PROJECT PROPOSAL AND FEASIBILITY STUDY

Team 5
STORBOT

Matthew Lubbers
David VanKampen
Ryan Mejeur
David VandeBunte

SENIOR DESIGN
ENGR 339
December 3, 2008
CALVIN COLLEGE

Abstract

The proposed senior design project intends to offer an automated solution for storage and retrieval of goods in a warehouse. The final product will be a demonstration of storage and retrieval using a Xilinx XUP Virtex FPGA in a prototype system. In addition to this development board, multiple peripheral PCBs will be designed to interface the FPGA and servo motors. A server will interface with the host computer to provide necessary inventory data to the user and the retrieval robot. We expect the project to be completed by the beginning of May 2009, and have scheduled our time to achieve this goal. In this paper, we have provided the project objectives, feasibility study, system designs, and other related information concerning the direction of our project.

Executive Summary

Problem

Most factories today spend unnecessary man-hours for employees to leave their positions on the assembly line to go retrieve a small box of parts. This is a simple task, and it is not only below the difficulty of work these employees are capable of doing, but it also results in inefficiency. Employees often lose additional productivity when they are required to look up the item's location and then document what has been removed from the warehouse.

A second problem faced in these warehouses (or small stores) is the cost barrier to implement an effective solution. Companies are always striving to make their processes more streamlined and efficient, but there are few feasible solutions on the market. Also, the process of finding an appropriate solution is cost exhaustive in itself.

Solution

The latest trend in warehouse and factory product management is a push for autonomy. This is because autonomous systems are capable of performing many of the necessary tasks in these industries. These tasks primarily consist of organization, storage, and retrieval of parts.

Our team's solution is to develop a small prototype robot which can organize, store, and retrieve items, therefore eliminating the problem of wasted time. As previously mentioned, cost will be a significant driving factor in our design. Our product will not be marketable if it does not function as a quick turn, low cost solution. Therefore, it is critical to develop a prototype which is reliable, inexpensive, and fully functional.

Design Objectives

1. Use a free, open source VHDL processor core to serve as a robotic control unit
2. Develop an I/O address port to allow CPU access to motors through addressing
3. Design a track system
4. Design a moveable base with wheels powered by electric motors
    a. Capable of moving weight of entire device and load
5. Design an arm lift and extension modules
    a. Must be capable of lifting specified load weight of 5 pounds
6. Design a gripper unit with pressure sensitivity or optical alignment capabilities
7. Design a Graphical User Interface for sending commands and monitoring performance
8. If time allows, incorporate a Database Server to monitor the stock and location of items

Contents

Table of Tables

Table of Appendix Tables

Table of Figures

Table 1: List of Acronyms

AC	Alternating Current
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
DC	Direct Current
ECOS	Embedded Configurable Operating System
FPGA	Field Programmable Gate Array
GNU	GNUs Not Unix
GPL	General Public License
GUI	Graphic User Interface
IP	Intellectual Property
ISA	Instruction Set Architecture
Nm	Newton Meter
OS	Operating System
OSHA	Occupational Safety and Health Organization
PCB	Printed Circuit Board
PPFS	Project Proposal and Feasibility Study
RTLinux	Real-Time Linux
RTOS	Real-Time Operating System
SPARC	Scalable Processor Architecture
UART	Universal Asynchronous Receiver Transmitter
VHDL	VHSIC Hardware Description Language
VSHIC	Very High Speed Integrated Circuit
XUP	Xilinx University Program

Table 2: List of Terms [13]

DC Motor	Motor that forces electric current through a coil to produce mechanical torque
Development Board	A circuit board used for prototyping and testing designs
Firmware	A computer program embedded in hardware
Hardware	The physical components of a computer system
Operating System	System Responsible for the management and coordination of activities for computer resource sharing
Processor	Machine for Executing Computer Programs
Servo Motor	Motor with error feedback for precise position control, incapable of multiple full rotations.
Software	Computer programs and procedures that perform tasks on computer systems
Stepper Motor	Motor with multiple “toothed” electromagnets that allow position encoding without a feedback mechanism

1 Introduction

According to the Bureau of Economic Analysis (BEA), the warehousing and storage industry grossed 48.6 billion dollars in 2006, which accounts for approximately .92% of the United States' total gross domestic product (GDP*)* [26]. Many improvements are being made to the warehousing and storage industry to streamline efficiency and maximize profitability. The new improvements to this industry include many different types of automated equipment from high-speed pallet shrink wrappers, to autonomous forklifts. Our design is aimed at providing a higher level of automation to the warehouse industry, especially the small and medium sized warehouses.

Today's warehouses require a very high level of control and efficiency due to the enormous size of the warehouses and the volume of product moving through them. Our design project attempts to solve the problem of controlling goods, while increasing the efficiency of warehouses by providing an inventory database and a robotic retrieval and storage system. In principle, the system will be controlled by a host computer that will interface with a FPGA driven robot and a Goods Control Server. The host computer will be password protected so that it is only accessible to authorized personnel. This security measure provides greater control of all goods in the warehouse. The host computer will track items coming in and out of the warehouse, as well as send updates to the Goods Control Server as needed. The host computer will communicate with the retrieval robot through the FPGA to request that goods be moved, retrieved, or stored in the warehouse. The FPGA will handle all requests from the host computer, but will be able to operate independent of the host computer while executing a command.

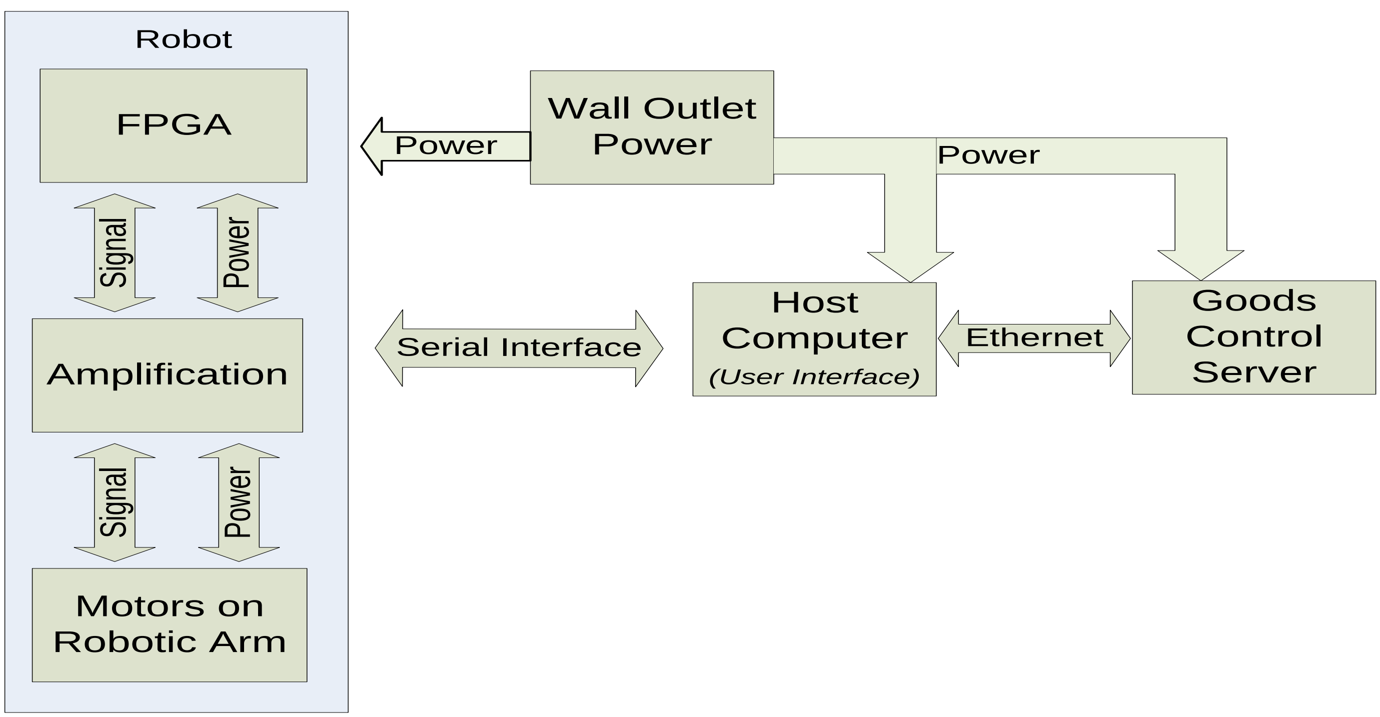

Figure 1: Top-level system view

The robotic system will be controlled by the FPGA through an amplification stage, which amplifies the power and control signals to operate the robot and its arm movement. The robot will be able to store and retrieve goods by using a three axis arm connected to a mobile base, which moves to the desired item location in the warehouse on a track system. The robotic arm will likely operate with servo motors, DC motors, and pneumatics. The robotic retrieval system will be designed for small items weighing less than five pounds, such as: plastic parts, screw boxes, nail boxes, gaskets, small motors, etc.

The complete system design will be flexible, allowing for easy future implementation into existing warehouses. Our project is to design a prototype robotic system which can perform the desired tasks, while staying within our $1500 budget. We hope to ultimately develop a functional prototype that moves on a small section of sample track. The functional prototype will be a fully incorporated functioning system and will demonstrate the concept of a full warehouse system, providing a platform for future development.

Our goal is to design a user friendly system that is intuitive to those operating and maintaining it. We will make every effort to adhere to the codes and regulations of governing agencies by implementing safety circuitry (i.e. emergency stop) and equipping the system with the necessary sensors to reduce risk to the personnel operating the equipment.

The remainder of this report describes both the technical and functional aspects of the design, as well as the feasibility of the project from an economical and technical perspective.

2 Team Information

The design team consists of four senior electrical engineering students: (left to right) Dave VandeBunte, Matt Lubbers, Ryan Mejeur, and Dave VanKampen. Dave VandeBunte is from Grand Rapids, Michigan; he is pursuing a major in Electrical Engineering and a minor in Computer Science. Dave is interested in

Figure 2: Team Five

computer hardware design and embedded systems; he currently works at DornerWorks Ltd. Matt Lubbers is from St. Joseph, Michigan; he is pursuing a major in Electrical Engineering and is Automotive Service Excellence Certified. Matt has worked in and is interested in electrical applications in the automotive industry. Ryan Mejeur is from Kalamazoo, Michigan; he is working on a major in Electrical Engineering and a minor in Biochemistry. Ryan is interested in pharmaceutical process machinery and is going into the US Army National Guard ROTC upon graduation. Dave VanKampen is from Portage, Michigan; he is striving for a major in Electrical Engineering. Dave is interested in embedded systems and hardware design and currently works at DornerWorks Ltd.

3 Project Objectives

The project has five design objectives, which are: selecting and modifying the processor core, writing software programs that store inventory information and control the robot, designing the peripheral circuitry, specifying the electronic motors requirements, and designing the mechanics of the robotic arm.

3.1 Select and Modify the Processor Core

The FPGA will be a Xilinx XUP Virtex II Pro development board with dual Power-PC cores and 30,816 logic elements, or a Xilinx XUP Virtex-5 development board with the same cores and even more logic elements. Both of these development boards provide a powerful platform to mount the LEON3 or openSPARC processor cores. The XUP Virtex II Pro development board is part of the Xilinx university program and was acquired at a reduced price of $350, a price reduction of $1000. In addition to the Virtex II Pro development board, the team received a Xilinx XUP Virtex-5 Evaluation Platform from the openSPARC university program as a donation. Both of the development boards are compatible with the LEON3 processor core. The LEON3 processor is an open source processor written in VHDL code by Gaisler Research Inc. and is designed based on the SPARC processor architecture with an emphasis on real-time processing. VHDL is a programming language used to design hardware that can be compiled and embedded in FPGAs. The LEON3 core is open source, making it very customizable for the needs of the peripheral interfaces and the programs we will be creating. Additionally the LEON3 core is optimal for real-time applications because it has multiple interrupt levels allowing the processor to respond very quickly to important incoming inputs. We have successfully loaded both the openSPARC and LEON3 processor cores on the FPGAs, and we are now in the process of modifying the LEON3 processor core to implement an I/O port. This I/O port will be used to interface with the robot's mechanical drives and receive real-time inputs from the robot's external sensors. The next step will be to successfully send and receive external signals to and from the processor core. Our conditions for successful processor core design are: the ability to create a fully functioning processor core able to run our programs, receive real-time feedback from external sensors, send control signals to the electromechanical drives, and communicate with the host computer through a serial connection.

3.2 Write Software Programs

We will write the applications in programming languages such as C, C++, and possibly JAVA. One program will be written for the goods control server, one mounted on the FPGA to control the robot, and a third program will be written for the host computer. The host computer's program will have a Graphical User Interface (GUI) that allows the warehouse personnel to operate effectively and quickly. The host computer program will control the robot and also the goods control database. Our team will structure the programs in an object oriented manner to allow for easy debugging and easy upgrading. The next step is to write a program that will run on the processor core. Our conditions for successful software programming design are: creating a GUI which runs on the host computer and interfaces with the robot and server, coding a server program that stores inventory data, and coding a program that operates the robot and runs on the FPGA.

3.3 Design the Peripheral Printed Circuit Boards

The peripheral circuit boards have not been designed yet, but the team has concluded that a peripheral interface board is needed to drive the stepper motors that are going to be used on the robotic arm and track system. The peripheral board will be capable of delivering the maximum current and voltage to the stepper motors, and will also provide protection for the users and the FPGA development board. After the electric motors are ordered, we will begin the design and layout of the PCB and then order the necessary components to populate the circuitry. Our condition for a successful circuit design is to construct a PCB that receives signals from the FPGA, amplifies the signals to the electromechanical drives, and protects the FPGA and warehouse personnel.

3.4 Create Design Spec for Electronic Drives

The team expects to use 3 stepper motors and a pneumatic solenoid to drive the mechanics of the robotic arm. We are currently undecided as to which type of DC motor to use for the track system, but we have discussed using a direct drive type of DC motor for cart mobility. These motors will be driven by DC current that is supplied by an AC to DC power supply. We decided to used DC current because DC motors are easier to implement, control, and are less costly than their AC counterparts. We decided to supply the robot with AC power because it is easily available through the wall outlet, allowing for easier implementation of the robot in it operating environment. We will develop a design specification for the DC motors after we have successfully designed and analyzed the mechanics of the robotic arm and track system. Our conditions for a successful drive specification are: meeting the electrical power requirements, moving the robot in a timely manner, and operating at or above the minimum load requirement.

3.5 Design Robotic Arm and Hand Gripper

The robotic arm is currently designed to have a height of four feet, a reach of two feet, and three axis of movement. A picture of the robotic arm is shown in the hardware section. We will design the arm to be light weight, durable, and easy to maintain. We will use a pneumatic closing mechanism for the hand, to reduce inertia on the horizontal component of the arm. Stepper motors were chosen to drive the robotic arm because they are easy to control and are much cheaper than servo motors. The next step will be to build a wooden model of the arm for testing and design improvements. Our conditions for a successful arm design are: that it is a simplistic and maintainable design, it is able to handle the minimum load requirements, and that it operates to our satisfaction.

4 Intellectual Property

We are not concerned with any Intellectual Property rights for our project because we are using an open source core, and therefore must publish all changes we make to it. The design itself will be completely independent of any innovative designs, but rather composed of many small mechanical parts to create a complete system. The main Intellectual Property concerns will revolve around protecting and dealing with the licenses for all software and hardware code.

4.1 Patent or Trademark Protection

It is not the intent of our design team to pursue any patents at this point in time. The hardware and software programming will be hidden and protected, which we feel will provide enough legal protection and will be adequate for protecting the company from “design poachers.” We do not feel that it is worth investing the projects time and money into a patent for our design.

4.2 Software Licensing

Gaisler Research Inc. [14] developed the LEON3 core and offers the source code openly to the public under the GNU GPL license [24]. It is therefore available to download, view, and implement this code. If any changes are made to the source code, the GPL license requires you to publish these modifications publicly. If we decide to modify the code, which we likely will, we will make our changes available through the Gaisler message board [25]. However, Gaisler also offers licenses which can be purchased by developers who make architectural changes and wish to keep them proprietary.

The only architectural changes we plan to make involve address mapping in an I/O device on the peripheral bus that the LEON3 already has. This is a standard modification which does not lead to any unique or patentable ideas and therefore there is no harm or loss associated with making our changes public.

5 Similar Products

The analysis of the intended market competition is a very difficult yet delicate process. It is important to determine the different levels and purposes of the mechanical automation companies and their capabilities being offered. It is then also important to evaluate whether these are turnkey solutions or whether they require a facility level integration which must be implemented while the original factory is built.

5.1 Existing Competition

Most of the existing warehousing automation companies today develop large scale products. They intend to have a warehouse run entirely from automation, and design palletizing robots that can handle very large loads. However, these systems often require millions of dollars to install and operate, large spaces to maneuver, and a long factory downtime to install. However it is possible that these large businesses are capable of quickly reducing the scale of their current product to create a shorter time to develop a similar product. The matrix shown in Table 3 outlines the current competition with respect to the intended small scale market only.

The strength of the existing competition lies within their experience and existing customer base. If the competitors' customers can be reached and information about our design is provided, we believe that we can attain some of this valuable market share by emphasizing customer service and specific solutions to their needs.

The weaknesses of the existing competition are what really create the appeal for this product. There are no apparent companies offering a very low cost and quick turnkey solution for automation. Many of the current companies require ground-up integration, which provides a large time and money commitment which most companies are unable to make. If we can convince people that we can be operational within a few days for a low price, people will be inclined to try our product.

Table 3: Competitive Matrix

FACTOR	**Senior Design Team 5**	**FATA Automation**	**Kiva Robots (Amazon)**	**Westfalia**
Low Price	5	1	3	3
Superior Quality	5	4	4	4
Customizable Product	5	3	2	4
Unique Features	4	5	4	4
Rapid Product Delivery	4	3	3	4
Customer Service	5	2	4	3
Total	28	18	20	22

5.2 Future Competition

As previously mentioned, most of the future competition will come from companies already doing work in the warehouse automation field such as such as FATA [21], Westfalia [22] or Kiva [23]. These companies will be able to assign their Engineers to design a new product, and utilize their currently existing implementation strategies to focus on developing a low cost solution. This will provide good competition for our design, and drive the market even faster to meet the customers' needs.

Another potential future competitor will be Engineers within warehouse companies themselves. If the system offered is simple and inexpensive enough, companies may be able to save time and money by simply designing a system themselves. This will allow for total customization, task specific programming, and full time on-site customer and robot support. In order to prevent this from becoming an issue, we must make clear that the design time is as quick as possible.

6 Design Norms and Christian Perspective

Design norms look at the project's design in a moral rather than physical and technical perspective. The design norms study will look at how this project will affect society and the environment in which the system will operate. This team is deeply dedicated to creating a finished product that is morally and ethically edifying to those using it and those affected by it. According to Professors Gayle Ermer and Steven VanderLeest, "Normative design attempts to balance design trade-offs not only among technical constraints but also among ethical constraints. Designing to such norms forces the engineer to consider the broader impact of the design on the society in which it will be embedded. Design norms include concepts such as cultural appropriateness, transparency, stewardship, integrity, justice, and caring [20]."

This design team is committed to making the robotic warehouse system culturally appropriate. We will be sure the design is appropriate for warehouse applications and does not interfere with the culture it will be implemented in. This minimal impact will be accomplished by creating a robot that is aesthetically pleasing and does not interfere with the warehouse personnel's activities. Although we will not have enough time during the senior design to create multiple language capability in the GUI, we would like those developing the system beyond senior design to create multi language capabilities for the host computer interface. The system will be user friendly and easy for warehouse operators to learn.

This design will also be as transparent as possible. We want the robotic warehouse system to be aesthetically pleasing to those operating and working around the equipment. The system will be reliable so that those who operate the equipment will be effective and efficient. We would like the equipment to operate in the background and not require excessive movement or work by the robotic machinery.

Our design will fully consider stewardship of the environment and humanity. The design will be efficient, allowing personnel to operate independently from the system which will optimize their time. The design will be energy efficient and made from recyclable materials to minimize the impact on the environment. The power consumption is especially a big concern because of the cost impact and more importantly the carbon emission impact that the system will have on the global environment. The system will also have a small footprint to optimize space; this will allow warehouse owners to be better stewards of the land that they own. Also, by creating a more efficient inventory system, we hope that the loss of inventory due to expiration and misplacement will be reduced so the industry will produce less waste.

In this project we hope to maintain integrity and promote integrity for those who will use our product. Hundreds of millions of dollars are stolen each year in merchandise by those working in warehouses and stores. This system hopes to reduce that high dollar loss by creating electronic accountability for the employees. We will accomplish this through an electronic inventory database that will track items coming to and from the warehouses. Lastly, we hope to maintain integrity with the design by not eliminating jobs, but rather make the people more efficient at their jobs.

We hope the design will be caring to all who use the equipment. This system will make the warehouse storage industry less physically demanding. Since warehouse jobs would be physically less demanding, those that normally could not get a job in a warehouse due to physical disabilities could now get a job. We would also like the system to be caring of those that are not technically literate, by making the GUI as user friendly as possible. Lastly, we feel that the design should not cut any jobs, but rather be supplemental to those already working in the warehouse, allowing them to work in other ways; such as, equipment technicians or maintenance personnel.

7 Preliminary Design

The preliminary design of the project is conceptually broken into three sections: hardware design, software design, and system design. The problems encountered and design solutions for each of these areas are presented below.

7.1 Hardware Design

7.1.1 Tools

The hardware design tools for the development of the project's FPGA core architecture were provided by Xilinx Inc. (both of our prototyping boards use Xilinx FPGAs). The Xilinx "ISE Design Suite" was used for writing, debugging, loading, and compiling the boards. Writing, debugging, and compiling of the VHDL source code was performed in ISE, while loading was performed in iMPACT (the Xilinx on-chip JTAG loader and debugger interface for Windows) [19]. The Design Suite also comes equipped with an Embedded Developer's Kit (EDK). The Xilinx EDK is useful for assembly level programming, which helps test the core's capabilities once it has been implemented in the FPGA. The Embedded Developer's Kit also manages address bus mapping, so the user can define low level details such as the interrupt enable signals and assign address spaces.

7.1.2 Development Boards

One development board designed for core and high current peripheral hardware testing is the Digilent Inc. XUPV2P [18]. There are three high current power supplies on the development board which will prove to be useful features for testing motor control. The XUPV2P board implements a Virtex-II Pro FPGA. The Virtex-II chip includes a hard PowerPC processor and can incorporate a SPARC processor, the instruction-set architecture this project is based upon. The reasons we chose this architecture is discussed in detail in the core architecture section below.

The XUPV2P board is intended for academic use. It has most of the common interfaces that a typical FPGA core may need, which makes it ideal for this project (since the capabilities required are always subject to change). Our design may require the FPGA to drive the GUI display, receive inputs from either a PS/2 port or serial connection, and handle a number of sensor inputs. We have chosen this development board because it can effectively achieve these tasks and more.

Recently, the design team received a donation from the openSPARC University Program [17]. The donated board was a Virtex-5 Evaluation Platform with OpenSPARC Development Kit (also by Digilent). This board is capable of performing in several different configurations. One important configuration is as a PCI card into a host computer. Depending on how our project develops, this may become a favorable market feature because it reduces the visible external hardware and creates a simple insertion method allowing it to be easily ported into existing manufacturing situations.

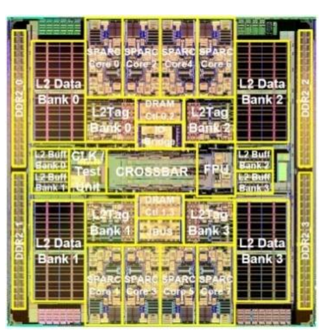

Figure 3: UltraSPARC Die

The hardcore of the Virtex-5 FPGA is eight parallel SPARC cores. If the core chosen is indeed the LEON3 or openSPARC, then it will ensure easy programming and implementation, given the similarity of the processors. A block diagram of the Virtex-5 ML509 board is shown in Figure 4.

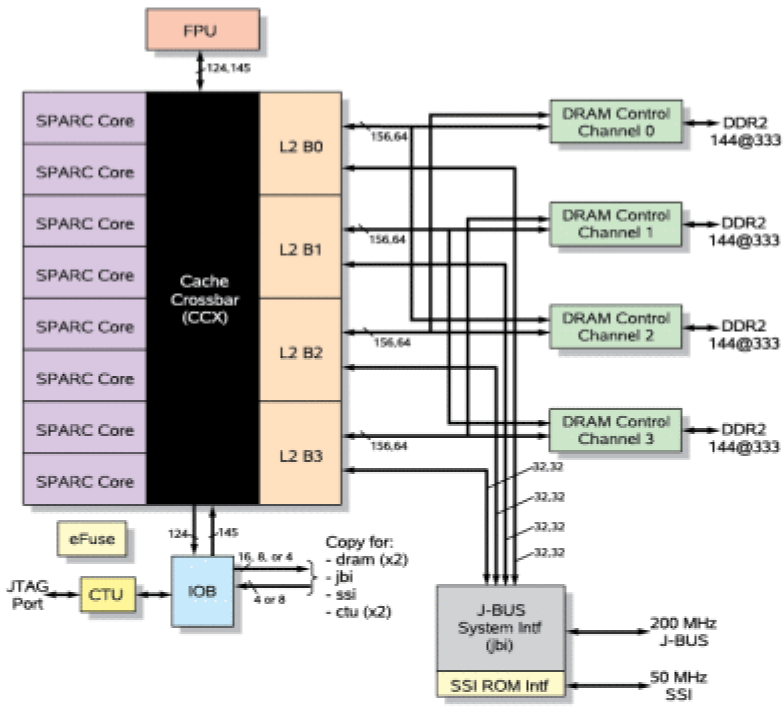

Figure 4: Virtex-5 FPGA Structure

A notable feature of this core is the DRAM controls, which allow for quick data storage and retrieval, lending themselves to real time capabilities. The interrupts that are involved in a real time system will demand significant process data storage.

7.1.3 Core Architecture Alternative Solutions

Table 4: Core Decision Matrix

Criteria	Weight	Leon2	Leon3	SPARC	ARM	PowerPC	MIPS
Cost	9	10	10	10	10	1	10
Vendor Support	8	8	8	4	2	5	2
Available ISA	3	8	9	10	6	6	3
RTOS Compatability	10	7	8	8	6	9	4
Development Tools	7	6	6	7	5	8	4
Documentation	6	5	9	8	6	8	3
Total		320	357	329	255	261	201

The decision matrix above outlines the criteria we considered while choosing between alternatives for the processor core. Obviously, we are assuming any code we use is open, so this is not included in the decision. However, some architectures are open at a price to the user, such as PowerPC. PowerPC licensing can cost upwards of $64,000, [16] depending on the application. So, although the core code is available, it is not economically feasible for this particular project. All the other cores that were investigated provided all source code at no cost to the user, provided programmers were willing to publicize any architectural changes they made.

The purpose of the vendor support criterion is to convey whether or not the code developers are willing and available to back up their code, answer any developer questions, and provide guidance for project development. This can be through a user message group, email or phone technical support, or any other manner of providing direct human advice for a specific problem.

The Available ISA (Instruction Set Architecture) criterion outlines how well defined the assembly code is for that particular architecture. The workgroup is readily equipped with textbooks for the SPARC architecture, which for the most part also applies to LEON2 and LEON3, but the other architectures ISAs are not as close at hand.

Development tools provided by the core vendor are important for several reasons. They allow the developer to spend more time writing code and testing out ideas, and less time dealing with subtle, unclear errors that might come from having to use a less effectively shaped development suite. One can write VHDL code in any text editor, but it helps to have a suite tailored to the specific target device because the compiler can then efficiently shape it, point out potential areas for improvement, allow more user interaction in floor planning, and other custom suggestions.

Documentation is very valuable in an academic project such as this. It is more important that the users learn about the basics structures of ISAs and processor cores than save money or turn out a product quickly, and documentation lends itself to achieving that goal.

With all factors considered, the SPARC architecture appears to be the best choice. The SPARC processor interfaces with any necessary peripheral components via the AMBA interface, a convenient proprietary bus from ARM. This interface is also tied to the serial UART interface used by most PCs, facilitating effective debugging.

7.1.4 Peripheral Hardware Interfacing and Drivers

Depending on whether an RTOS is implemented, we may require that a majority of the electrical design done for the robot to computer interface to be performed in hardware, writing code with the VHSIC Hardware Description Language (VHDL). These drivers will control the motor direction and speed, as well as the proximity monitoring.

Another option is to write modular hardware drivers that can be driven using software device drivers within the operating system. This will still be done using VHDL, but implemented as separable blocks, rather than one cohesive, fully functioning unit. This particular arrangement is shown below. As mentioned earlier, the separate blocks controlling external hardware will tie to the SPARC processor using the AMBA bus interface [15].

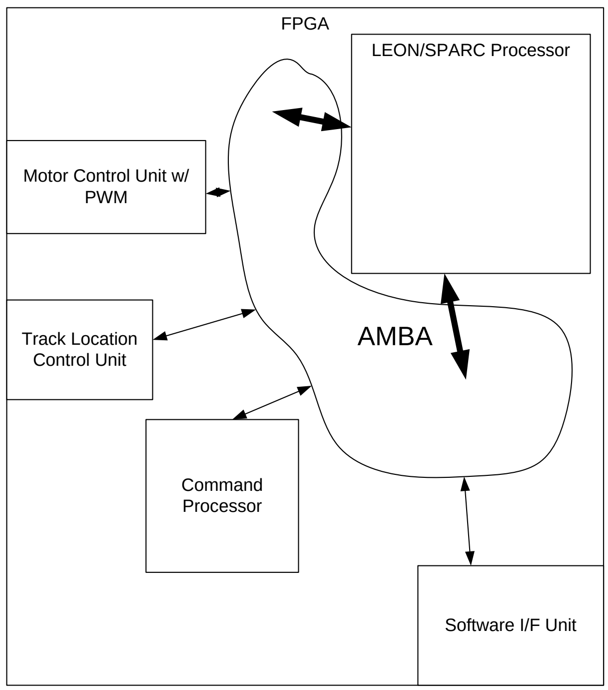

Figure 5: FPGA Layout and Interfacing

7.1.5 Electric Motors

There are several electric motor options available. The track driver motors will likely be DC Motors, since these are capable of turning full revolutions at high speeds, and providing high amounts of torque. A separate braking mechanism might be incorporated in order to allow for precision stopping along the track, but the current plan is to use the DC track motor to place the device in the proper position.

7.1.5.1 Stepper Motors

For arm assembly rotation, a stepper motor will be used. Stepper motors can complete full rotations, usually at slower speeds than DC motors. They are also capable of precision angling, which will allow for accurate object locating, as well as the ability to make up for slight track placement errors. This stepper motor will need to produce enough torque to turn the arm while in the unextended position, both with and without an added applied load.

7.1.5.2 Servo Motors
For arm joint control, the motor of choice will be the servo motor. Servo motors do not complete full rotations, usually impeded by a mechanical device on the rotating gears. Servo motors are capable of precise rotational placement, which will allow for minute adjustments, allowing the arm to continue correcting its own position as the object to be lifted is obtained.

7.1.6 Arm Construction Alternative Solutions
We considered two options for the design of the mechanical arm, referred to as the human arm structure and elevator arm structure.

7.1.6.1 Option A
The final design for the arm assembly is not completely determined. Depending on the results of pending market and feasibility studies, the project requirements might dictate an entirely different type of arm unit, so the following diagram and explanations are written within this context. For this first option, the proposed plan is to imitate a human arm, with a shoulder, elbow, and wrist joint. The wrist joint will be detailed more in the hand construction section.

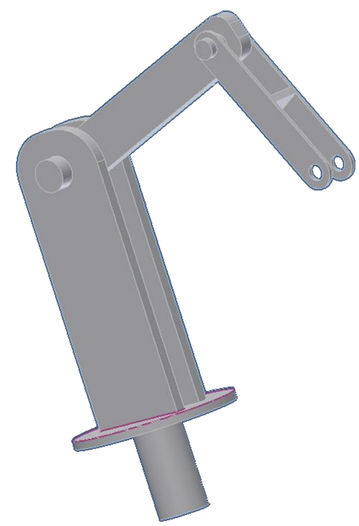

Figure 6: "Human" Arm Structure

The design of the arm and motors needed must be based on the weight of the object lifted, because weights, torques and moments exerted on each joint are base earlier joints they hinge on. So, if the plan is to lift a 5 pound object at a radius of three feet, the solenoid unit in the hand structure must be capable of exerting 5 pounds of force. A solenoid with this force requirement averages around 4 ounces, or ¼ pound. The material of the hand itself will likely be aluminum or an equivalent metal, weighing approximately 1/3 pound. Therefore, the hand structure will be approximately ¾ pound. If the forearm unit is 1/3 the required radius, or one foot, then the torque necessary from the elbow joint will be:

$$T = F * R$$

Or force times radius. The force of the hand and object mass is

$$F = m * g$$

Or approximately 2.61Kg * 9.8 m/s^2 (25.58 N). The radius, in meters, will be .3048m, so the torque on the elbow is 7.67Nm.

For the shoulder joint, one may assume that the elbow joint is solid. Therefore, the shoulder has to hold up to torque from the load at 1 meter and from the elbow at .6852m. If it is assumed that the motor for the elbow will weigh 0.6 kilograms, so the force at 0.5 meters will be 5.88 Newtons. Based on all this, the total torque on the shoulder joint will be approximately 4.02+25.58N which is about 29.60Nm.

There is one potential issue worth mentioning here. Most electric motors that are used for precision control do not operate in this torque range. To produce torques at this level, a motor must have substantial mass and spin at a high velocity. Such motors are not intended for control, but for high power spinning applications, such as driving a washing machine.

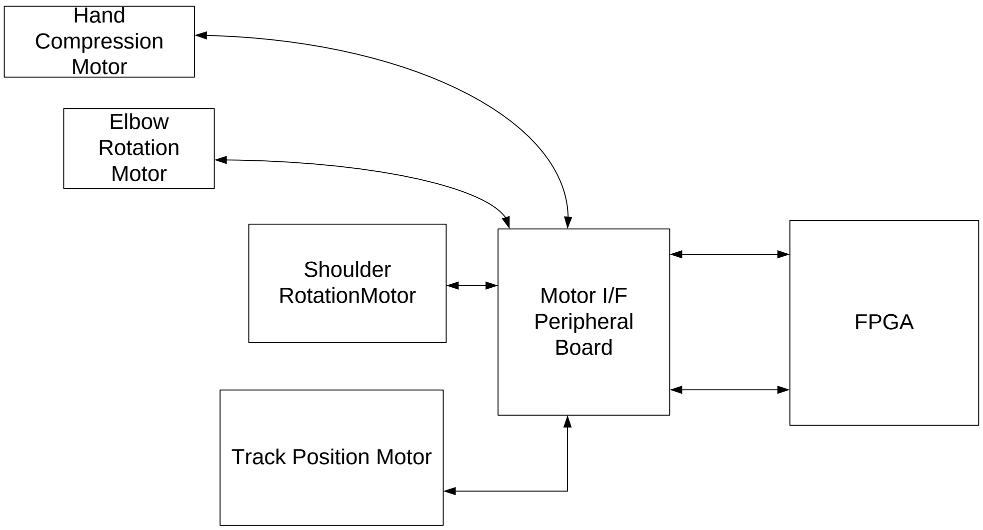

Figure 7: "Human" Arm Block Diagram

7.1.6.2 Option B

Figure 8: Elevator Arm Structure

Due to the potential high torque demands on the traditional arm structure, it is necessary to have a secondary arrangement planned and ready to implement if the first arrangement does not provide enough strength. The second structure will implement more of a crane or forklift approach, with an extendable arm being lifted straight up by a hydraulic or electric solenoid, then being pushed forward on a rack and pinion system, with the hand at the end of the unit moving forward. To help visualize this, see Figure 8 to the left.

The base of the arm will rotate the same as in the original plan. The unit parallel to the ground will be able to travel up and down on the vertical post, then forward and backwards to reach for the device. This will allow much greater mechanical strength to handle the torque and moment arms, as opposed to requiring electric motors to handle all of those forces. It will potentially help keep costs down, as well as implementation time. The range of motion while travelling down the track for this design will be greater, so there are more potential safety issues that will need to be addressed. The scale here can be pictured by assuming the base unit is the size of a push driven lawn mower. The device should not be much higher than height one can infer from this scale to ensure it does not become a danger to employees.

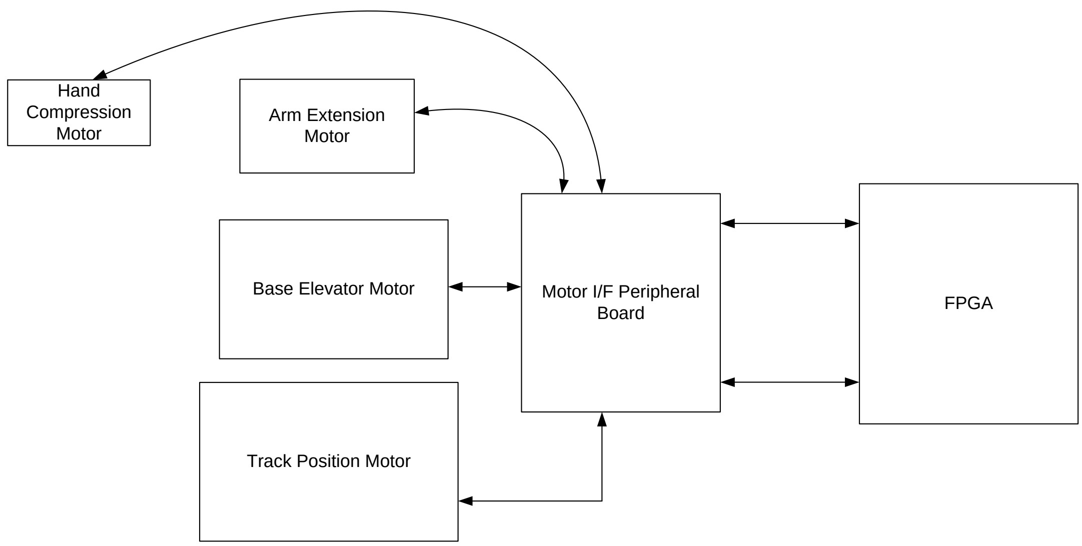

Figure 9: Elevator Arm Block Diagram

7.1.7 Hand Construction

We plant implement the gripper hand for the robot to be able to lift one to fifteen pounds, while still allowing for potential force feedback as well as optical sensing. Of course, the amount of weight that can be lifted will depend on the static friction between the hand and unit that was picked up. We are also currently researching the advantages of a pneumatic gripper hand. A simple electronically controlled solenoid will be used to open and close the gripper. An Autodesk Inventor 2008 drawing of our current hand design has been modeled after a sample gripper, pictured in Figure 10 below.

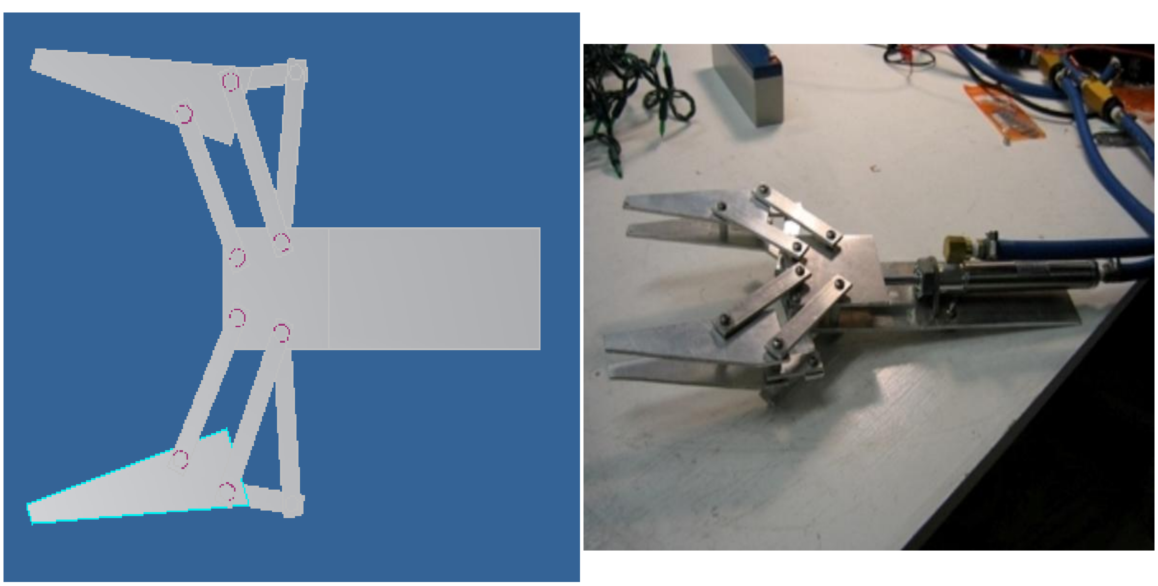

Figure 10: Hand Gripper Design

7.1.8 Track Construction

The track for this device will consist of two parallel pipes or C-shaped beams. The purpose of the track is to provide the range of travel for the entire device, as well as a source of stability for the moments that will be created while the arm unit is reaching out and picking up loads. The track should be long enough to cover exactly one width of shelving, and it will not incorporate any bends or turns; the plan is for a one dimensional range of motion, along the face of a set of shelves. The demonstration model that the group will produce will have around 10 to 15 feet of track across which it can travel, with assigned stopping intervals along the way. Those intervals will be determined by a travel timer, along with a motor encoder. The motor that propels the cart down the track will have a tachometer that keeps track of revolutions, and every stopping position will have an assigned revolution value that must be achieved to have successfully reached that point. This will be backed up by optical sensors that scan for either a barcode naming the shelf, or for the object to be picked up, helping to refining alignment.

7.1.9 Proximity Sensors

In order to ensure safe and reliable operation, this device must be equipped with the necessary proximity sensors to keep it from running into obstructions while moving down the track, and while spinning the arm around. It may be possible, through the use of IR reflectors, to also use the proximity sensors as track position sensors, so they double as a device that tells the core when to stop the track movement.

There will need to be a proximity sensor for the forward and back positions for track movement, and then for 360 degree rotation for the arm. There will also need to be a sensor within the grasp reach of the hand, and this can double as object detection, to determine whether or not there is an object within its reach.

7.1.10 Connections

The device will run from an AC power supply hooked up to mains, so it shall be equipped with any necessary internal power supplies to convert voltage and current levels. It will also be connected to the host computer where it receives the commands. Therefore, there will need to be at least two wire bundles connected to the mobile unit. This will be done by use of a rolling wire protected within a cable drag chain, as shown here.

Figure 11: Cable Drag Chain

For the purpose of product planning and business development, it is reasonable to expect that the final product be controlled wirelessly, with power coming from a brush contact with a power source along the track, but that is outside the immediate scope of this project.

7.2 Software Design

The goals of the project impose several design requirements on system software. The system configuration requires a complete intelligent system controller, some device drivers, GUI design, a small database, and the design of interaction between these components.

7.2.1 Controller Alternative Solutions

We are considering three alternatives for the arm's intelligent controller:

1) Executive System
   - In an executive system, a simple assembly language program called an executive would interact directly with registers provided in hardware.
2) Operating System
   - If we choose to use an operating system, applications running on the operating system would control hardware through device drivers.
3) Real-Time Operating System
   - A real-time operating system would provide control of both the logical and temporal results of the system's operation.

A comparison of the advantages and disadvantages of each option is shown in Table 5.

Table 5: Advantages and Disadvantages of Controller Alternatives

	Executive	**Operating System**	**Real-Time Operating System**
Advantages	Simple, easy to implement.	Medium complexity. Provides cleaner separation of processes.	Provides real-time capabilities.
Disadvantages	No real-time specifications, problem would need to be solved completely in hardware and assembly.	No real-time specifications, large memory requirements.	May be overkill for this project. Complex implementation. Large memory requirements.

Because of the complexity of the Real-Time Operating and System and the limited benefits it may provide, we are currently planning on using an executive or non-real-time operating system for the intelligent controller. If further testing and research shows that a real-time operating system can be more easily implemented on an FPGA, this option will be pursued.

7.2.2 RTOS

For the arm's parts to respond in sync to the demands of a physical environment, we are considering deploying an RTOS. An RTOS will also be beneficial for being able to boast short response times in general (in a situation where speed is important).

7.2.2.1 RTOS Requirements

7.2.2.1.1 Timing Constraints

The real time system the team requires would be considered “firm” because failure to respond to events in real time will cause failures in the operation of motors (and picking up objects) but a low probability of failure to do these operations can be tolerated [9].

7.2.2.1.2 Deterministic Behavior

Because of the timing constraint specifications, the system does not need to be completely deterministic (though knowledge of the majority of relevant latencies will be required for operation).

7.2.2.1.3 Interrupt Priority

It would be useful to have a system with multiple priority levels, and since this is likely to be the case with most available RTOSs, we will be taking advantage of this feature. It is expected that problems related to priority levels (priority inversions, starvation) will not be dealt with directly by we during design [10].

7.2.2.1.4 RTOS implementation

We are not particularly concerned about using a kernel that is natively RTOS versus one that has an extension to make it appear so. Although a modified kernel is more likely to have soft real-time responses, a low probability of failure can be tolerated [11].

7.2.2.2 Programming

To properly take advantage of an RTOS, we will be doing basic real-time programming. The current plan is to write real-time code in Ada or Java with real-time extensions (Sun Java Real-Time System or Java RTS) [12].

7.2.2.3 Real-Tim e Operating System Alternative Solutions

The decision for the RTOS included technical and practical concerns. From the decision matrix (see Matrix Table 6), we made the determination that eCos was the best option for the immediate implementation of the arm. As further considerations in the design arise, we will continue to monitor the feasibility of an RTLinux port to the LEON processor.

Table 6: Real-Time Operating System Decision Matrix

	Weight: 0-1	Score: 1-10 RTOS Options			
Feature	**Weight**	**RTLinux**	**eCos**	**VxWorks**	**rtems**
1. Integration with LEON	0.5	2	8	2	4
2. Cost	1	10	10	0	10
3. Multiple Priority Levels	0.5	10	10	10	10
4. Support	0.8	6	3	8	4
	Totals:	20.8	21.4	12.4	20.2

7.2.3 Device Drivers

The system will require several device drivers if an operating system is used. Device drivers will be needed for the different parts of the arm, the arm's hand, the wheels connected to the track, and any associated sensors.

Currently, the plan for these device drivers is to have signals come in and out from the mechanical arm as voltages on available pins. These voltages will be read and interpreted in VHDL and presented to the processor as a series of control and data registers. The function of all registers used will be documented, and we will create some C libraries for higher level control.

7.2.4 GUI Design

The project should include a small graphical user interface (GUI) to allow users to order the arm to perform certain functions, or at the highest level, order it to get a part from available options. The GUI should present to a user a list of parts and allow the user to search for parts in the database. If the user asks the arm to get a part, the arm should go to the "warehouse" (a small collection of items for demonstration) and bring back the desired item. If the arm is implemented using a RTOS, the time it takes for the arm to obtain the item and come back to the computer will be approximately known.

The two options for implementing the GUI are Java and C++. The advantage of C++ will be simpler integration with the operating system environment, but a GUI is generally harder to implement in C++. A GUI in Java will require a Java Virtual Machine (Micro Edition) in the operating system, but will be simpler to write.

7.2.5 Database design

We are planning on using a database to keep track of the parts available to the user. This may be a very small database in the system or one accessed through a TCP/IP connection, such as MySQL. The complexity of the database will be dependent on how much time can be put into this aspect of the project.

7.2.6 GUI/database interaction

If the GUI is created in Java, the Java Database Connectivity API will be used to connect to a database available only through TCP/IP. A local Java database will be relatively simple to manage. A local or remote database in C++ gives access to lower levels of hardware and so will in some ways be simpler to manage than Java implementations.

7.2.7 Software Configuration

The overall software configuration is shown in Figure 12. The dashed entries represent optional add-ons to the project.

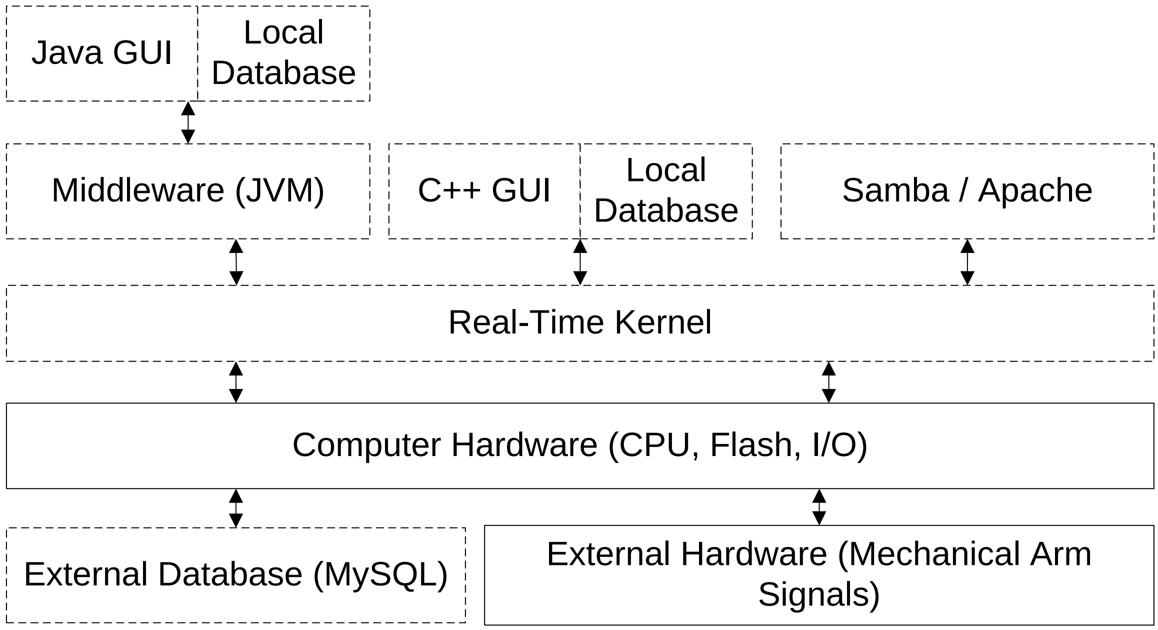

Figure 12: High-level Software Interaction

7.2.8 Software Tools

For managing hardware and software, programs for many different functions will be used. A short list of software programs this project has and will be taking advantage of are listed below.

1. eCos Configuration Tool
   - Used to create customized eCos kernels
2. Eclipse
   - Integrated Development Environment for C, C++, and Java programs.
3. Samba
   - Free software to provide file and print services to various Microsoft Windows clients
4. Apache
   - Open-source web server

7.3 System Design

The overall system, as shown in the rest of this document, involves the incorporation of several complex elements to create one fully functional, useful unit. The software user interface system will have to interact with the hardware of the robot motor control to make sure things run smoothly. However, if the device is indeed to have some level of autonomy, as desired, then the commands required from the software system should be sparse enough so that it can be a sort of “touch and go” interface, where the software system sends a single, short command, then lets the hardware unit run off and perform the task all on its own, with minimal monitoring required.

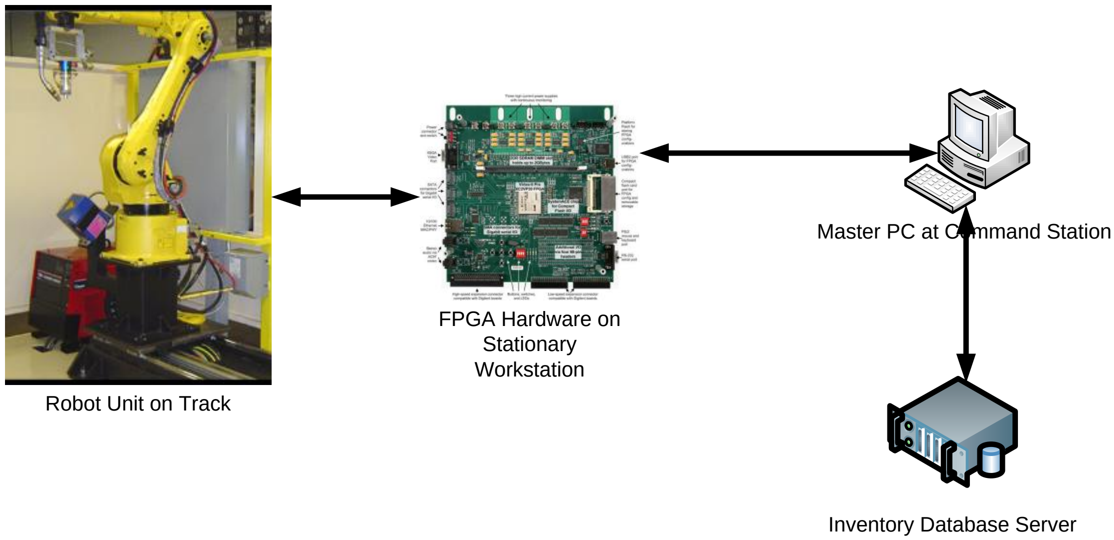

Figure 13: System Block Diagram

All of the components described so far will work together in one fully functional unit.

Any general task will follow this pattern:

(1) The user/operator will enter a command in the GUI for which stocked object they desire.
(2) The Software will check with the database to see if it is in stock, and if so, where.
(3) The software device driver then passes a simple command to the OS on the expansion board.
(4) The OS takes over from there, interpreting the command by doing the following:
    a. Ensuring device arm is in moving position
    b. Beginning movement down track, checking tick marks along the way
    c. Stopping at proper location, locking base upon arrival
    d. Arms swings into position
    e. Hand grabs object, retracts to moving position
    f. Base unlocks, device travels back to home position
(5) After the User removes the object, the arm OS notifies the Software driver that is is no longer busy and available for the next command

All of this will be completed with enough layers of protection to ensure the user cannot disrupt the process by requesting another product while the arm is in motion, and so that the device will not run into anything while moving on the track or swinging the arm.

For the purpose of a thorough demonstration, the team plans to unplug the host PC and mobile device (after one command is sent). This will demonstrate that the device is running under its own control and "brain" power, and not being controlled by the host PC. This is the desired level of automation that should be achieved, as mentioned in the Requirements for Success section.

7.3.1 Product Specifications

7.3.1.1 Product Weight

The weight of the complete product will depend on the amount of track used in the custom system. However, the weight of a typical robotic arm mounted on the track (the amount that must be moved) is estimated in Table 7 below.

Table 7: Robotic Arm Weight Estimation

Component	Quantity	Weight (lbs.)
Wheels	4	0.5
Motors	3	1
Base	1	3
Elevator Box	1	3
Hand	1	0.8
Forearm	1	1
	Total:	12.8

The track is expected to weigh about 1 lb. per foot, but will most likely be delivered by employees of the company (so there is no charge for shipping).

7.3.1.2 Product Power Consumption

The power consumed is projected to be under 260W, as shown in Table 8.

Table 8: Robotic Arm Power Consumption

Component	Quantity	Power (W)
Stepper Motors	3	20
DC Motor	1	100
Pneumatics	1	100
	Total:	260

If the team requires greater power consumption, a listing of additional power consuming units will be provided.

7.3.1.3 Operating Environment

One goal is to have the object be able to operate within the standard temperature range of most electronic elements, for this project -40 to +55 degrees Celsius. This will also make the device capable of working in some environments that are less than ideal for human involvement, which is a new market possibility as well.

8 Project Task List and Time Allocation

A complete breakdown of the tasks the team must complete by the end of the project is included in Appendix B: Task Breakdown. A shortened version of this table is provided for convenience in Table 9 below.

Table 9: Work Breakdown Summary

Task Name	Estimated Work	Completed Work
Team 5 Senior Design Project	826 hrs	223 hrs
Project Management	89 hrs	22 hrs
Research	90 hrs	70 hrs
Feasibility Study	159 hrs	88 hrs
Implementation	389 hrs	0 hrs
PPFS	44 hrs	43 hrs
Prepare Final Presentation	23 hrs	0 hrs
Prepare Final Report	32 hrs	0 hrs

The schedule shows that the team has put in approximately 223 hrs. of work into the project, which corresponds to the amount of work documented in other timekeeping records.

9 Research

Several different methods for research are being explored by the team. Since two of our team members are currently carrying out internships at DornerWorks, regular status meetings are held there. We are also making sure to use on-campus resources to their fullest extent.

9.1 Meeting with Professionals

To help guide the project from conception to completion, regular meetings have been performed with an industrial mentor. Calvin Engineering Professor Steven VanderLeest, who is currently on Sabbatical at DornerWorks, Ltd, is the team's industrial mentor. Before the semester began, we began formulating project ideas and sent in our resumes to Professor VanderLeest in order to be considered for an industrial sponsorship. Professor VanderLeest also suggested several project ideas, while still allowing the group to develop ideas they would be interested in. Ultimately, it was Professor VanderLeest that suggested pursuing the work of putting an RTOS on an FPGA processor, and he also provided some suggested guidelines to work within. The idea to use the RTOS to control some sort of automated robot grew out of that, and the work has since drifted away from the RTOS aspect and more towards the robotics control side of work.

Regular meetings are still being maintained with Professor VanderLeest and another DornerWorks employee, Nicholas Goote. Nick is working on a master's project at Grand Valley State University, and hopes to use the research that comes out of this design project to spring board into a better master's project. Every three weeks, a status update meeting is held at DornerWorks between Steve, Nick, and

David VanKampen. Steve has been a Senior Design professor for several years, so he offers good guidance and advice to ensure the project is successful. Nick graduated from Calvin College in 2006, so he recently completed the senior design course and can offer good insight into what it takes to be a successful senior design student.

We have also been set up with an industrial consultant at GE Aviation. One meeting with our consultant, Gregory Barber, was held on November 20, 2008. Further meetings are planned for our next semester.

9.2 Literature Research

We have been involved with many sources for literature research. These sources include meeting with head of Heckman Library Research Glenn Remelts, web research, and journal articles. The Gaisler Research web site has been especially helpful with obtaining written information. We have used manuals on the LEON3 Processor, ECOS operation and compilation, SnapGear Linux, the XUP Virtex II Pro FPGA, and the XUP Virtex-5 FPGA. In addition to the user manuals we have used books on Linux Device Drivers, Sun SPARC architecture, and Understanding the Linux Kernel. We have also found and continue to find valuable journal and magazine articles, listed in our bibliography. We expect to continue to use the Heckman Library Search Database and also the internet, especially the Gaisler Research Website.

9.3 Similar Products

There are few products available in exactly the same market that we are targeting. There is one company that does have a promising product already available, Kiva Systems. Kiva Systems makes a product that picks up an entire shelf and brings it to a human packer to pull an item off the shelf and place it into a box to be mailed. Many (50+) of these robots can work in conjunction with each other, but it does have several downfalls. Aside from Kiva Systems there are many companies that offer large scale warehouse automation, but these companies are not targeting the same markets we are and therefore are not direct competition for our product. To the best of our knowledge, this is the first product of its kind!

10 Project Management

10.1 Work Division

The project has been divided into teams for each work category.

10.1.1 FPGA and Processor Core Design

David VanKampen and David VandeBunte will head up the work on the FPGA and CPU. They will implement an open core processor, as well as make any additions or modifications necessary to it. This involves compiling code, making sure there are no licensing issues, and writing whatever new code is necessary for the motor driver units.

10.1.2 Software Programming

Dave VanKampen and Dave VandeBunte will also be in charge of any software programming necessary. This involves writing or using code for either a real time operating system, or simply an executive, as well as overseeing the GUI development. This also covers the writing of any software device drivers that may be used, if hardware control is not sufficient.

10.1.3 Peripheral Electronics

Ryan Mejeur and Matt Lubbers will oversee the peripheral board development. This covers designing the amplification circuitry necessary for any control signals coming from the FPGA, as well as directing power from the AC supply to all the motors. It will also entail some PCB layout and fabrication, as well as the population of all needed parts.

10.1.4 Motors

Ryan Mejeur and Matt Lubbers will also head up the motor development process. Regardless of the final motor type being used, Ryan and Matt will have to make clear the type of communication necessary to control the motors, since these are the signals that will be produced by the hardware and software designed by David and David.

10.1.5 Robotic Arm

Ryan and Matt will also be in charge of the development of the arm and hand design. This will be where the majority of their time will likely be spent. They will incorporate the motors into the mechanical construction of the arm and base, oversee production and testing of the prototype, and make sure everything is ready to be passed on to the David VandeBunte and David VanKampen for specific controller integration.

10.2 Team Structure Diagrams

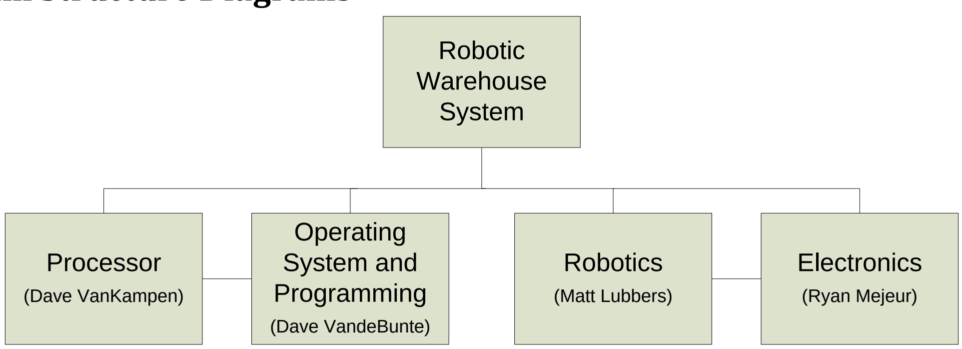

Figure 14: Work Division

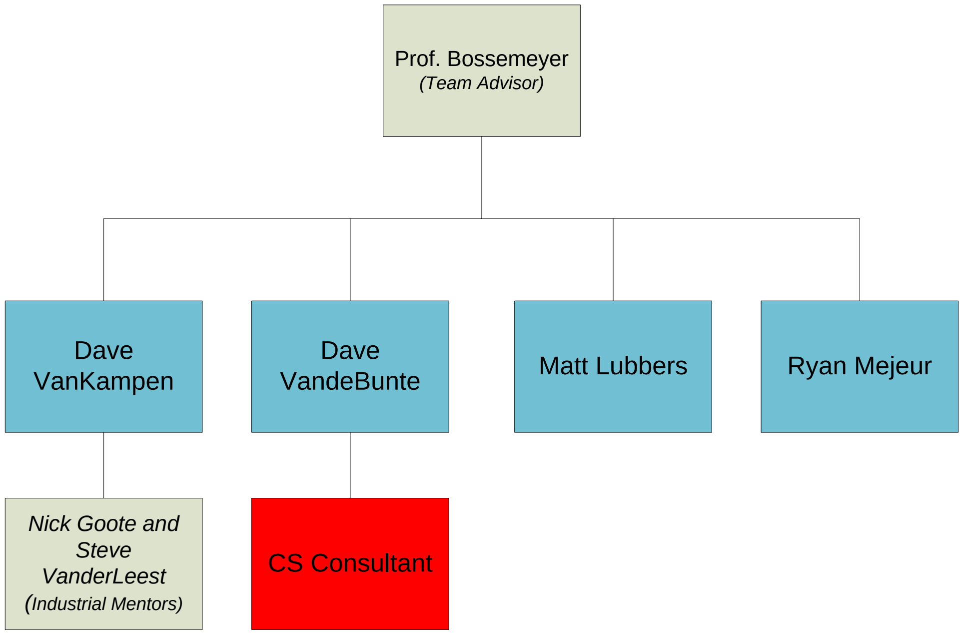

Figure 15: Organizational Chart

10.3 Conflict Resolution and Delegation

We will delegate tasks in a fair manner that uses the strengths of each team member. When someone does not complete their task by the given deadline, we are planning on having a team meeting to discuss what changes need to be made to get back on track, as well as what changes need to be made in the individuals' role. If a technical disagreement arises, we will do more research to get to the bottom of the matter. If a design decision is required, then the entire team will need to be involved in that decision.

Delegation is performed based upon previous experience coupled with preference. If someone has internship experience related to a topic, the task will likely be assigned to them. However, if a person really does or does not want to perform a specific task, reassignment will be done to insure that everyone is aware of the work that is expected of them.

10.4 Conditions for Success

We have decided that the Senior Design Project will be a scalable to allow for a project that can be completed in stages. The design team has set out the below conditions for success:

1. The team will have embedded a real-time processor in the FPGA with a real-time operating system running on top of the embedded processor.
2. Completion of a robotic arm that is driven by its servos and motors.
3. Integration of the robotic arm with the FPGA, running off of the real-time operating system.
4. Development of a database program that operates in conjunction with the robot and controls the robot.
5. Completion of a demonstration with 10 or more items.

11 Business Plan (see also "Formal Business Plan")

Each year, billions of dollars are spent on stocking and distributing goods in the warehousing and storage industry. With the continual growth in the industry, warehouses are striving to become more and more efficient, by making their processes more efficient and cost effective. Our business strategy is to capitalize on this industry, making it even more efficient and cost effective by using robotics as a new warehouse distribution system.

We intend to provide custom and predesigned robotic equipment to warehouses for storing and fetching goods electronically via a computer system. By using electronic equipment and computers to control warehouse distribution, companies will have more control of their goods in storage, which will be very important for perishable goods in warehouse storage. Millions of dollars are lost each year due to goods that are unaccounted for, or goods that perish while in the warehouse. A robotics solution will allow warehousing companies to not only reduce the number of lost or perished items, but also reduce the staffing needs and labor intensity of the warehouse.

A typical warehouse requires anywhere from 10 to 30 fulltime people to operate effectively. According to the U.S. Labor office, the average warehouse worker makes around $15 an hour [27]. We anticipate a 25% reduction in personnel required to operate a warehouse effectively and efficiently. A robotic warehouse implementation could produce a cost savings of anywhere from $10,000 to $1.2 million in yearly labor costs, and even more through liability insurance, employee benefits, and other additional labor expenses. It is not our intent to eliminate work completely, but rather create new and more skilled positions in robotics.

In addition to implementation and design of robotic warehouse applications, we will create a consulting and technical service to assist warehouse management in making good business decisions. The technical aspect of the service will support the robotic applications installed, while the consulting aspect of the service will provide efficiency concepts and automation advice for warehouse owners and management. The service division of the business will add jobs to the economy while also providing the company with a steady stream of post sale income. Additionally, we also plan to sell replacement parts, maintenance parts, and commissioning and qualification documents. It is expected that the services department will be a large source of revenue.

11.1 Marketing

There are currently many manufacturers of large palletizing warehouse robots; however these are not our current competitors. Our business plan is to create a small and very inexpensive robot capable of retrieving small objects. We believe that if we can create an automatic storage retrieval robot for $1500 to $2000 that it would open an entirely new market sector in automated warehouse retrieval systems. These systems could then easily earn back their initial investment, making them very appealing to warehouse or even factory applications.

11.2 Strategies

We believe that by starting as a small business which develops specialized solutions for individual clients, our company can begin with a relatively low overhead, concentrate on customer service, and develop great relationships with our customers which will lead to additional business opportunities.

11.3 Managerial and technical experience

Another advantage of starting off as a small company is that we do not need a large number of employees. The core of the company consists of four electrical engineers with a variety of experience and backgrounds. We believe that we will be able to effectively manage the technological research and development of the product, as well as the technical support for our customers. We also believe we can manage the finances of the company at initial start up, but we would like to hire an additional employee to handle the finances when the company begins to grow and it becomes economically feasible for the company.

11.4 Financial request and cash flow

We have already begun the research and development sector of our business. We have purchased an electronic development board and are currently in the process of installing a real-time operating system onto it. This will act as the robots intelligence and control the movement of the arms and its location on the track. This will allow it to receive information about the desired item and its location, along with monitoring for any obstacles along its path. We are also currently investigating the mechanical components of the robotic arm such as electrical servos, arm material types, and a claw type. These decisions will be based on our evaluation of the best balance between object dimensions, warehouse requirements, and final production cost of the robot.

Our current cash flow will be used to finish an operating prototype, along with continual research and investigation of possible applications. Once the prototype is completed, additional budgeting for further add-on features to tailor to specific applications will be needed.

12 Parts and Project costs

As mentioned previously, our goal is to develop a marketable robot for a total price of $1500 to $2000. We hope to develop the current prototype for $1300, which we received funding through both Calvin and DornerWorks. The team has also recently received $200 through the Calvin Bizplan competition thanks to the KEEN Grant. The project cost breakdown analysis is very difficult at this time because we are still exploring several different methods of building our prototype and because we are deciding which specific parts will be the best. The approximate part prices have been obtained from Digi-key.com

12.1 Track System

We are currently planning to use one or two DC motors to power the wheels which move the robot along the track. This will enable the robot to travel to a specific spot on the track directly in front of the desired object. A DC motor controller is then necessary to handle the large voltage and current required to power the DC motors as well as to control when the DC motor turns on and off.

Another design possibility for a “track system” would be to use a black line painted on the floor. The robot would then have infrared sensors installed to follow this line, and this would eliminate the necessity for a physical track. By eliminating a track we would eliminate the tripping hazard imposed by a 12” track raised off the floor. Also, this could greatly reduce our cost for implementing our system in the warehouse. However, the problem with eliminating the track is that the robot would lose stability. When the robot extends its arm and picks up an object, it could have the possibility of tipping over if the item is too heavy.

We have also considered using a self-propelled push lawn mower as the base of our robot. The lawn mower base would provide a perfect sized base, and we could modify the electronic driven wheels to run off of the power supply. This design alternative could further reduce the cost of our prototype. The estimated parts cost for the prototype and production costs are shown in the table below.

Table 10: Track system costs

Track system	Prototype	Production
Wheels	$15.00	$10.00
Axles and Hardware	$10.00	$5.00
DC motor	$100.00	$75.00
Base Material	$20.00	$15.00
Track (10' - 15')	$100.00	$500.00

12.2 Arm Motors and Material

An important decision which will have a large impact on the product cost is the arm delivery type. We are currently deciding whether to use a “human based” delivery arm, or an elevator arm with a rack and pinion. These two delivery arm types have been discussed in preliminary hardware design, but here we will investigate the pricing of the two options.

Because of the difference in torque produced in the two methods, the delivery type decision will effect which motors we must use for the arm. In both delivery types, we plan to use a DC motor to rotate the base of the robot. If we decide to implement the human based delivery arm, we will need to use industrial strength servo motors with large torque capabilities to control the various joints of the arm. The estimated parts cost analysis for the human based delivery arm for prototype and production is shown below.

Table 11: Human Based Delivery Arm Costs

Human Based Delivery Arm		
Arm Motors and Material	**Prototype**	**Production**
DC Motor for Rotating Base	$100.00	$75.00
Shoulder Servo Motor	$200.00	$150.00
Elbow Servo Motor	$100.00	$75.00
Wrist Servo Motor	$50.00	$25.00
Arm Material	$100.00	$50.00

If we instead chose to use the elevator arm with rack and pinion, we will be able to use DC motors to both elevate and extend the arm to reach the desired part. The torque requirements for this type of arm will be significantly lower, and therefore the cost of the DC motor will be less. The estimated parts cost analysis for the elevator arm with rack and pinion for prototype and production is shown below.

Table 12: Elevator with Rack and Pinion Delivery Arm Costs

Elevator with Rack and Pinion Delivery Arm		
Arm Motors and Material	**Prototype**	**Production**
DC Motor for Rotating Base	$50.00	$30.00
DC Motors For Elevator Screws	$100.00	$75.00
DC Motor for Rack and Pinion	$100.00	$75.00
Rack and Pinion Material	$100.00	$50.00
Elevator Housing Material	$50.00	$30.00

12.3 Hand

Once the arm has been moved in front of the desired item, the gripper will extend to touch the item. The gripper will be instrumented with a simple pressure sensor, which will insure that the gripper hand is touching the item before it closes. Once the pressure sensor indicates that the gripper is touching the item, the gripper solenoid will be activated, and it will close around the desired object. The estimated cost analysis for the hand gripper and solenoid is shown in the figure below.

Table 13: Mechanical Hand Costs

Hand	Prototype	Production
Hand Gripper	$30.00	$15.00
Gripper Pneumatics	$50.00	$30.00

12.4 Controllers

We began our project by purchasing a Xilinx XUP development FPGA to control the movement of the robot and its arm. We simultaneously entered a competition to win a second FPGA. We were not expecting to be awarded the second board, but we thought it wouldn't hurt to try. We were awarded the board, and have already begun to load the soft core and operating system onto it. Along with the two FPGA's, we expect to purchase a DC motor controller to control the operation of the DC motors. It should be noted that while we are using a development board for our prototype, we do not believe it necessary to need this level of "computing power" for future production systems. For this project we determined the best approach was to initially purchase a quality development board with plenty of computing power in case it was needed later. After our prototype robot system is complete, we expect the cost of the necessary controller to be significantly lower. The estimated cost analysis of the controllers for our prototype and production models can be seen in the figure below.

Table 14: Costs associated with Controller

<table>
<tr><th>Controllers</th><th>Prototype</th><th>Production</th></tr>
<tr><td>XUP Virtex-II Pro Dev. System:</td><td>$356.90</td><td rowspan="2">$200.00</td></tr>
<tr><td>XUP Virtex - V ML509</td><td>FREE</td></tr>
<tr><td>DC Motor H-bridge Components</td><td>$60.00</td><td>$30.00</td></tr>
</table>

12.5 Object Detection Sensors

Since all of the robots movements will be performed independently of a human operator, safety issues are very important. To insure that the robot does not move down the track while someone or something is in its path, we will implement object detection sensors on the robot. We will use three different types of sensors to detect objects at different distances from the robots. We will use a long distance infrared sensor to detect objects from 40" to 216", a mid range infrared sensor to detect objects in the range of 4" to 31", and a pressure sensor which can detect if an object is touching the surface. The close range sensor will be placed on the gripper to insure that it is touching the object to be picked up. Also, in case of any malfunctions, we will implement an "Emergency Kill Switch" to cut all power to the motors and servos. This is necessary to insure the safety of all those working near the robot, and should be easily accessible and clearly marked on the robot. The estimated cost analysis of the sensors for the prototype and production models are shown in the figure below.

Table 15: Sensor Costs

Sensors	Prototype	Production
Mid Range Infrared Sensor	$8.99	$5.00
Long Range Infrared Sensor	$18.50	$10.00
Pressure Switch	$12.00	$6.00
Emergency Switch	$40.00	$20.00

12.6 Wiring and Hardware

As of right now, the wiring and hardware required to make our robot operational is unknown. We believe we will need several peripheral boards, including one to control the movement of the servos or DC motors, and another board to link the separate object detection sensors to the controller. The estimated cost analysis of the wiring and hardware for the prototype and production models are shown in the figure below.

Table 16: Wiring and Hardware Costs

Wiring and Hardware	Prototype	Production
Wiring and Peripheral Boards	$75.00	$25.00

12.7 Total Costs

As expected, our estimated prototype and production costs are very close to the $1500 price we were hoping for. Obviously there will be fluctuation in the actual price paid for each purchased part, as well as additional parts that may need to be purchased, however this is our best attempt of estimating the cost of the project. The planned production model costs are based on an assumption of 10 units sold per year, and therefore we expect the price of parts to be less expensive with a larger quantity of parts being purchased. While the planned production model will be similar to the prototype, we plan to customize an automated warehouse solution for our customer, and therefore the cost may differ from our predictions. The total prototype and production costs for both delivery arm types are shown in the figure below.

Table 17: Total Cost for Rack and Pinion

Elevator with Rack and Pinion	Prototype	Production
TOTAL	**$1,296.89**	**$1,206.00**

Table 18: Total Cost for Human Based Delivery Arm

Human Based Delivery Arm	Prototype	Production
TOTAL	**$1,503.89**	**$1,377.00**

13 Alternative Solutions

To summarize all of the options discussed above, here we will outline the several alternatives that have been investigated, and reiterate why certain decisions were made. The rational for some choices is outlined completely in the Preliminary Feasibility Analysis section below.

13.1 RTOS Choices

As summarized before, the team believes an RTOS may be overkill for this particular project. Although eCos and similar options will still be pursued, they will likely be billed as “background tasks”, and the robotic controls systems will take precedence for both the software and hardware processor teams.

13.2 Hand Type

There are several directions the hand style could go. A claw that is controlled with a single pneumatic solenoid is the current best option, but electric servo control is still a possibility, and there are several options for pressure or optical sensing as well. The current plan involves pneumatic control, but this can easily incorporate the further sensors, so the development of those systems will move ahead in parallel.

13.3 Track Type

There are also two main alternatives in the track style area of the design. Having a physical track made of metal beams will provide stability, mechanical direction control, and help constrain the scope of the project. However, a painted stripe laid down along the projected path of travel will increase marketability and feasibility of real world use for the product. Therefore, both options will be pursued in parallel in this design as long as each remain feasible.

13.4 Arm Type

As outlined in the hardware design section, we have investigated two different arm types. Due to the weight requirements of the intended market, an arm with high mechanical strength and weight capabilities is optimal. Since the elevator arm (arm option B) will provide the greatest amount of load capacity, it will therefore be the best choice for our design. Also, due to its mechanical design, it will not require larger motors to carry the additional weight. Therefore this type of arm will allow the robot to carry heavier objects, without increasing the weight of the robot itself due to larger motors. An Autodesk Inventor drawing of our robot design is shown below in Figure 16.

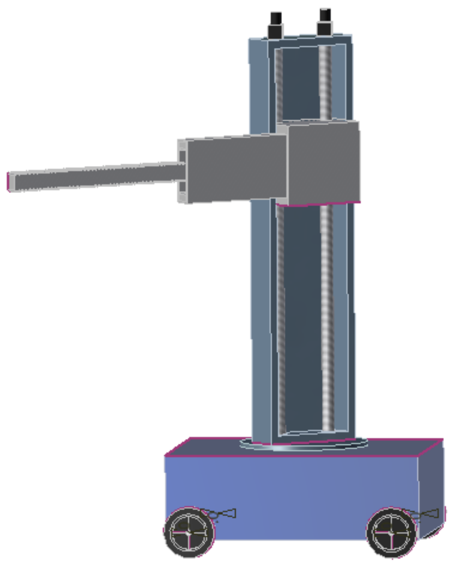

Figure 16: Robot Base

14 Preliminary Feasibility Analysis

Several hurdles stand in the way of us completing the project successfully and the proposed product moving successfully into the warehouse market. The most notable technical challenges are the RTOS, mechanical design problems, and FPGA and robotic arm interaction. The key non-technical issue this project must deal with is the product cost.

14.1 Operating System

Implementing a real-time operating system on an FPGA introduces many complications in the design process. A real time operating system requires a separate real time clock which must be monitored and used in real-time software. There have also been very few instances of real-time operating systems running on FPGAs because of the lack of open-source processor cores with corresponding open-source real-time operating systems.

The design could also be implemented with a non-real time operating system, but the robot arm would operate less reliably and less predictably. The advantage of a real-time system is that operations will be able to happen in synchrony, while with a typical processor and system the operations will happen as quickly as the processor receives orders.

If implementing an RTOS on an FPGA turns out to be involved, we can fall back on a normal operating system. The time it takes to perform operations will be approximated and timing will be tracked with other methods.

14.2 Mechanical Design

Important mechanical problems in the design are being investigated and will be key challenges in the future. The two technical challenges here are optimizing the mechanical arm for strength and implementing a successful mechanical hand.

14.2.1 Arm Strength

The overall design of the mechanical arm will greatly affect the weight that the arm can handle. A typical arm with multiple joints (see Figure 17) is only capable of carrying weights around five lbs. If the arm's joints are rigid (with mechanical parts holding them in place rather than servos) the amount of weight that can be transferred increases significantly (see Figure 8).

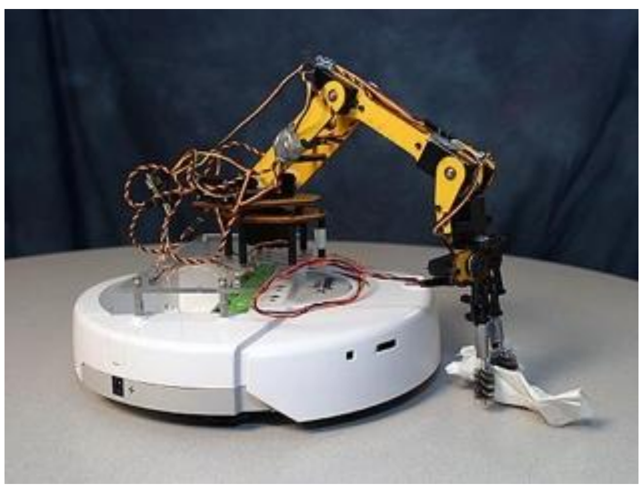

Figure 17: Example Robotic Arm

14.2.2 Mechanical Hand

Implementing a successful mechanical hand will also be a significant mechanical design problem we will undertake. The two design options being considered now are based on either servos or solenoids.

14.3 FPGA and Robotic Arm interaction

The current strategy for making the robotic arm for this project work out is having pins from the FPGA drive voltage levels (with amplification) on the arm. Working with the arm will require enough pins on the FPGA to be able to drive all of the arm's components and ensuring that signals are passed reliably from the FPGA to the arm.

14.4 Product Cost

The ultimate cost of the product will be an important factor in how it may move to the warehouse market. In the Business Plan section we have outlined the costs that are foreseeable at this point.

15 Project Requirements

The project's scope has been defined in terms of its goals in computer software, computer hardware, and electrical/mechanical areas.

1. Hardware Requirements
    - Obtain and compile an open-source processor core and implement it on an FPGA.
    - Construct working communication lines through serial, Ethernet, monitor, and optionally keyboard/mouse
    - Must implement exceptions and external interrupts
2. Software Requirements
    - Demonstrate file system application (for addition of new application software)
    - Demonstrate application as a device driver (servo or actuator)
    - (optional, time dependent) Demonstrate as a Web Server/File Server (Apache or Samba)
    - (optional, time dependent) Manage database of locations for stock of different items, grab them itself, rather than being fed
    - (optional, time dependent) Develop a Graphical User Interface (GUI) to control device operation (e.g. tell it I need a certain product at this certain location along the track)
    - (optional, time dependent) Teach arm to take object back to its place if wrong object was retrieved.
3. Electrical/Mechanical Requirements
    - Design arm servos and configuration
    - Develop control mechanisms and sensors for robot to do the following:
        - Move on track
        - Stop to avoid obstructions
        - Continue to assigned location
        - Stop at location
        - Deliver product with extending arm
        - Stop delivery to avoid obstructions
        - Retract arm upon successful delivery
        - Return to stock room/home area, still stopping to avoid obstructions
            - Product specifications
        - Maximum Weight: 5 pounds
        - Maximum Dimensions: 6" wide, 6" deep, 12" tall
        - Minimum reach: 3'
    - Track system on which arm is mounted
    - 110 volts, 60 Hz, and 20 amp

16 Acknowledgements

We would like to thank the following parties for their continued support and guidance throughout the duration of this class:

- Dr. Robert Bossemeyer, for filling the role of official team advisor.
- Dr. Steven VanderLeest, for serving as industrial mentor and sponsor liaison.
- Nicholas Goote, for providing technical support and advice, as well as project ideas.
- Gregory Barber, for serving as industrial consultant, and providing intelligent guidance as we developed our business plan.

17 Conclusion

Senior design team five is confident that the first three design goals will be achieved by May 2009: hardware and software operation on the FPGA, robot operation independent from Host PC control, and integration of the robotic arm and the FPGA. We have received funding from DornerWorks, Ltd, been admitted into the Xilinx-Sun University Program, received a development board through the Xilinx-Sun University Program, and purchased a development board through the Xilinx-Sun University Program. The next steps are to develop a preliminary design for the robotic arm and purchase the necessary parts for that design. Additionally, a PCB will need to be designed after the robotic arm is constructed and operational. Team five is confident that we will stay within the projected project budget and will have a functional prototype by the end of the spring 2009 senior design class.

Bibliography

[1] Floden, Jonas. "LEON Integrated Development Environment for Eclipse." Gaisler Research 1.3.1.1 (2008): 1-16.

[2] Floden, Jonas. "SnapGear Linux for LEON." Gaisler Research 1.36.0 (2007): 1-52.

[3] Yaghmour, Karim. Building Embedded LINUX Systems. Sebastopol, CA: O'Reilly & Associates, Inc, 2003.

[4] Paul, Richard P. SPARC Architecture, Assembly Language Programming, and C. 2nd ed. Upper Saddle River, NJ: Prentice Hall Inc, 2000.

[5] Uni-directional DC Motor Controllers. Ocean Controls. 10 Nov. 2008 <X:EngineeringTeamsTeam5websitesdc_motor_controllers.htm>.

[6] Robotic Gripper Sizing: The Science, Technology and Lore. 1996. ZAYTRAN, Inc. 3 Nov. 2008 <X:EngineeringTeamsTeam5websitesgripperinfo.htm>.

[7] Sharp IR Rangers Information . 10 Oct. 2008. Acroname Robotics. 3 Nov. 2008 <X:EngineeringTeamsTeam5websitesirsensors.html>.

[8] Robot Sensors. 2004. Summerour Robotics Corp. 4 Nov. 2008 <X:EngineeringTeamsTeam5websitessensorsandcontrollers.htm>.

[9] Baskiyar, S. "A survey of contemporary real-time operating systems." Informatica 29.2 (2005): 233-40.

[10] Fowler, Kim. "Part 2: Real-time operating systems (RTOS)." IEEE instrumentation & measurement magazine 5.2 (2002): 48-49.

[11] Stankovic, John A. "Real-Time Operating Systems." Real-Time Systems 28.2-3 (2004): 237-53.

[12] Shaw, Alan C. Real-Time Systems and Software. New York: John Wiley, 2001.

[13] Wikipedia. 2 Dec. 2008. 2 Dec. 2008 <www.wikipedia.org>.

[14] Gaisler, Jiri. Aeroflex Gaisler. 2008. Aeroflex. 2 Dec. 2008 <http://www.gaisler.com/cms/>.

[15] Gaisler, Jiri. GRLIB Manual. 2008. 2 Dec. 2008 <http://www.gaisler.com/products/grlib/grlib.pdf>.

[16] PowerPC Licensing. IBM. 2 Dec. 2008 <http://www-03.ibm.com/technology/power/licensing/index.html>.

[17] OpenSPARC UInivserity Program. 8 July 2008. Sun Microsystems. 2 Dec. 2008 <http://www.opensparc.net/edu/university-program.html>.

[18] XUP V2P Development Board. Digilent, Inc. 2 Dec. 2008 <http://digilentinc.com/Products/Detail.cfm?NavTop=2&NavSub=453&Prod=XUPV2P>.

[19] ISE Design Suite 10.1. Xilinx. 2 Dec. 2008 <http://www.xilinx.com/products/design_resources/design_tool/>.

[20] Ermer, Gayle and Vanderleest, Steven H. Using Design Norms to Teach Engineering. June 2002. Calvin College. 2 Dec. 2008 <http://www.calvin.edu/academic/engineering/faculty/svleest/abstracts/asee02.htm>.

[21] Fata Automation. Fata Automation. 2 Dec. 2008 <http://www.fatainc.com/>.

[22] Automated Storage & Retrieval System. Westfalia USA. 2 Dec. 2008 <http://westfaliausa.com/>.

[23] Kiva Systems. Kiva Systems. 2 Dec. 2008 <http://www.kivasystems.com/>.

[24] The GNU General Public License. 29 June 2007. GNU. 2 Dec. 2008 <http://www.gnu.org/copyleft/gpl.html>.

[25] LEON_SPARC. Ed. Jiri Gaisler. Yahoo. 2 Dec. 2008 <http://tech.groups.yahoo.com/group/leon_sparc/?v=1&t=search&ch=web&pub=groups&sec=group&slk=3>.

[26] Bureau of Economic Analysis. The U.S. Government. 2 Dec. 2008 <http://www.bea.gov/>.

[27] U.S. Department of Labor. The U.S. Government. 2 Dec. 2008 <http://www.dol.gov/dol/topic/wages/index.htm>.

Appendix A: Motor Considerations

Appendix Table 1: Track Motor Type Decision Matrix

			Motor Types		
Criteria	**Weight**	**DC Motors**	**Stepper Motors**	**Servo Motors**	**Linear Actuators**
Precision	3	2	6	9	8
Power	9	8	7	3	2
Cost	4	4	3	6	8
Size	8	8	6	8	6
Total		158	141	142	122

Appendix Table 2: Horizontal Motor Type Decision Matrix

			Motor Types		
Criteria	**Weight**	**DC Motors**	**Stepper Motors**	**Servo Motors**	**Linear Actuators**
Precision	8	2	6	9	8
Power	6	8	7	3	2
Cost	4	4	3	6	8
Size	5	8	6	8	6
Total		120	132	154	138

Appendix Table 3: Arm Rotation Motor Type Decision Matrix

			Motor Types		
Criteria	**Weight**	**DC Motors**	**Stepper Motors**	**Servo Motors**	**Linear Actuators**
Precision	5	2	6	9	8
Power	8	8	7	3	2
Cost	2	4	3	6	8
Size	9	8	6	8	6
Total		154	146	153	126

Appendix Table 4: Vertical Lift Motor Type Decision Matrix

			Motor Types		
Criteria	**Weight**	**DC Motors**	**Stepper Motors**	**Servo Motors**	**Linear Actuators**
Precision	5	2	6	9	8
Power	7	8	7	3	2
Cost	6	4	3	6	8
Size	9	8	6	8	6
Total		162	151	174	156

Appendix Table 5: Hand Compression Motor Type Decision Matrix

			Motor Types	For Hand compression	
Criteria	**Weight**	**DC Motors**	**Stepper Motors**	**Servo Motors**	**Linear Actuators**
Precision	10	2	6	9	8
Power	6	8	7	3	2
Cost	2	4	3	6	8
Size	9	8	6	8	6
Total		148	162	192	162

Appendix B: Task Breakdown

Appendix Table 6: Work Breakdown Structure (WBS)

Task Name	Estimated Work	Completed Work
Team 5 Senior Design Project	826 hrs	223 hrs
Project Management	89 hrs	22 hrs
Status	17 hrs	5 hrs
Time Keeping	10 hrs	3 hrs
Effort Logging	7 hrs	2 hrs
Scheduling	6 hrs	1 hr
Budgeting	8 hrs	2 hrs
Meetings	52 hrs	11 hrs
Formal Team Meetings	20 hrs	3 hrs
Informal Team Meetings	20 hrs	5 hrs
Mentor/ Customer Meetings	10 hrs	3 hrs
Industrial Consultant Meetings	2 hrs	0 hrs
Goal Setting	6 hrs	3 hrs
Initial	3 hrs	3 hrs
Reevaluations	3 hrs	0 hrs
Research	90 hrs	70 hrs
RTOS	18 hrs	16 hrs
Free	9 hrs	9 hrs
Works on our Core	9 hrs	7 hrs
Core	19 hrs	19 hrs
Modern Architectures	8 hrs	8 hrs
Free	8 hrs	8 hrs
VHDL	3 hrs	3 hrs
Motor System Control	28 hrs	20 hrs
Software	12 hrs	8 hrs
Device drivers for each motor	12 hrs	8 hrs
Hardware	16 hrs	12 hrs
Microcontrolller for each/all motors	16 hrs	12 hrs
Arm and Hand Management	12 hrs	11 hrs
Software	2 hrs	2 hrs
Hardware	10 hrs	9 hrs
Solenoid vs. Servo	7 hrs	7 hrs
Voltage feedback to RTOS	3 hrs	2 hrs
User Interface	3 hrs	2 hrs
Software	2 hrs	1 hr
Hardware	1 hr	1 hr

Database System	10 hrs	2 hrs
Feasibility Study	159 hrs	88 hrs
RTOS	44 hrs	25 hrs
Determine Value of RT Capabilities	12 hrs	5 hrs
Determine scope of Project without RTOS	14 hrs	6 hrs
Cost and Flexibility vs. Free and Hard to Use	8 hrs	6 hrs
Determine Time for Implementation	10 hrs	8 hrs
Core	6 hrs	4 hrs
Capabilties Required	4 hrs	3 hrs
Design Vs. Copy & Paste	2 hrs	1 hr
Motor System Control	38 hrs	17 hrs
Weight vs. Motor Power Study	15 hrs	3 hrs
Steppers and Servos	15 hrs	6 hrs
DC Motor to drive down track	8 hrs	8 hrs
Arm and Hand Management	22 hrs	14 hrs
Complexity of Hand and object sensors	6 hrs	3 hrs
Solenoid vs. Servo for hand	9 hrs	9 hrs
Weight of motors	7 hrs	2 hrs
User Interface	21 hrs	6 hrs
Any reuse available	4 hrs	2 hrs
Complexity Necessary	8 hrs	3 hrs
Database requirements	9 hrs	1 hr
Decision Making	28 hrs	22 hrs
Budget	4 hrs	4 hrs
Ease of Implementation	10 hrs	6 hrs
Educational Value	8 hrs	6 hrs
Market Value	6 hrs	6 hrs
Implementation	389 hrs	0 hrs
RTOS	115 hrs	0 hrs
Compiling	50 hrs	0 hrs
Debugging	60 hrs	0 hrs
Loading	0 hrs	0 hrs
Write Summary	5 hrs	0 hrs
Software Device Driver	20 hrs	0 hrs
Core	47 hrs	0 hrs
Customizing	16 hrs	0 hrs
Compiling	20 hrs	0 hrs
Testing	8 hrs	0 hrs
Write Summary	3 hrs	0 hrs
Motor Control System	45 hrs	0 hrs

Circuit Design	20 hrs	0 hrs
Schematic Entry	3 hrs	0 hrs
Breadboarding	2 hrs	0 hrs
Update Schematics	2 hrs	0 hrs
Parts Acqusition	1 hr	0 hrs
PCB Layout	3 hrs	0 hrs
PCB Fabrication	1 hr	0 hrs
PCB Assembly	2 hrs	0 hrs
Testing	8 hrs	0 hrs
Write Summary	3 hrs	0 hrs
Arm and Hand Management System	56 hrs	0 hrs
Sensor Arrangement Design	10 hrs	0 hrs
Circuit Design	15 hrs	0 hrs
Schematic Entry	8 hrs	0 hrs
Breadboarding	3 hrs	0 hrs
Update Schematics	2 hrs	0 hrs
Parts Acquisition	1 hr	0 hrs
PCB Layout	4 hrs	0 hrs
PCB Fabrication	3 hrs	0 hrs
PCB Assembly	3 hrs	0 hrs
Testing	5 hrs	0 hrs
Write Summary	2 hrs	0 hrs
User Interface and Database	60 hrs	0 hrs
GUI Software Design	25 hrs	0 hrs
Java	5 hrs	0 hrs
C#	20 hrs	0 hrs
Database Design	35 hrs	0 hrs
Parts Tracking	35 hrs	0 hrs
Inventory	20 hrs	0 hrs
Warehouse Location	15 hrs	0 hrs
GUI/Database Interface	0 hrs	0 hrs
Project Assembly and Testing	46 hrs	0 hrs
Write Test Procedure	10 hrs	0 hrs
Connections	5 hrs	0 hrs
Track Layout	4 hrs	0 hrs
Individual Unit Tests	27 hrs	0 hrs
Track Motor	3 hrs	0 hrs
Arm Joint Motors	3 hrs	0 hrs
Hand Motors	3 hrs	0 hrs
User Controlled Track Motion	4 hrs	0 hrs

User Controlled Item Retrieval	6 hrs	0 hrs
User Controlled Item Retrieval With Database	8 hrs	0 hrs
PPFS	44 hrs	43 hrs
Introduction	2 hrs	2 hrs
Research Summary	2 hrs	1 hr
Team Bios	1 hr	1 hr
Design Norms	2 hrs	2 hrs
Business Plan	8 hrs	8 hrs
Project Management	5 hrs	5 hrs
Project Requirements	3 hrs	3 hrs
Hardware Design	6 hrs	6 hrs
Software Design	4 hrs	4 hrs
System Design	2 hrs	2 hrs
Feasibility Study	7 hrs	7 hrs
Conclusion	2 hrs	2 hrs
Prepare Final Presentation	23 hrs	0 hrs
Slides	4 hrs	0 hrs
Demo Motion Sequence	8 hrs	0 hrs
Ready for Uneducated User Use	5 hrs	0 hrs
Prepare differences for Design Night Presentation	4 hrs	0 hrs
Rehearse	2 hrs	0 hrs
Prepare Final Report	32 hrs	0 hrs
PPFS Resuse	2 hrs	0 hrs
Intro	2 hrs	0 hrs
User's Manuals	20 hrs	0 hrs
Project Review	4 hrs	0 hrs
Conclusion	2 hrs	0 hrs
Review Report	2 hrs	0 hrs