

# Yellow Jacket 2005

2004-2005 Autonomous Vehicle Team of  
Cedarville University



Team Members:

Tim Linden

Tim Storer

Tyler Bixel

Faculty Advisor:

Dr. Clint Kohl

# Table of Contents

1. Introduction.....	3
2. Team Organization.....	3
3. Hardware Design .....	4
3.1. Robot Chassis.....	4
3.2. Drive Train.....	5
3.3. Motors and Gearing .....	5
3.4. Motor Controllers.....	5
4. Electronics System.....	6
4.1. Vision System .....	6
4.2. Sensors .....	7
4.3. Field Programmable Gate Array (FPGA).....	7
4.4. Power .....	7
4.5. Emergency Stop.....	8
5. Software Design.....	8
5.1. Overview.....	8
5.2. Image acquisition .....	9
5.3. Down Sampling .....	9
5.4. HSV conversion .....	9
5.5. Median Filter.....	10
5.6. Thresholding in the H and S plane.....	10
5.7. Color Segmentation .....	11
5.8. Hough Transform.....	12
5.9. Heading Calculation.....	12
5.10. FPGA Control .....	13
6. Robot Cost .....	14
7. Conclusion .....	15

## Certification of Design Work Performed on Yellow Jacket by the Cedarville University Design Team

I, Dr. Clint Kohl, Associate Professor of Electrical Engineering at Cedarville University, certify that the members of the Cedarville University engineering design team that worked on Yellow Jacket have done significant engineering design work on the robot that is equivalent to the work which is awarded credit in a senior design course.

Signed: Clint Kohl, Ph.D.                      Date: May 23, 2005

### 1. Introduction

The Cedarville University 2005 Intelligent Ground Robot team is proud to present Yellow Jacket. Yellow Jacket is a modification of the 2004 Cedarville University autonomous robot, Little Blue. Significant changes made to Little Blue include an enhanced wide-angle camera, shaft encoders, and a completely new software and firmware design. The simplicity of our design is intended to increase its reliability and reduce the overall cost while still accomplishing the task at hand.

### 2. Team Organization

The Cedarville University Yellow Jacket robot team was organized as follows:

Member Name	Responsibilities	Major and Class	Total Hours
Tim Linden	Software Design & Programmer, Electrical Design	Computer Engineering, Junior	320
Tim Storer	Electrical Design, Firmware Programmer	Electrical Engineering, Junior	250
Tyler Bixel	Electrical Design, Software Programmer	Electrical Engineering, Junior	250
		<b>TOTAL</b>	<b>820</b>

## 2.1 Design Process

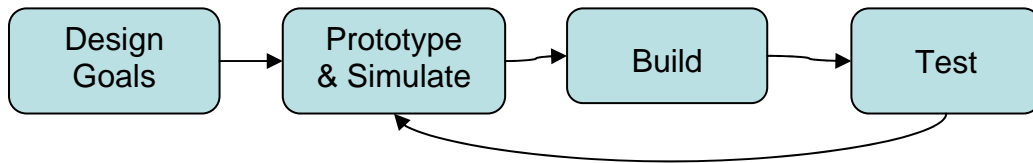


Figure 2.1 Design Process

The team started by setting the goal to significantly improve the performance of the previous year's design. The design team divided up responsibilities and worked in parallel on hardware, firmware and software design. The software and firmware designs have been through several cycles indicated in Figure 2.1 above.

## 3. Hardware Design

### 3.1. Robot Chassis

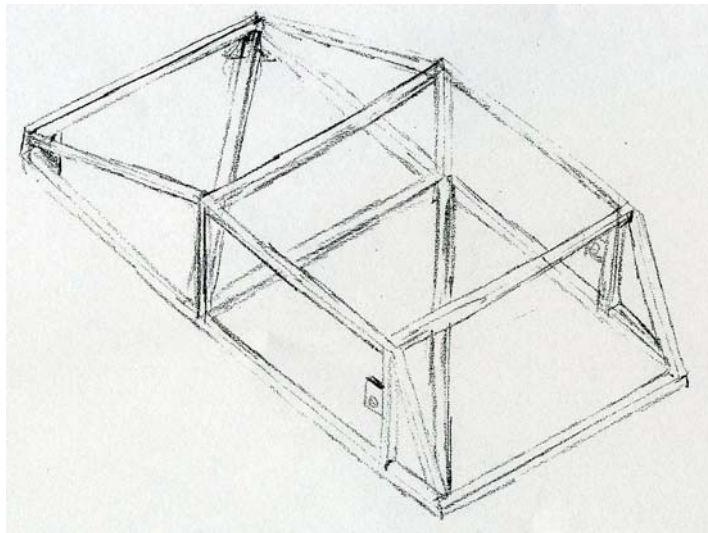


Figure 3.1 Initial sketch of Chassis Design.

The robot chassis was retained from our 2004 entry, which was custom designed and then built by our shop technician. It is 18 inches high, 24 inches wide, 48 inches long and was constructed using 1/2" square steel tubing. A completely new camera mount was fabricated on a post that extends from the top of the chassis enabling the camera to have a much better

viewing angle. Since most of the electronics are mounted within the chassis, we can protect them from rain by simply mounting plastic panels on the sides top and front.

### 3.2. Drive Train

The drive train consists of two custom fabricated 26-inch bicycle wheels mounted near the front of the robot and a single 5-inch caster at the rear of the robot. Each of the custom made bicycle wheels was and mounted on a shaft with a cogged pulley which is connected to the motor via a cogged belt so that each wheel it is driven independently; allowing the robot to maneuver in tight spaces.

### 3.3. Motors and Gearing

We chose  $\frac{1}{4}$  horsepower right angle gear motors from Bodine Electric Company to propel our robot. These motors have 24VDC windings and draw a maximum of 8.8A at max load.



These motors provide more than adequate power to ensure good ramp climbing ability. We have successfully climbed and descended grades of at least 20% without difficulty. Since the maximum output speed of the gear motors is 125 revolutions per minute, we used a 2:1 gear reduction to couple the motors to the wheels so that the maximum speed of our robot is 4.83 miles per hour. This ratio meets the speed limit of 5 miles per hour established in the competition rules.

### 3.4. Motor Controllers

The motor controllers we chose have built-in inrush current limiting and are rated for 20 Amps at 50VDC, providing plenty of margin for our 8.8 Amp motors. These controllers interface to the Flex 10K20 Field Programmable Gate Array (FPGA) and provide pulse width modulated speed control for maximum torque at all speeds. The drive train provides an excellent response time and is able to accelerate the vehicle



to top speed in under 4 seconds and to stop from full speed in under 2 seconds.

#### 4. Electronics System

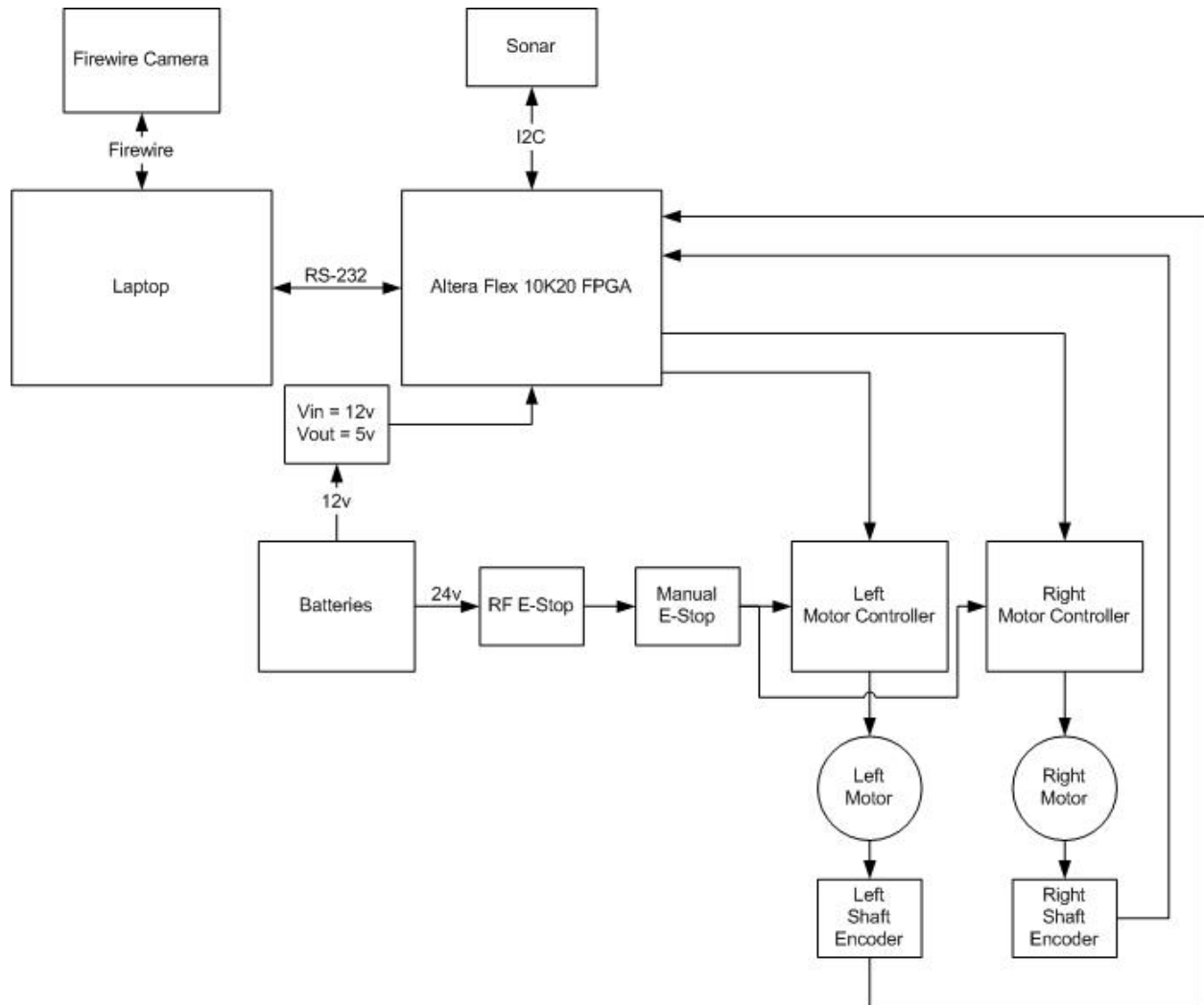


Figure 4.1 Robot Electronic System Diagram.

##### 4.1. Vision System

A Firewire camera was used to act as the eye of the robot. Using the Firewire camera to capture 640x480 RGB images gives a simple interface to the computers image processing algorithms. Using freely available drivers and libraries, software was written to interface to the Firewire camera and



display the resulting images of intermediate processing on the screen for debugging purposes. The image is processed to find lines and obstacles that must be avoided. A resulting map of impassable areas is used to calculate a heading which is split into a left and right wheel speed.

#### **4.2. Sensors**

In addition to our vision system we have two other sensing systems to aid in navigating the robot. The first is an SRF08 sonar sensor from Robot Electronics mounted on the front of the robot. The sensor interfaces to the FLEX 10K via the I<sup>2</sup>C bus. This system is primarily used as a worst-case backup, so that if the vision processing fails to detect an obstacle, the sonar will detect it and stop the robot before it runs into the obstacle.

The second sensing system is a shaft encoder on each of the two drive wheels. The signals from the shaft encoders are fed into the FPGA which uses it to measure the speed of each wheel. This system is used in a feedback control loop to ensure that the robot moves in accordance with the control system intentions.

#### **4.3. Field Programmable Gate Array (FPGA)**

Modern FPGA's allow for complex digital logic to be easily implemented in a reconfigurable chip. The Flex 10K20 FPGA from Altera is used to interface with the sonar and motor controllers so that the laptop can be devoted to image processing and path planning. The FPGA is also programmed to interface with shaft encoders which transmit the actual rotation speed. The laptop communicates with the FPGA via RS-232 and commands the speed to each wheel.



#### **4.4. Power**

We chose to use two 12VDC 22AH motorcycle batteries to power the robot. These batteries are connected in series to provide 24VDC for the motors and will provide approximately 2 hours of running time under full load. One battery is used to power the electronics interface board and the camera, and we can connect a power inverter to the other battery to power the laptop computer when its battery runs low.

#### 4.5. Emergency Stop

To enhance the safety of the robot our RF and manual emergency stops are connected in series with the battery power to the motors. This arrangement provides a safe way of stopping since the geared motors will not allow the vehicle to move when the power is removed. The RF emergency stop is an RC transmitter and receiver connected to a small servo that mechanically actuates a switch to stop the power to the motors. The manual emergency stop is a push button toggle switch with a red disc mounted on it and located at the rear of the robot.

### 5. Software Design

#### 5.1. Overview

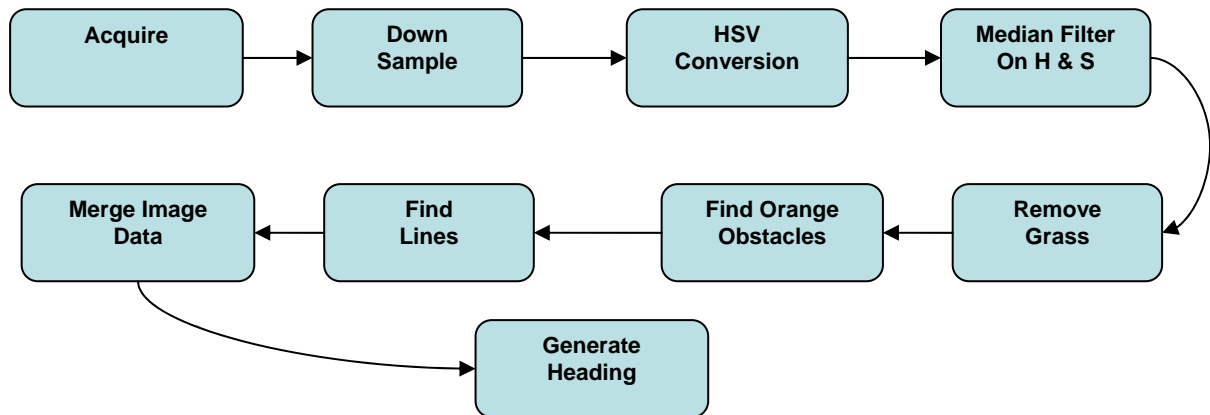


Figure 5.1.1 Image processing algorithm

One of our design goals was to maximize the speed of the image processing. To achieve this goal we used a Dell Latitude C800 Pentium III laptop with a 1 GHz processor and 512Mbytes of memory; this laptop also has a built in Firewire connector. The camera used was a Unibrain Fire-I Firewire digital camera which gave a simple interface to the laptop. An Altera FPGA was used to control the speed of each wheel and provide a feed-back loop to maintain the given speed independent of load and battery voltage. The FPGA receives the computer command for each wheel speed, through the RS-232 interface, and then performs feedback control to maintain these desired speeds. Sonar information is also passed back to the computer by the FPGA, through the RS-232 interface allowing the image processing algorithm to be aware of a obstacle that is very near.

C code was written from scratch on a Linux platform to maximize the speed of image processing. Drivers for the Firewire camera were obtained from [www.linux1394.org](http://www.linux1394.org), and for graphics display on the laptop, the Simple DirectMedia Layer (SDL) library was utilized.

## 5.2. Image Acquisition

The camera uses a Firewire interface to the laptop's built-in Firewire port. This combination was chosen to simplify the camera's hardware connections. Software interfacing to the Firewire camera was accomplished by using the Linux iee1394 drivers combined with the digital camera Firewire library available at <http://sourceforge.net/projects/libdc1394/>. This C library provided simple functions for camera control and image acquisition. The camera is mounted at nearly six feet above the ground and angled downward in such a way that obstacles are identifiable about 10 feet in front of the vehicle.

## 5.3. Down-Sampling

The acquired image from the camera is a 640x480 RGB image which contains significantly more data than can be processed on the laptop computer in a reasonable amount of time. This was overcome by down-sampling the image to 160x120. Down sampling the image is accomplished by averaging a square block of 4x4 pixels to obtain one pixel in the new smaller image. By using this method of reducing the image size it is simple to write efficient C code while retaining valuable information about the course.

## 5.4. HSV Conversion



Figure 5.4.1 Original

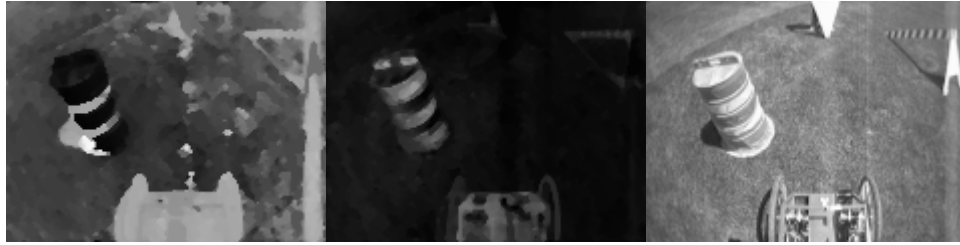


Figure 5.4.2 Hue

Figure 5.4.3 Saturation

Figure 5.4.4 Value

Segmenting the image based upon the color information allows the software to distinguish the differences between green grass, orange barrels and white lines painted on grass. Using the Hue (Fig 5.4.2), Saturation (Fig 5.4.3) and Value (Fig 5.4.4) color space, it is possible to pick out specific colors. As humans, we can observe an orange barrel and describe it as orange. The goal of this conversion is to allow the computer to identify barrels in the same manner.

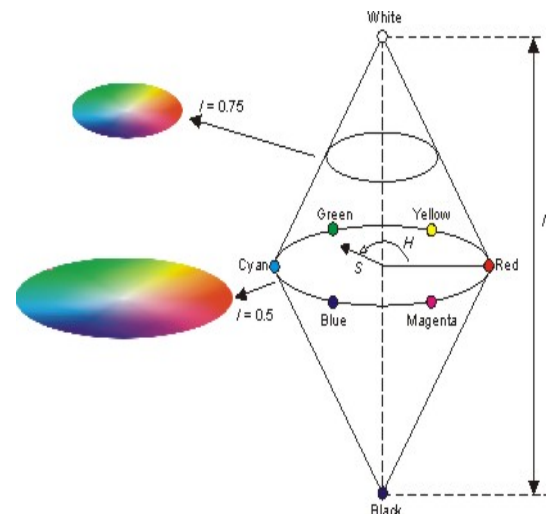
The acquired RGB image makes color segmentation difficult as colors vary in Red, Green and Blue and are significantly affected by intensity. The dependence upon intensity causes problems with shadows and cloud cover. Once the image is in the HSV color space the intensity can be ignored, greatly reducing the affect of shadows, sun glare, and cloud cover.

### 5.5. Median Filter

An image from the camera contains a significant amount of noise. This noise causes random pixels in the image to appear to be a part of an object or line. It is necessary to reduce the noise in the image in order to accurately identify lines and objects that must be avoided. To accomplish this task a median filter is used. This 3x3 median filter sorts all pixels in a 3x3 block with the current pixel at the center. Once the pixels are sorted the pixel in the middle (the median) of the sorted list is selected as the new value for the current pixel.

### 5.6. Thresholding in the H and S Plane

Thresholding is used as the basis for color segmentation in the image processing algorithm. It became very important to identify the color orange to be able to observe barrels on the obstacle course. Three different color spaces were analyzed to determine which color space would yield the best



results for identifying the color orange as well as other key colors.

Red-Green-Blue (RGB), Luminance-U axis blue component-V axis red component (YUV) and Hue-Saturation-Value (HSV) color spaces were analyzed by using a 3-D model of the color space. A simple program was written to visualize these color spaces as pictures were taken from the camera. In the RGB color space most of the color information lied along the x,y,z axis making thresholding difficult. YUV color space is significantly dependant upon intensity, reducing its effectiveness. The HSV color space is significantly less dependant upon intensity and separates colors allowing for thresholding regions based upon 2-D rectangles rather than 3-D polygons.

### 5.7. Color Segmentation



Figure 5.7.1 Segmentation of Obstacle



Figure 5.7.2 Segmentation of Line

The color space is segmented based upon two dimensional rectangles in the Hue and Saturation planes. Segmenting the image in this fashion allows the software to select specific color ranges. It is possible to select the shades of range in which an orange barrel exists and the shades of green in which green grass exists. Using this method, green grass can be removed from the image, revealing lines and obstacles. Lines and obstacles can be identified and marked so they can be avoided. Figure 5.7.1 shows an example of using color segmentation to identify a barrel and a saw horse. Figure 5.7.2 shows an example of using color segmentation to find the line painted in the grass.

## 5.8. Hough Transform



Figure 5.8 Hough Transform Result

Finding lines in a resulting segmented image becomes the next greatest challenge. Lines are not guaranteed to be fully connected in the image, therefore; a method was needed to make a single line crossing the whole image. The hough transform is used to accomplish this task. Using the hough transform is a two step process. Every possible line must be calculated for each pixel in polar coordinates to avoid the infinite slope on the Cartesian coordinate plane. The resulting two dimensional image is scanned for a maximum value yielding a single rho and theta coordinate. A line is calculated from the result and the region outside the line is marked impassable. Figure 5.8 shows an example of the hough transform result which was calculated from the image in Figure 5.7.2

## 5.9. Heading Calculation

A new heading is calculated from an approximation of the centroid of the resulting region in front of the robot that is not marked as impassable. This region is generated by overlaying the obstacle map with the resulting hough transform map to create a composite map of area that the robot cannot pass through. The image is known to contain areas that are not useful to the calculation of the centroid. A region of the bottom of the image contains a portion of the robot which is clearly not a region that should be considered, as this region is not in front of the robot. To eliminate this region, the binary map is windowed to remove this area. Impassable regions are defined as regions outside the line boundary or areas occupied by an object or behind the object. When the inverse hough transform is performed the region outside the line is filled in as black denoting an impassable area. Obstacles are extended to include the area behind the object. This is accomplished by scanning the object map from the bottom to the top. When scanning the object map any black or 'off' pixel is extended to

fill in the area above it. This is done because the robot cannot go through an object and therefore any area behind an object is also impassable. The approximated centroid is determined by summing rows and columns of the image to resulting in two one dimensional arrays. Each element in the arrays is normalized to one with the largest value in the array being one. After the arrays are normalized they are then multiplied by their x or y position where x and y are normalized to +1 or -1. The sums of the two arrays yields the resulting centroid.

### 5.10. FPGA Control

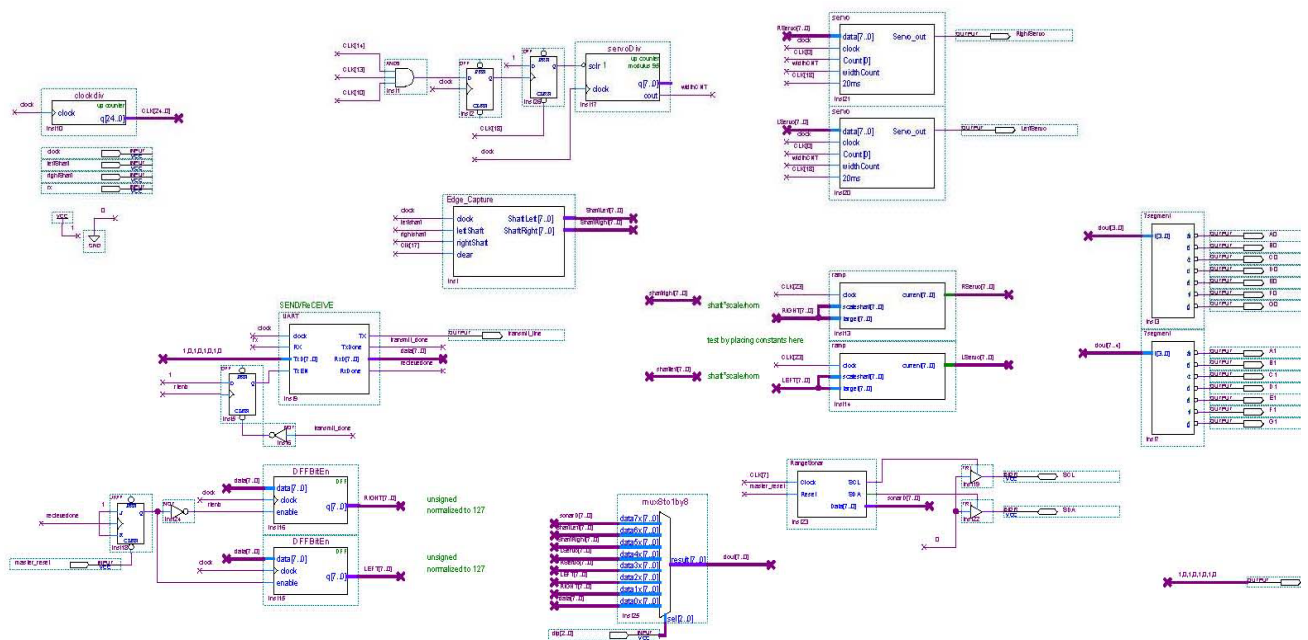


Figure 5.10.1 Top Level Digital Logic Design implemented in the FPGA

Control of the robot is accomplished by using the laptop to command the speed of each wheel and the FPGA to maintain the feedback loop to maintain the given speed of each wheel. This method of controlling the robot was chosen because of its flexibility. The differential steering control gives a simple mechanical solution for locomotion without restricting speed or turning ability. Using the computer to calculate wheel speeds allows for ease of implementation of different control algorithms.

The FPGA interfaces to the sonar via I<sup>2</sup>C and maintains the most recent distance value in inches. RS-232 is used to communicate between the laptop and the FPGA. The computer

sends the right wheel speed and the left wheel speed and will receive the most recent sonar value which it can use to determine if it has made an error in detecting an obstacle.

## 6. Robot Cost

Item	Unit Cost	Total Cost
Gear-motors*	\$450	\$900
Motor Controllers	\$100	\$200
Batteries	\$30	\$60
Chassis Materials	\$75	\$75
Pulleys and Belts	\$100	\$200
Bicycle Wheels	\$100	\$200
Caster	\$12	\$12
Fire-I Firewire Camera	\$83	\$83
Flex 10k FPGA	\$150	\$150
Sonar	\$50	\$50
Shaft Encoders	\$5	\$10
Electronics Interface Board	\$40	\$40
Emergency Stops	\$30	\$30
Power Inverter	\$50	\$50
Laptop Computer	\$900	\$900
Miscellaneous Parts		\$250
	<b>Total</b>	<b>\$3210.00</b>

\* Donated by the Bodine Electric Company.

## **7. Conclusion**

The overall goal of our design team's entry was to maximize performance while minimizing cost and complexity. Autonomous robots that meet the needs of a variety of customers must be affordable, reliable and safe. Our entry, Yellow Jacket, minimizes cost while maintaining adequate complexity, reliability and safety.

Yellow Jacket's hardware was programmed using some of the latest computer aided design software and an FPGA, making the robot more adaptable to any changes that may be required in the future. Overall, we have designed and constructed a simple-to-understand, easy-to-operate, and yet highly sophisticated autonomous robot which will meet the goals of the competition.