Automated Foosball Table
Final Report

Submitted to Dr. Michael McGuire

University of Victoria
ELEC/CENG 499A Project
July 29, 2005

Report prepared by:

Darren McElhinney	0322189
Jonathan Turner	0322242
Tiran Matsubara	0329704
Dave Grabowski	0330861

SUMMARY

The Autonomous Foosball Table was our group project for the ELEC 499 project class during the summer session of 2005. The project was to develop and implement automation of the goalie on one half of the table.

The foosball table consisted of many systems. Every system was controlled by the HC11 microprocessor. Each system design was a mini project on its own. In the end, all the systems were put together with the HC11 to control them. The foosball table was now a complete and working prototype of a system used to control the movements of the goalie based on the location of the ball.

The controller was actually able to track the movement of the ball and position the goalie in front of it to stop a goal from being scored. During the project presentation the autonomous goalie was able to defeat a handful of opponents in games up to ten goals. This is impressive given the short timeframe allowed for design and the limited budget available for equipment.

The recommendations for this project would be to invest in better motors and use microcontrollers with more memory. A lot of work was put into this project, but more details could have been added to it. This project has a lot of potential.

TABLE OF CONTENTS

LIST OF FIGURES

INTRODUCTION

The Autonomous Foosball Table is a foosball table that has the goalie on one side being controlled by microcontrollers and the associated hardware. Foosball is a table game/sport played by many people of all ages. It requires good eye-hand coordination and quick reflexes. The concept of the automated foosball table came about from a desire to have some of the players automated so that either a player may focus his attention on the non automated players (thus making it easier for novices), or that (eventually) one whole side of the table may be automated and therefore allow a player to play against the computer.

This project allowed the team to apply electronic and control knowledge learned at UVic to a system that would spark interest from the public and also just provide entertainment. The main purpose of this project was to make the goalie on one side of the table autonomous, and to apply education learned through UVic engineering in the process. This was accomplished with the Autonomous Foosball Table.

The scope of the Autonomous Foosball Table was to be able to track the movements of the ball and position the goalie such that the ball may not pass by it. Also, when the ball was directly in front of the goalie, the goalie should kick the ball. This was accomplished and demonstrated during the project presentations.

This report will first give an overview of Autonomous Foosball Table and its components. The separate systems (hardware and software) will be outlined and then described in detail. Errors will be discussed and then recommendations for improvement and a conclusion will follow.

This report is intended for robotic and/or electronic technicians or engineers that would like to know more about automating a foosball table. It may be interesting to some computer programmers to see what kind of systems may be controlled, also. This report is written at a level such that students or technicians in the field of electronics will be able to quickly understand the ideas behind my project. A background in digital electronics, feedback, DC motors, the Motorola HC11 and assembly language code, and foosball is necessary to understand this project.

1 SYSTEM OVERVIEW

The Autonomous Foosball Table may be broken down into two main systems: Hardware and Software. The software consists of all the routines and inputs/outputs required to monitor and control the system. The hardware has 4 sub-systems: the grid circuit (used to track the ball), the coordinate logic circuit, the microcontrollers and the motor driver. These systems (with the software) monitor the location and vector of the ball and move the goalie appropriately. If the ball rolls in front of the automated players, they will attempt to kick it, thereby exercising the objective of keeping the ball away from the defensive end.

The system block diagram is presented in Figure 1.

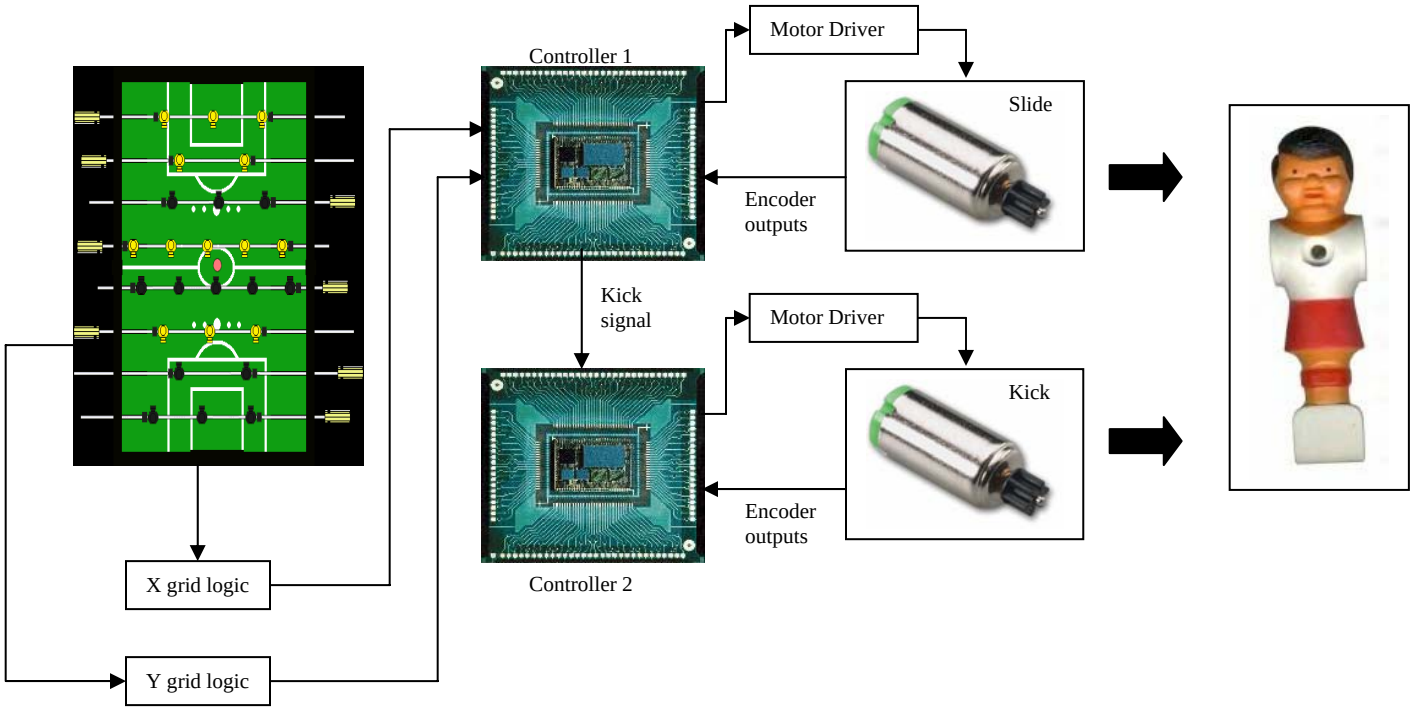


Figure 1: System block diagram

2 HARDWARE

2.1 Grid Layout

A grid of lasers was used to determine the position of the ball on the table. Holes were drilled (¾" between each hole) along the edges of the table to allow for a laser beam to cross the top of the playing surface (just over the table) to be only interrupted if the ball passed in front of them. The grid of lasers is shown in the diagram below (Figure 2). There are photo-detectors (transistors) on the opposite side of the table to detect if the laser beam has been interrupted.

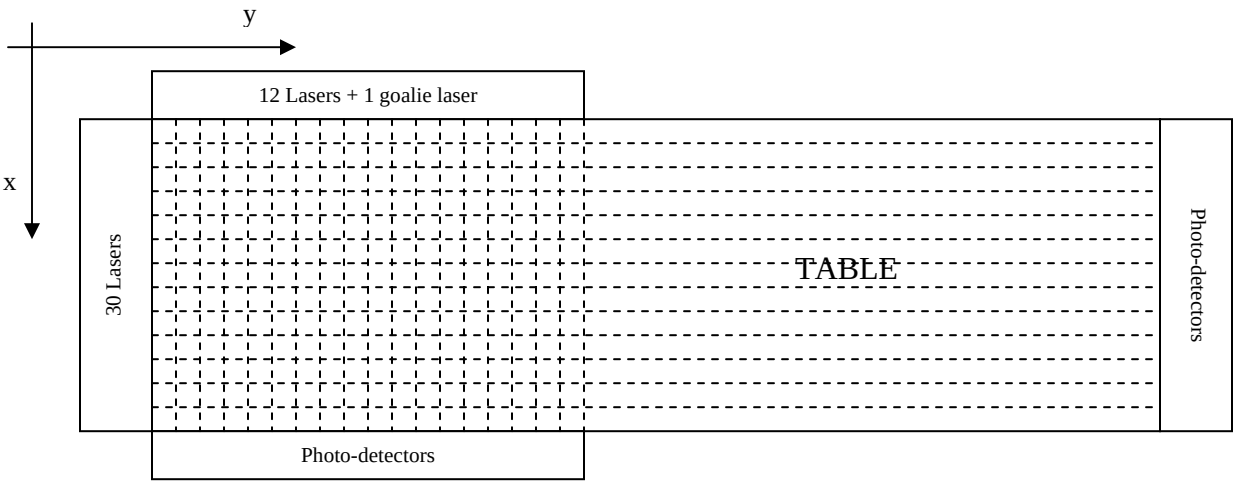


Figure 2: Laser grid

As the ball travels along the table, it intersects both a *y* laser as well as an *x* laser. The corresponding photo-detectors get triggered and output their logic to logic circuits to determine the triggered coordinates.

Note: Only a portion of the table contains *y* lasers, as it is not necessary to monitor the ball's vertical (y) coordinates far away from the goalie. There is a laser in the goal area to indicate if and when a goal has been scored.

2.2 Coordinate Logic Circuit

There are 42 lasers (digital inputs) to keep track of the ball's position. Since the microcontroller can not allow for 42 individual inputs (ie. a word with 42 bits), a method had to be developed to allow the information to be transmitted from the lasers to the controller. A combination of OR gates is used to convert the triggered laser's number

into binary format. The following truth table was used to construct the circuit for 30 lasers, as well as 13.

Laser																												Bit Output							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	0	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	1	1	0	1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	1	0

Table 1: Laser output truth table

The logic obtained from the truth table for the x-coordinate:

Bit 0: 0+2+4+6+8+10+12+14+16+18+20+22+24+26+28
Bit 1: 1+2+5+6+9+10+13+14+17+18+21+22+25+26+29
Bit 2: 3+4+5+6+11+12+13+14+19+20+21+22+27+28+29
Bit 3: 7+8+9+10+11+12+13+14+23+24+25+26+27+28+29
Bit 4: 15+16+17+18+19+20+21+22+23+24+25+26+27+28+29

Using the same method as above to determine the y-coordinate using only 13 lasers:

Bit 0: 0+2+4+6+8+10+12
Bit 1: 1+2+5+6+9+10
Bit 2: 3+4+5+6+11+12
Bit 3: 7+8+9+10+11+12

From this truth table and the logic above, it is shown that two words (one of length 5 and the other of length 4) need to be input into the controller. This is much more reasonable that inputting a word of length 43. When these words are read into the controller, software can determine what laser beams are interrupted and thereby determine where the ball is.

Due to the lack of precision in the laser grid layout, sometimes the ball would interrupt two lasers on the x-grid. Therefore a circuit had to be built to send a signal to the controller to tell it if two lasers were being broken in order for it to make sense when determining the coordinates. The signal from the circuit is entered into PE7 on the board.

2.3 Microcontroller

Two Motorola 68HC11 microcontrollers are used for real-time program response. One microcontroller acts as the master, determining the ball's coordinates and vector, the goalie's position, and where the goalie should move to. The second microcontroller is used to control the goalie's kick. The only communication between the two controllers is when the master controller sends a signal to the slave when a kick is needed.

The master microcontroller's interface is shown in *Figure 3*. The binary outputs from the coordinate logic circuits are connected to ports E and C of the controller. These ports read the digital state of the pins. Remaining bits on the ports (unused pins) are connected to ground (low).

Signal A from the motor encoder is connected to a pin (IC1) on the controller that generates an interrupt on a rising edge (when a pulse occurs). This interrupt sends the controller immediately to a routine to count pulses (and thereby keep track of motor orientation). In the routine, Signal B is read (from PA7) to determine the direction of motor rotation.

Port B is used to output signals to the motor (forward or reverse), to signal to the second microcontroller for a kick, as well as to turn on and off an error light. Pin 0 is used to move the motor forward and pin 1 to move it in reverse. Pin 2 controls the error light and pin 3 controls the kick signal.

To allow for a quick termination of the program, an emergency stop button is connected to the XIRQ pin which has the highest priority over all interrupts. When this signal is triggered, the program immediately stops.

Two microcontrollers were needed because of memory limitations in the chips as well as the need for multiple interrupt inputs to keep track of the positions of the motors.

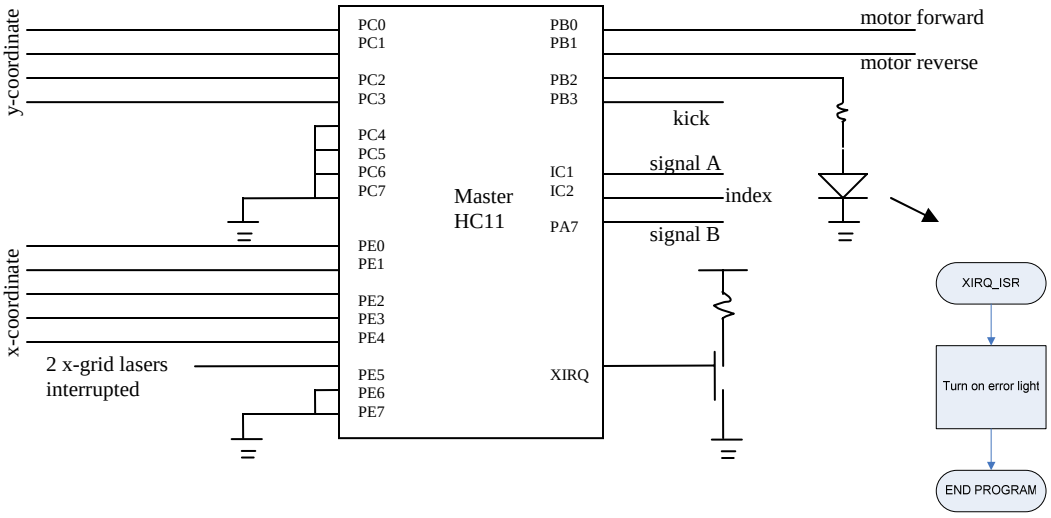
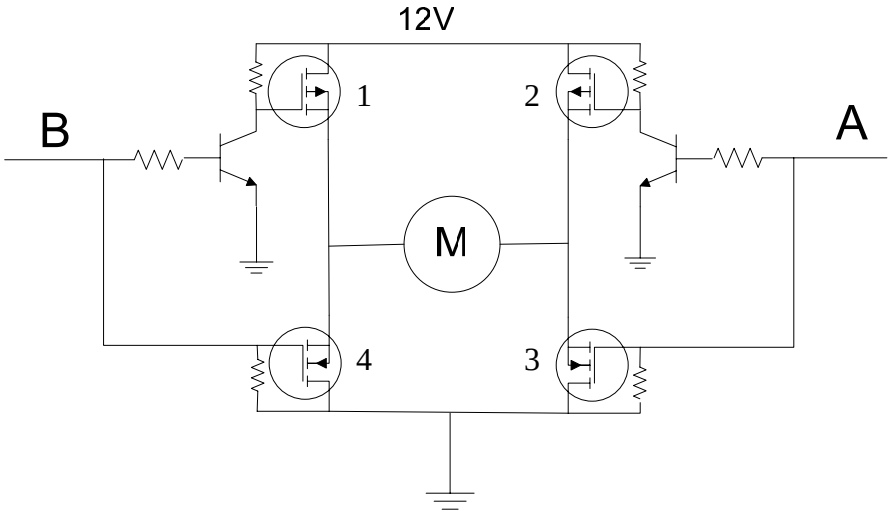


Figure 3: HC11 I/O Schematic

2.4 Motor Driver

An H-Bridge is used as the motor driver's structure. This allows for simple forward, reverse, and stop capabilities using just two signals outputted from the microcontroller. To move the motor forward, the controller outputs a high on signal A and low on signal B. This turns on transistors 1 and 3 allowing for current to flow across the motor creating forward movement. To reverse the motor, the signals are negated, A becomes low and B becomes high. Now transistors 2 and 4 turn on, forcing the motor to turn in the reverse direction. When both signals are low, friction acts as a brake to the motor.

Power MOSFET transistors were used in the H-Bridge. Originally, power BJT transistors were used as they were readily available (on hand) and the students were very familiar with their operation. However, BJT transistors have a large voltage drop across the collector and emitter junctions when large currents flow. This large drop with the high currents consumes a lot of power and this power comes in the unfriendly form of heat. The heat destroyed the transistors and therefore the H-Bridge was redesigned using power MOSFETS which have a much lower voltage drop and can handle higher currents.



All resistors	10k ohm
BJT	2N3904
Mosfets	IRFZ20 and IRF9530 (have internal diode)

Figure 4: Motor driver controller

3 SOFTWARE

The microcontrollers' software is programmed in assembly language. Many subroutines were developed to control the hardware and monitor the tracking of the ball. These routines are discussed below.

3.1 Optical Encoder Tracking

Optical encoders are used to keep track of the revolutions of the motors. Two signals, with a resolution of 500 pulses per revolution, are outputted from the encoder to the controller. The 90 degree phase shift in the encoder signals allows the control to distinguish direction of rotation of the motor. The interrupt routine used to count encoder pulses is generated by a rising edge on Signal A.

3.1.1 Master

In the interrupt subroutine, signal B is checked. If it is high, then the motor is moving forward. The forward pulse count increments if there are no reverse pulses, otherwise the reverse pulse count decrements. Also, the number of forward or reverse rotations updates accordingly if the pulse counts reach zero or 500. (See Figure 5 for the routine used to count pulses).

3.1.2 Slave

In the interrupt subroutine, signal B is checked. If it is high, then the motor is moving forward. The forward pulse count increments if there are no reverse pulses, otherwise the reverse pulse count decrements. The revolution count does not need to be remembered as the motor will only rotate a maximum of 90 degrees in either direction.

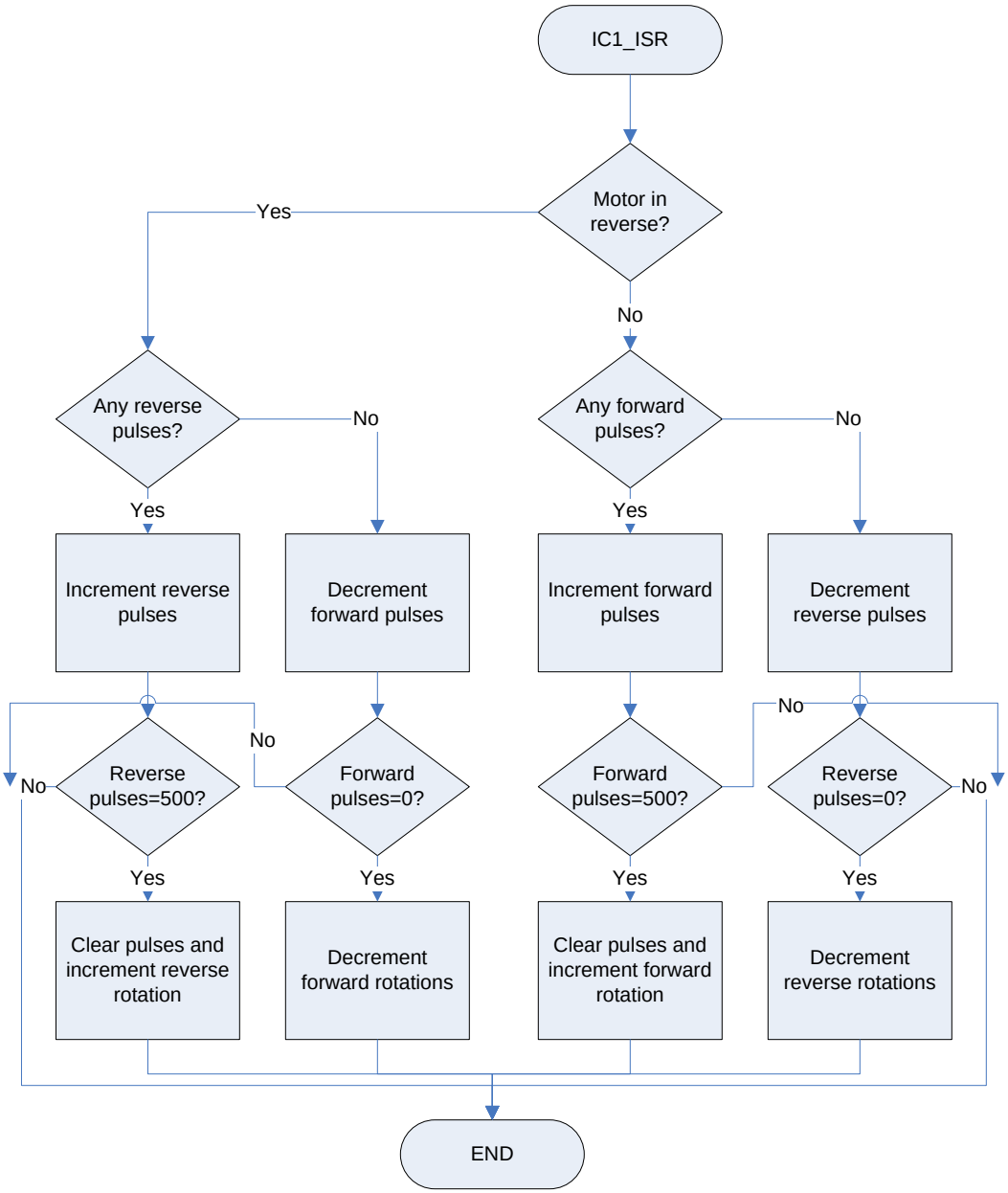


Figure 5: IC1 Interrupt Service Routine

3.2 Homing

Homing the motors is required to give the controller an exact starting position of the goalie. This position is used as a reference and may be recalled every time the program restarts, a goal is scored or if the ball leaves the table. Homing the motors involves orientating the motors to a known location so that incremental adjustments (throughout game play and operation) are relative to an accurate location. The index pulse from the encoder is used in homing as it occurs only once per rotation of the motor.

3.2.1 Master

At the start of the program, the goalie must be manually placed near its home position. The motor will then move forward for a certain number of pulses. If the index pulse was not found, then the motor reverses for a certain number of pulses. When the index pulse triggers an interrupt, the motor stops and all pulse counts reset to zero. The goalie's position is now homed. If the pulse wasn't found, an error light is turned on and the program exits.

Throughout program execution, the goalie is homed whenever the ball isn't detected for more than five seconds, or the goalie was scored on. A "pre-homing" subroutine is first run, before the regular homing subroutine, to move the goalie close to its home position. The homing subroutine is then run to find the precise home position.

3.2.2 Slave

At the start of program execution, the slave microcontroller sends a signal to the motor to rotate forward until an index pulse is found. The index pulse is located when the goalie is standing straight upright and so it will rotate until it reaches this position. The index pulse will generate an interrupt causing the motor to stop. The goalie's kick position is now homed. The goalie's stance will be homed after every kick.

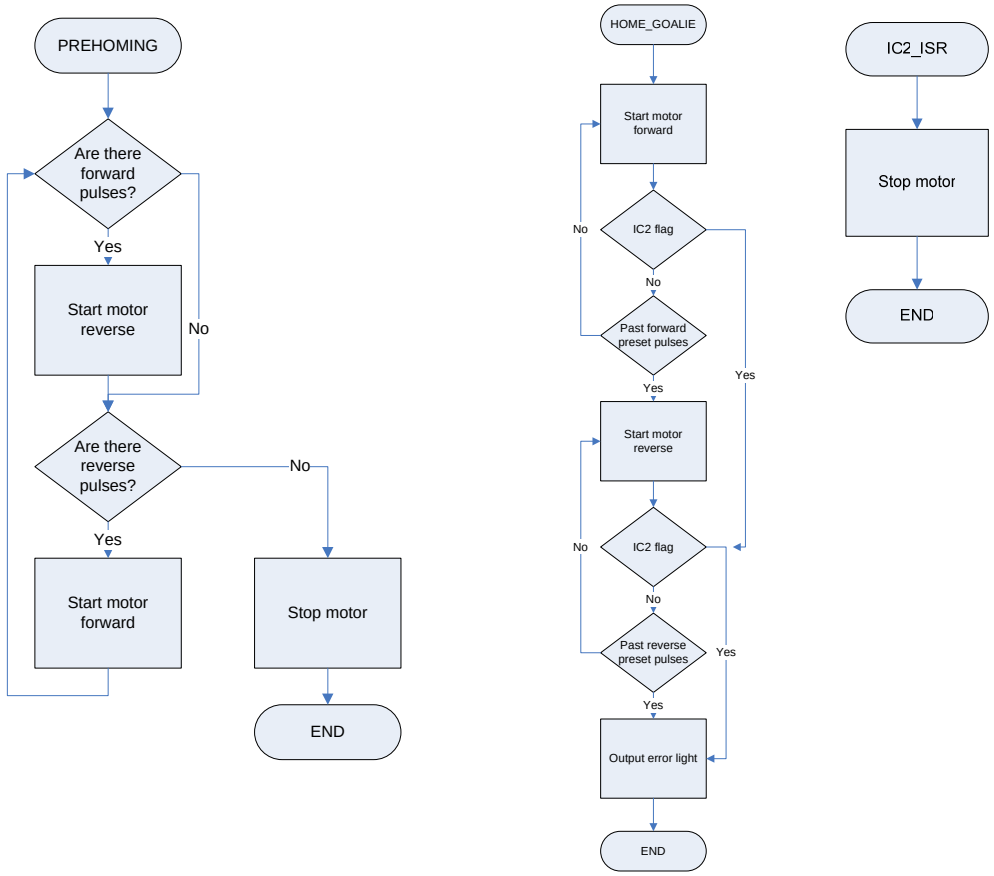


Figure 6: Subroutines for homing

3.3 Computing the Goalie's Position

Computing the orientation of the goalie is only required for the master controller (controlling the horizontal movement of the goalie rail). The rotation of the goalie (the kick) is not as critical as the rotational orientation of the goalie since it is homed after every kick.

The program determines first if the goalie has moved forward or backward relative to the home position. It then multiplies the number of rotations is has gone by 500 (the number of pulses per revolution) and adds the remaining pulse count to this number. This gives the total number of pulses the motor has turned relative to the home position. To determine where the goalie is, relative to the table's home position, this pulse count is divided by the number of pulses per unit (one unit is 0.75"). The result is the number of units from the home position. Finally to determine the absolute x-coordinate of the goalie, this value is either added to or subtracted from the home position value depending on if the motor is turned forward or reverse respectively, relative to the home position.

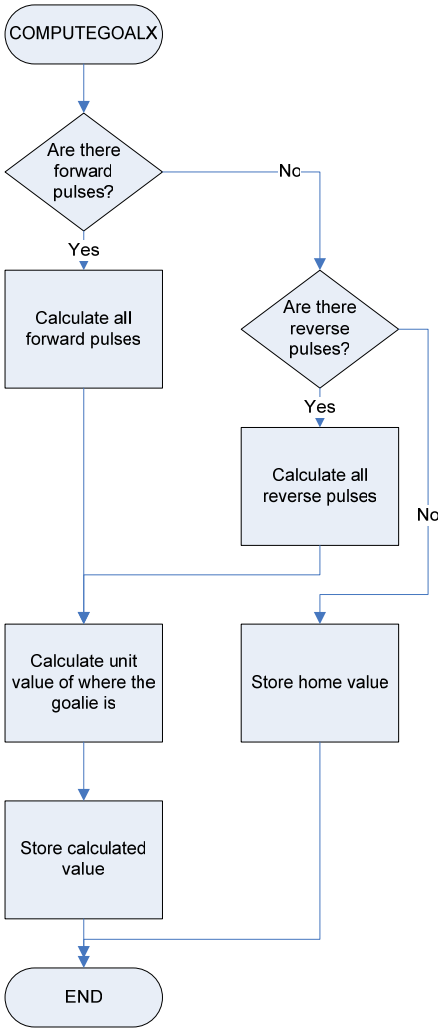


Figure 7: Subroutine to compute the goalie's position

3.4 Computing Where the Goalie Should Move To

The algorithm used to determine where to position the goalie is fundamental for the foosball table to become autonomous. The routines used to compute the position of the goalie involve current and previous coordinates of the ball along with interrupt routines that capture the coordinates of the ball.

A real time interrupt of 4.1ms is used to capture the position of the ball. During the interrupt subroutine, the inputs from the x and y coordinate logic circuits are read. If both coordinates are the same as previous values, then the ball hasn't moved (changed grid points) and so all calculations are skipped. Otherwise, the previous coordinate values are saved in memory and the current coordinates become the current points in memory. This gives us a current position of the ball and a position of the ball 4.1ms earlier. With this information, a vector of the balls travel may be determined.

Because two sets of coordinates (previous and current) are now available, a calculation of where the ball is heading may be performed. If the previous and current y-coordinates are equal, the ball is moving horizontally and so the goalie (or goalie's defensemen) follows the ball's x-coordinate. If the current y value is greater than the previous y value, the ball is moving away from the goalie and so once again the goalie (or goalie's defensemen) follows the ball's x-coordinate. Essentially, unless the ball is moving towards the goalie, the goalie will be position directly in front of the ball. If the current y value is less than the previous y value (the ball is moving toward the goalie) a trajectory of the ball is calculated using the following formula:

$$X = \frac{(Y_{prev} - 2) \bullet (X_{now} - X_{prev})}{Y_{prev} - Y_{now}} + X_{prev}$$

The goalie is located along the y-axis at 2 (this is why a value of 2 is in the formula). This equation predicts the location of the ball along the x-axis as it passes the y-axis at 2 (where the goalie is).

If the vector of the balls trajectory is off of the table (past the edge of where the goalie is able to move), then only the x-coordinate of the ball is used as the destination. If the ball is heading towards a point to the left of the goal, but within the table, the goalie will move to (x-point + 9 units) to allow the goalie's left defenseman to hit the ball. If the ball is heading towards a point to the right of the goal, but within the table, the goalie will move to (x-point – 9 units) to allow the goalie's right defenseman to hit the ball. The defensemen are spaced 7" or around 9 units from the goalie in both directions.

Two last checks of the current y value determine if the goalie line should kick or if they've been scored on. If the y value is 3 (one unit in front of the goal line), then a kick signal is sent. If the y value is 13, the laser in the goal was triggered and so they've been scored on. A flag is updated and the subroutine exits.

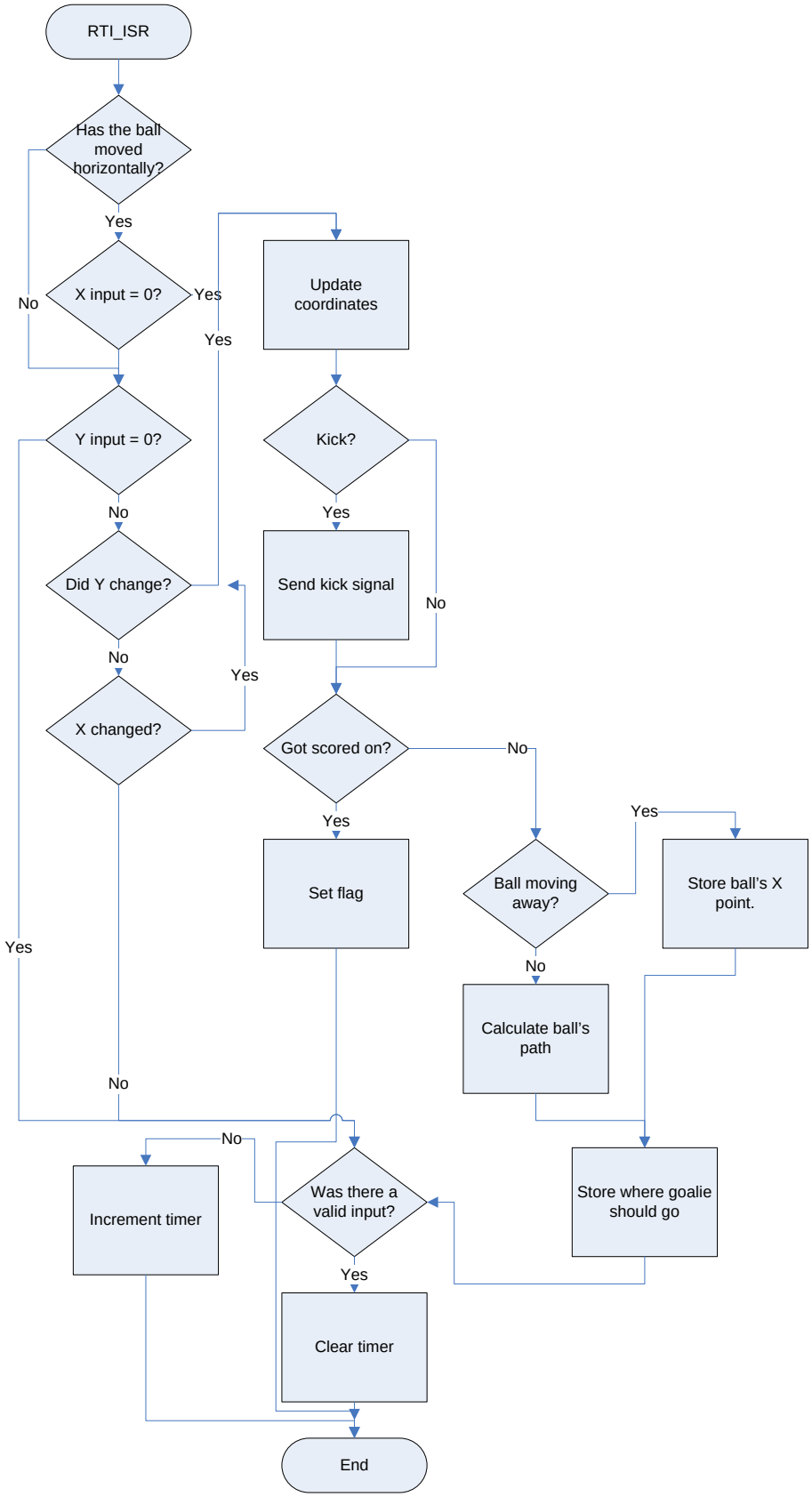


Figure 8: RTI interrupt subroutine

3.5 Main

The main program is in a loop constantly checking if the goalie is in the correct position. It also checks that the timer hasn't gone over 10 seconds. If it has, it means the ball is not on the table and the goalie should re-home itself.

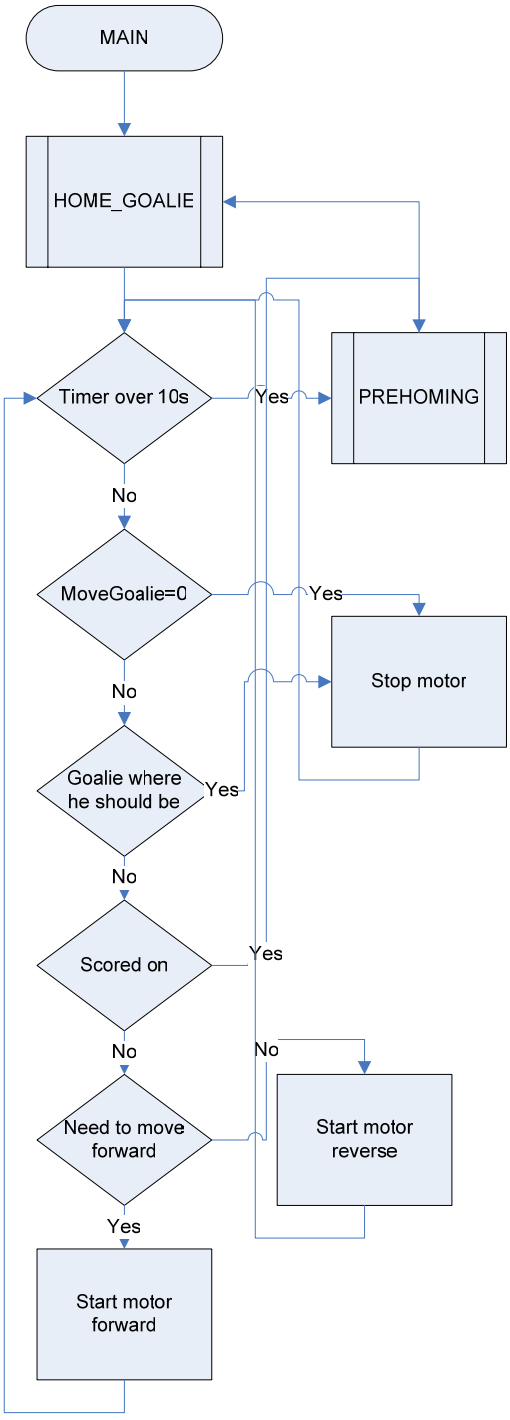


Figure 9: Main program

4 PERFORMANCE REQUIREMENTS

The goal of this project was to control the orientation of the goalie on the foosball table such that it would prevent the ball from entering the goal. If the ball is within reach of the goalie, the goalie should kick the ball to return it to the opposition's end of the table. The computer was able to track the position of the ball and control the movement of the goalie sufficiently to accomplish the performance requirements.

5 LIMITATIONS

- The biggest limitation is the HC11's memory size and processing power. Very complicated algorithms to control the hardware could not be implemented because the memory available and processor speed were not sufficient.
- The response of the motor will affect the performance of the goalie. If it goes too fast, many encoder pulses will be missed and the goalie's accuracy will diminish.
- A way to react quickly to the ball's position needed to be devised and so the laser grid system was implemented. It is not the most accurate way to determine the ball's position, but it is the quickest.
- Again because of the HC11's memory size, only a simple formula could be used to calculate the ball's trajectory. However, because the trajectory is being constantly updated, this should prove to be enough.

6 CONCLUSION AND RECOMMENDATIONS

The project did meet the goals set out to determine if it was a success. However, as with all projects, there are ways that future models and systems may be developed to allow for more efficient operation. The main systems that could be improved would be the laser grid system and the microcontrollers.

The laser grid system was the most convenient choice for the project given that there was a limited budget. However, since the lasers were cheap ($1 each) they were not reliable. Sometimes lasers would die or burn out. Also, the software used the value of .75" between each laser beam to calculate the position of the goalie (from the coordinates of the ball). Since the holes were drilled using a hand drill, accuracy cannot be confirmed and therefore there are errors in the positioning of the goalie. Basically, the grid setup was not very stable. Impacts to the table or especially to the railings set the lasers off line and ineffective in tracking the ball. To track the ball more accurately a sturdier and more effective means must be developed.

The microcontrollers had limited memory capabilities and this was an issue for software development. With more room for code and quicker calculations, routines could have been developed to produce optimum motor control.
The motor control used was very basic, digital ON/OFF. Even though the encoder would allow the computer to know where the goalie was positioned, it was not used as feedback to correct for overshoot (for example). Once the goalie reached the desired location (based on encoder pulse counts) the motor would be put into brake mode and allow friction (from the sliders and motor) to stop the goalie. This proved to be sufficient due to the short distances the goalie moved (and therefore didn't build up enough momentum to overshoot too much), however if the code was used on a different piece of hardware with different physical variables then problems could arise.

Feedback control was originally used, but memory space was required for other systems and also, the feedback would cause the motors to vibrate in position, essentially bouncing back and forth between encoder pulses. This can be corrected in software, but also requires room in memory and more computation time. The software developed did meet the requirements of the project and therefore was implemented.

With the limited time and budget, the project was a success in such a way that someone may be able to play against the autonomous goalie and find it a challenge. The system is not perfect such that goals could still be scored on the goalie; however, for a novice to intermediate player, a challenge would be met in playing against the automated side. To increase the level of difficulty against the automated side better motors with faster responses and associated software could be developed.