches would cause the correct LEDs to illuminate as programmed.

3.1.3 Motor Control

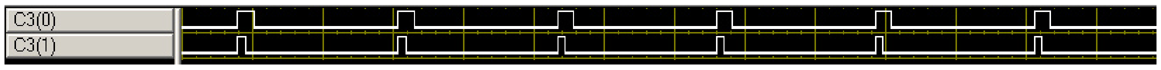

Figure 3.3: Logic Analyzer Waveform - Robot Forward

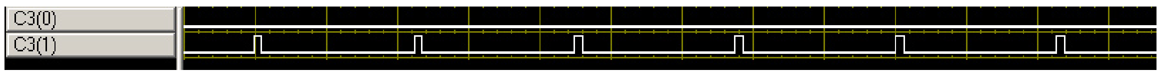

Figure 3.4: Logic Analyzer Waveform - Robot Right

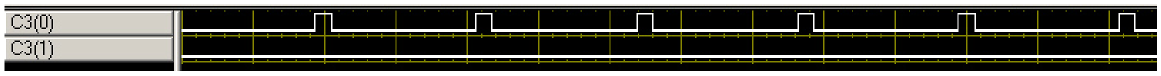

Figure 3.5: Logic Analyzer Waveform - Robot Left

To get the motors spinning, a PWM signal needed to be generated from the XCR Plus. This is done simply by keeping a running count of clock ticks and setting an output to high or low depending on what the count is. I coded the chip to output a logical high for 3 clock ticks and then low for 57 clock ticks to produce a 1 ms pulse for a 50 Hz frequency. The evaluation board has a potentiometer that allows the user to control the clock speed of the chip. Therefore, I was allowed to adjust the potentiometer until I had the exact signal I needed. I verified the correct PWM output with a logic analyzer with a sample trace provided in Figure 3.3 through Figure 3.7. It should be noted that due to the mechanical mounting configuration of the motors, opposite signals must be fed to the motors to render forward movement.

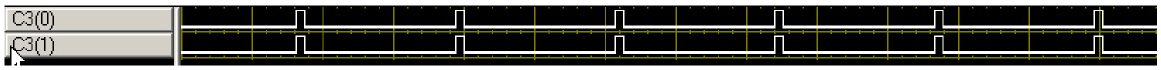

Figure 3.6: Logic Analyzer Waveform - Robot Hard Right

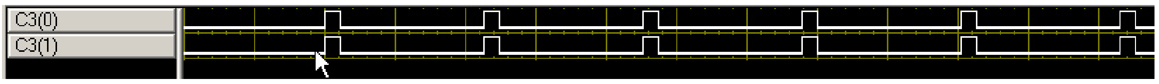

Figure 3.7: Logic Analyzer Waveform - Robot Hard Left

3.1.4 Vision

The vision sensors output an analog voltage signal, however, the XCR Plus chip needs a digital input to process what the robot is “seeing”. This was the tricky part of the interface because the XCR Plus didn’t have an onboard analog to digital converter. Along with that, no complex hardware could be added because many students would not have the background to work with it. (The EE4490 course prerequisite is EE2390 Digital Systems Design.) My solution was to route the sensor’s output through a resistor and then into a digital input on the XCR Plus board. This reduced the voltage to a level that was suitable to distinguish a digital 1 or 0 based on if a maze wall was within 14 cm of the sensor. This voltage reduction amount is based on the size of the resistor compared to the internal resistance of the input pins of the development board.

Because there are only four sensors and the board could take in eight inputs, I had two sets of four resistors providing the robot with eight digital inputs. One set of four resistors

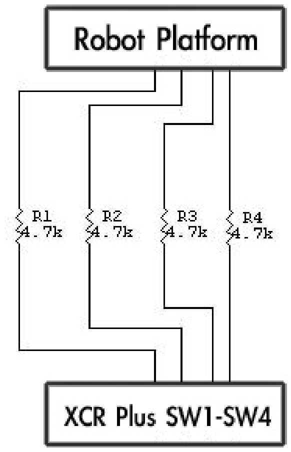

Figure 3.8: Robot Vision Wiring Diagram - 4 Resistors

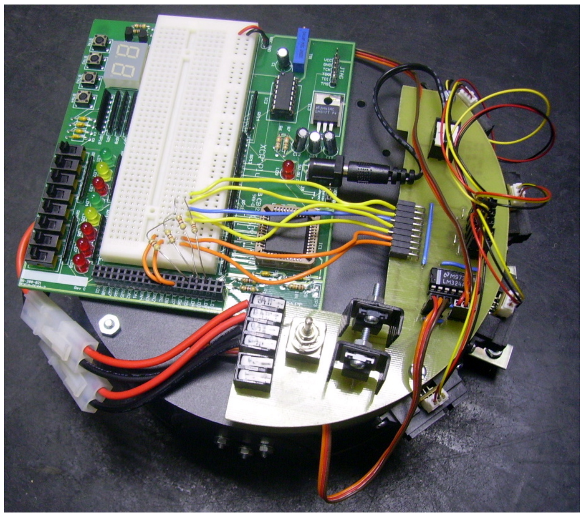

Figure 3.9: Robot Vision Wiring Picture - 4 Resistors

showed if a wall was within 14 cm and the second set of four resistors showed if a wall was within 21 cm. The circuit configuration is provided in Figure 3.10.

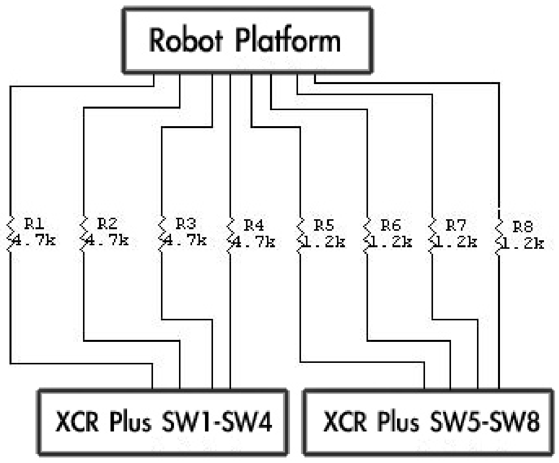

Figure 3.10: Robot Vision Wiring Diagram - 8 Resistors

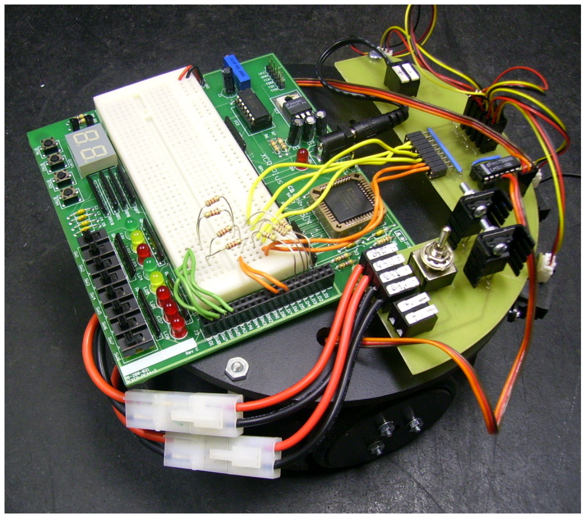

Figure 3.11: Robot Vision Wiring Picture - 8 Resistors

3.1.5 Simple RTOS

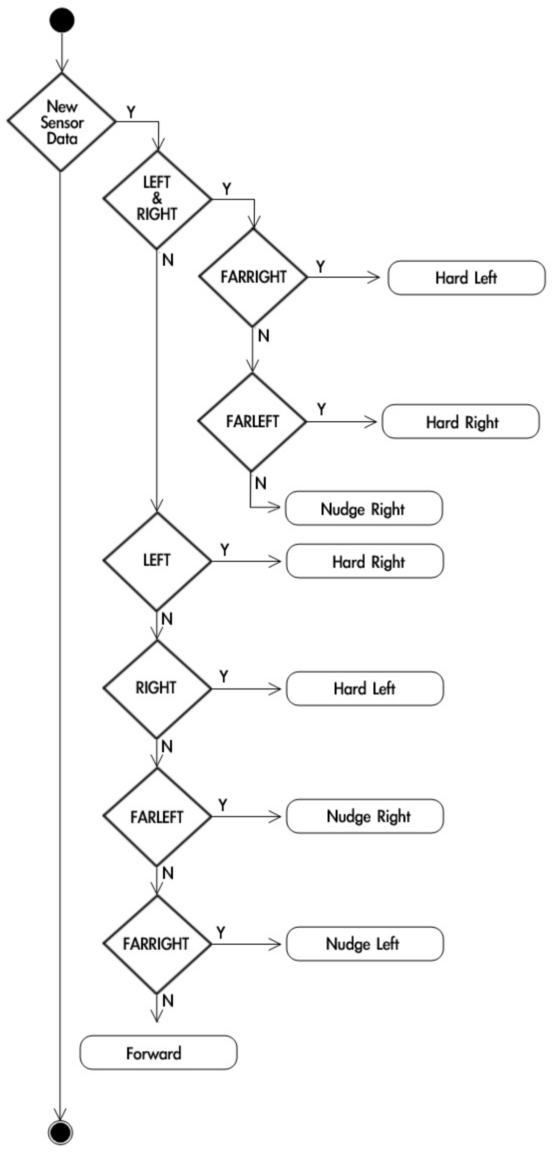

Figure 3.12: Simple Wall Avoiding Robot Real Time Operating System

Once the robot had vision and movement, it needed a brain, or real time operating system, to process the vision inputs and produce movement outputs. The algorithm I designed was quite simple; if the robot saw a wall to the left it would nudge right, if it saw a wall to the right it would nudge left, and if it saw a wall in front of it, it would take a hard turn away. A structure chart and a Unified Modeling Language (UML) activity diagram for the implementation is provided in Figure 3.12. A detailed listing of the Verilog HDL code is provided in Appendix A.

3.1.6 Advanced RTOS

With the eight vision inputs, the chip was able to do a little more advanced processing of its environment. For instance, when it approached a wall, it could tell whether a left turn or a right turn would result in a wall collision and could avoid turning the wrong way. Along with these advantages, it could also follow walls better and more precisely predict when it was stuck and needed to turn around. A structure chart and a UML activity diagram for the implementation is provided in Figure 3.13. A detailed listing of the Verilog HDL code is also provided in Appendix A.

3.2 The Guide

Once I had the robot working, I then went through and wrote a step-by-step guide that could take a student with only a limited knowledge of HDL and get them to the point of designing a robot that could navigate the maze and avoid walls. This guide covers the details of every step including; concepts, writing the code, compiling the JTAG, assigning pins, and downloading and testing each part of the robot's operating system. The guide most likely contains too much information to be used as a laboratory handout, but would work well as an online student tutorial. To turn each part of the guide into labs, the code from each section has been replaced by an explanation of what needs to be done at each section of the laboratory. A copy of the detailed tutorial is provided in Appendix A and a student copy with some information omitted has been included as Appendix B.

3.3 Short Term Solution

I have written the homework assignments used in the Fall 2005 offering of the EE4490 class. Therefore, I know that the robot design could easily be implemented along side of the current curriculum. In previous classes, students worked in a group to design a final project that they would simulate in software. This final project could be easily replaced by a final robot design project.

3.3.1 The Short Term Problem

The only problem with this short term solution is, due to lack of funding, the amount of robots would be limited and students would still have to rely somewhat on software simulations. Also, the students would have to be provided with most of the information that is needed to make the robot work so the amount of hands-on testing needed would be limited. This means some of that critical thinking that goes into the robot design would be lost. It is estimated that a total of US $250.00 per lab group would be required to fully implement this laboratory experience. That includes a robot platform (US $200.00) and an XCR Plus development board (US $50.00). It should be noted that we expect an enrollment of 48 students (24 laboratory groups). The ECE department currently has 15 educational robots available.

3.3.2 The Short, Short Term Solution

As mentioned previously, the class currently consists of a final project. One quick way to get a foot in the door would be to purchase 1 or 2 boards for US $50.00 to $100.00 and have a group follow my guide and implement the working robot operating system in hardware as their final project. This means no changes to the previous curriculum and little money spent. Plus, it would be a good way to prove that an HDL student could easily follow the laboratory guide to building the robot. I would suggest requiring software simulations from these students to help guide later groups.

3.3.3 The Short Term Solution

Software simulations could be designed to test if the robot was going to perform correctly in a given situation. Therefore, some class time toward the end of the semester could be set aside for all of the students to start formulating algorithms to control their robots given all of the necessary code to have vision and motor control already provided. Then, when their designs were working in software they could be given the chance to download their code onto one of the limited robots and see how it performs in the maze. Meaning, some sharing of robots could take place, but in the past, we have found that problems arise when too many students are operating the same robot. Therefore, for 50 students, we would have to purchase at least 15 full platforms at US $200.00 per platform and US $50.00 per micro-controller.

3.3.4 Short Term Thoughts

The EE4590/5590 Real Time Embedded Systems course continues the exploration of the HCS12 Mini-Dragon micro-controller. As advanced concepts are covered in the classroom, they are then implemented in the laboratory. Each of these laboratory sections focus on the different subsystems required to navigate the robot through the maze. Students have the ability to spend numerous full length laboratories building and testing their robots using concepts that are previously covered during regular class time.

Although they would be at a disadvantage to the embedded class, I would recommend that initially these two classes share the robot assets. The HDL class would still have the opportunity to see their code physically perform enough to make minor changes. Because of this disadvantages, I would recommend initially having a separate competition between the two classes.

3.4 Long Term

In the long run, I think we would all like to see the Verilog HDL class competing along side of the EE4590/5590 Real Time Embedded Systems class in the end of the year robot competition. To do this, at least 20 XCR Plus boards would have to be purchased at a cost of US $49.00 each. Furthermore, more robot platforms would also have to be built to handle this new load of students. Each of these robot platforms costs US $200.00 to manufacture locally. If we could accomplish this, the students would only need two or three class periods per month to get their robots programmed and running in the maze.

3.4.1 The Long Term Problem

The obvious problem with this solution is funding and the fact that students would need class time to complete the robot design. Because the class is only a 3 credit hour class, a separate lab section is unlikely. This would require compacting actual classroom lecture time to two 50 minute periods per week to allow inclusion of a two hour laboratory session. This solution would also require more work from the teaching assistant (T.A.) because he/she would be needed to assist the students in the laboratory. This course has historically enjoyed enrollments of up to 48 students per section. This would require the addition of four 12 student laboratory sections. This would entail two T.A. positions to support this new laboratory section.

3.4.2 The Long Term Solution

If eight or nine class periods were set aside to get the robot code written and tested, the students should be able to produce competitive robots, but an independent laboratory section would be much more advantageous. With the source code removed from the guide I have written, it would work as a laboratory guide to get students headed in the right direction without "spoon feeding" the answers. This student guide is provided in Appendix B. The T.A. could then use the guide provided in Appendix A as a whole for grading. If multiple T.A.s could not be provided, more information would need to be given to the students. This would be something that could be adjusted as needed.

3.5 Ideal Solution

The ideal situation would be for this class to have enough time and money for every student to have an XCR Plus board to work with. In this situation, I would suggest incorporating as much of the curriculum as possible with the hardware. Almost every homework assignment involves writing some sort of code that could be downloaded and demonstrated in actual hardware. On top of that, if the students had more time they could produce some very complex robots. The XCR Plus evaluation board costs approximately US $49.00. We could require the students to purchase the board as a required component for the course. We informally polled the students in the Fall 2005 offering of the course concerning this idea. They were generally in favor of this idea. Unfortunately, this is a hefty fee for a college student.

3.6 Recommendation

My short term solution could be put in to use right away. Students could take my guide and get a robot working within only a few class sessions. After a semester of this, and more money is allocated, a transition to the long term solution could be made where students spend much more time and effort perfecting their robots and eventually competing with the Embedded Design class.

3.7 Future Work

The digital design industry is migrating toward hybrid designs including both HDL components and micro-controllers. The next stage of development in advanced digital design course work would follow this trend. For example, the EE5390 Computer Architecture course could be taught using a hybrid design approach. The EE4490 HDL and EE4590/5590 Real Time Embedded Systems courses could serve as prerequisites for the course.

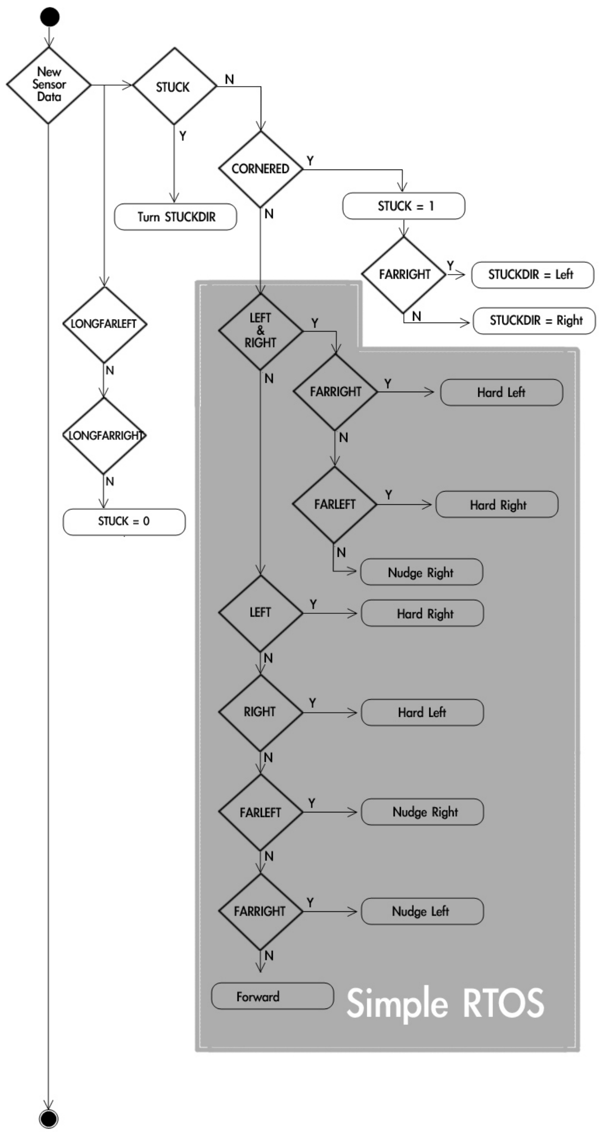

Figure 3.13: Advanced Wall Avoiding Robot Real Time Operating System

References

[1] PROFBOT Robot Owner's Manual Appendix D.

[2] Digilent DXCRP, https://www.digilentinc.com/Products/Detail.cfm?Prod=DXCRP

Appendix A

Full Guide

Appendix A is a full guide including all information and source code needed to design the robot to autonomously navigate the maze.

Appendix B

Student Guide

Appendix B is a student guide including a limited amount of information and source code that can be used as a laboratory guide without providing all the answers.

Appendix C

Xilinx XCR Plus Reference Manual

For further reference, the Xilinx XCR Plus Reference Manual has been provided.

Appendix D

PROFBOT Robot Owner's Manual

The Robot "Owner's Manual" was used throughout this project as a reference and is included as Appendix D.