Mega Pinball

A Report Dedicated

To

Dr. Furman

In Fulfillment

Of

ME 106 Fundamentals of Mechatronics

Department of Mechanical and Aerospace Engineering

San Jose State University

By

Bryan C. Asuncion

Sophaly Hiep

Israel Alleyne

Submitted December 9, 2003

Summary

The object of mega pinball was to make a project that would function as a cross between foosball and pinball. The original ideal entailed making the project with eight motors to turn the paddles, an actuator to make the table turn while playing, and a stepper motor in the middle to have a small area rotate while playing. Our end product varied from our original concept in a variety of ways.

Instead of having eight motors, the group used four solenoids for our paddles. The actuator and the rotating stepper motor were eliminated as a result of time and budget constraints. In the end, the group added two sensors to both sides of the project so that they could mark when someone scored a point. In addition D-type flip-flop chips were added in order to keep count of the points scored and 7 segment BCD up/down counters were added to display the time during the game. In order to keep track of time and to limit the amount of game play, we programmed a timer and a score sequence in the OOPic to monitor the activity of the game. In order to display the outcome of scoring a point, the score, and the time counters, 7 segment BCD displays and LED's were used. The group made aware early on in the development of the game that the solenoids were going to use a good deal of current, and so the solenoids were provided with their own power source, all of the other chips, diodes, and counters were attached to another secondary power source independent of the power source used for the solenoid.

When making the project, it was realized that the counters needed to connect resistors in series in order to function properly. In the lab, this was done using a resistor pack; however the group decided to accomplish this using regular resistors soldered together individually. Initially we did not have this resistor connection, and burned out an element of one counter display. In the end, we found out that the difference between theory and reality must be taken into account in order for a project to work successfully.

Introduction

The assignment for the semester project was to design and build an interactive game that can be played by one or more players. The game should be able to be operated in an obvious manner by buttons or switches to any player. The game should also give feedback to the player(s) during and after the game play. Feedback can be provided by way of a point system or timing system. The OOpic microcontroller must be implemented and be used to control a certain aspect of the game. The skills necessary to program the microcontroller was obtained through weekly laboratory sessions. Each lab consisted of experiments that of which incorporated various circuit elements for the game.

The first step in developing a game is brainstorming various ideas that could be possible projects. It was decided that in order to create a successful game, it must be fun and highly interactive among the player(s). The group was focused on creating a game that had continuous game play and that would allow the player(s) to constantly be active in the play. The idea to create a pinball machine arose as its game play involves the player to constantly pay attention while the ball is in action and that the player is always informed of their points during the game. The group decided that combining the idea of pinball to the project would meet our criteria as a fun game. In order to make the game even more intense, the idea of a multiple player game would be introduced into the project. The combination of the described elements of our idea of a fun and interactive game lead to the creation of Mega Pinball.

Description

The object of the game is to score as many goals as possible within the determined time length. The game time frame of playing is set at 99 seconds and the maximum score a player could reach is 9. Goals are scored when a player shoots the ball past the other player’s goal. Balls are shot by paddles fitted on either end of the goal. A buzzer sounds during the start of the game, at each goal score, and at the end of game play. The player with the most scored goals within the time frame wins.

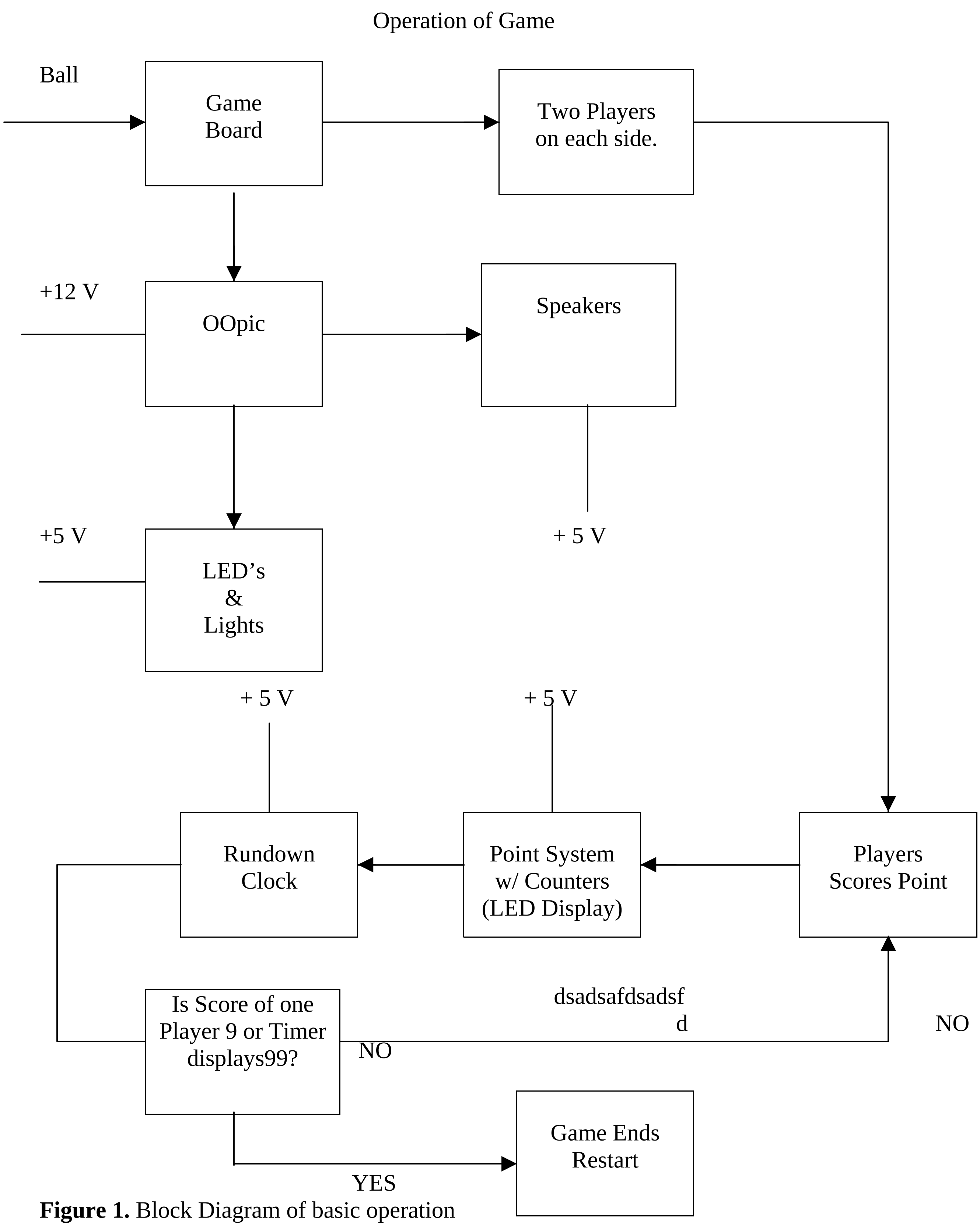


Figure 1. Block Diagram of basic operation of game.

Starting Game

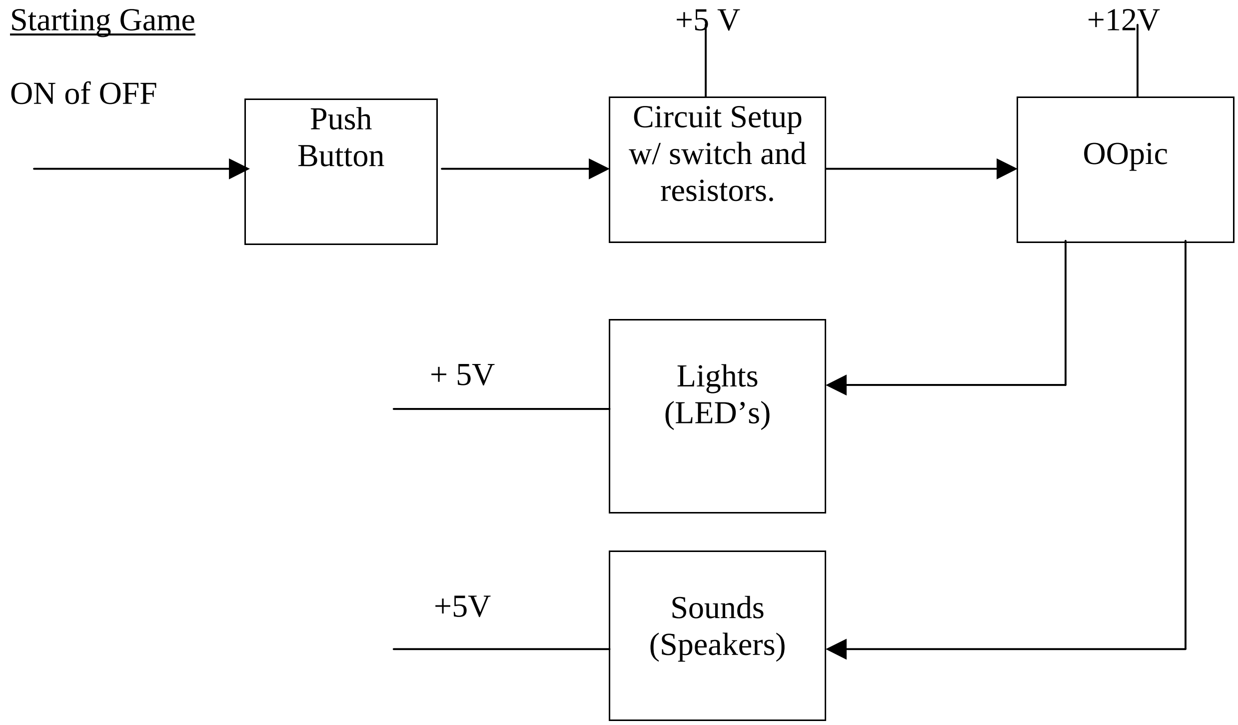


Figure 2. Block Diagram for starting the game.

Sensor for Ball Drop

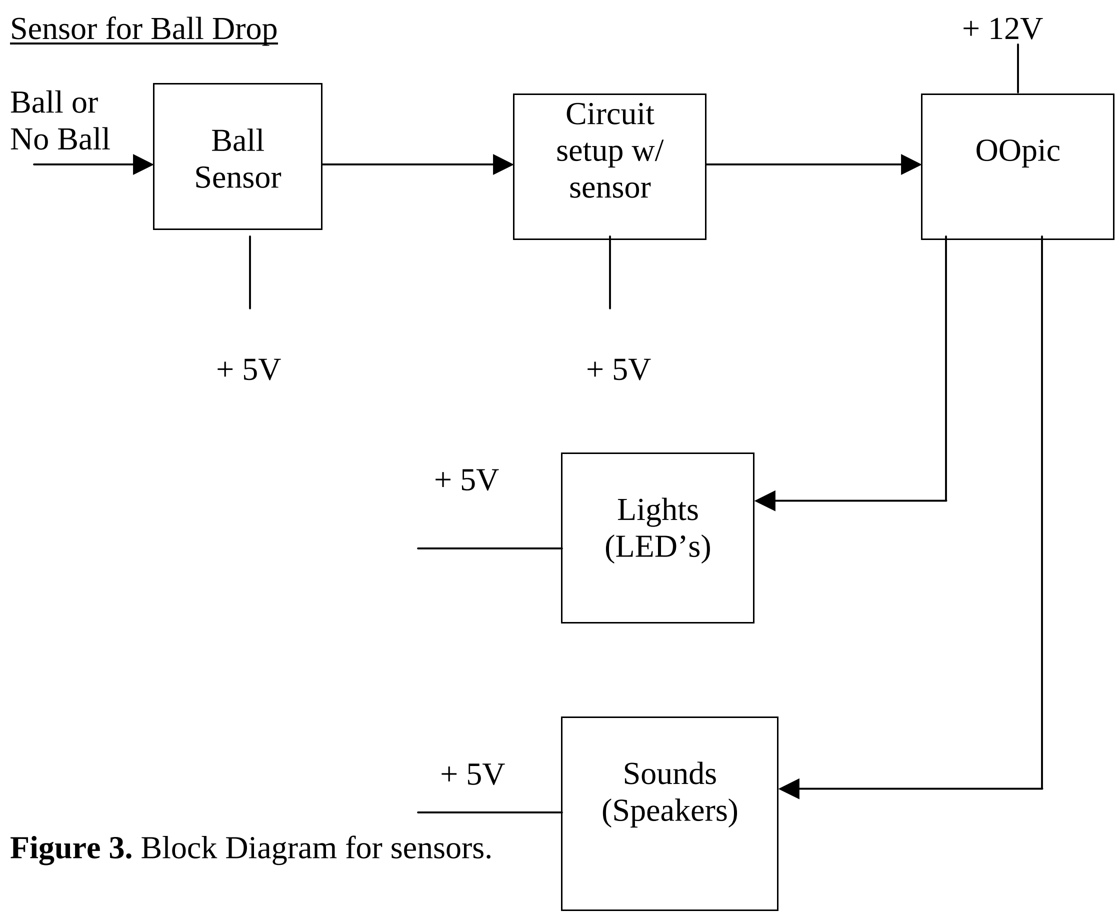


Figure 3. Block Diagram for sensors.

Solenoid Controlled Paddles

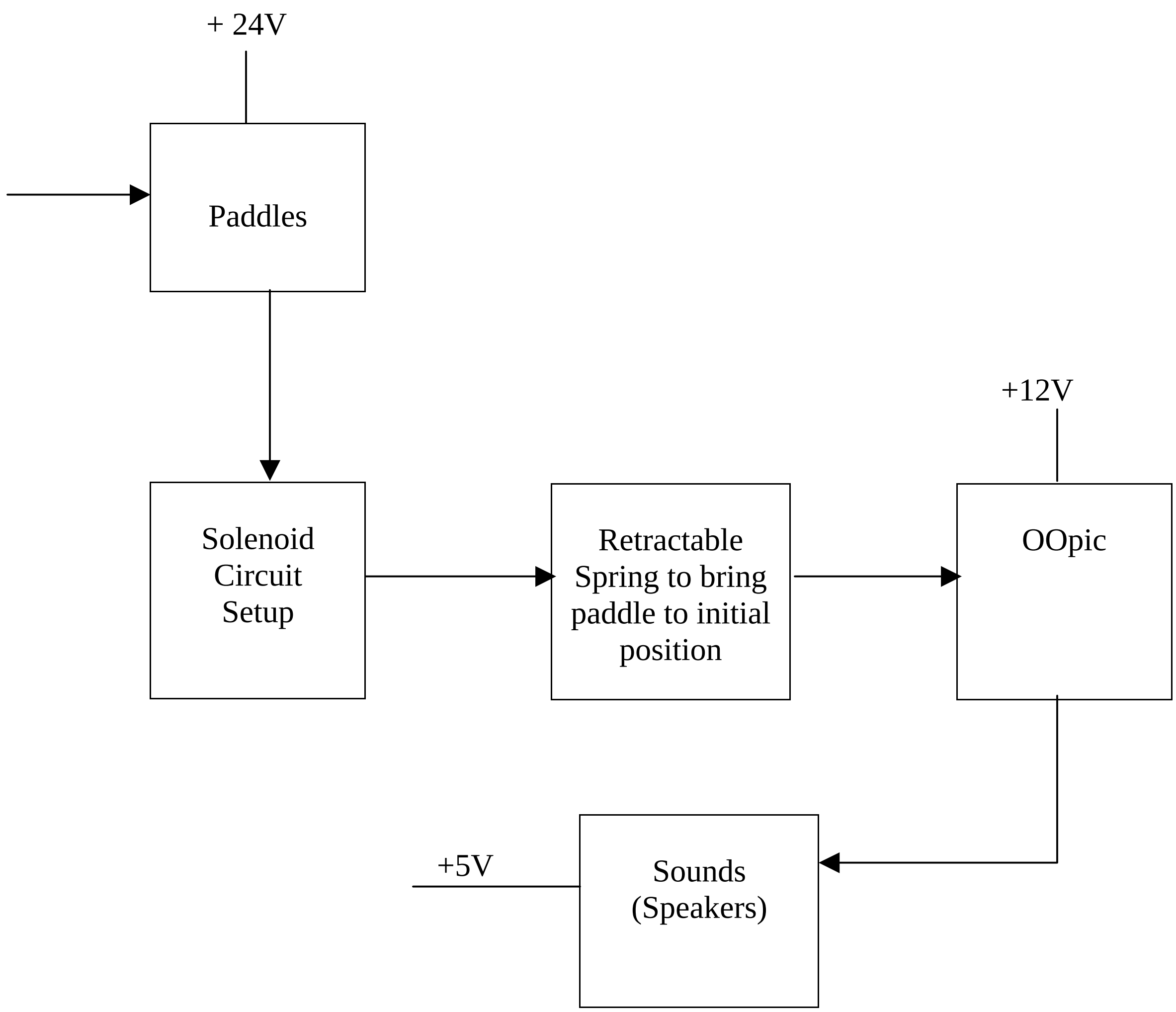
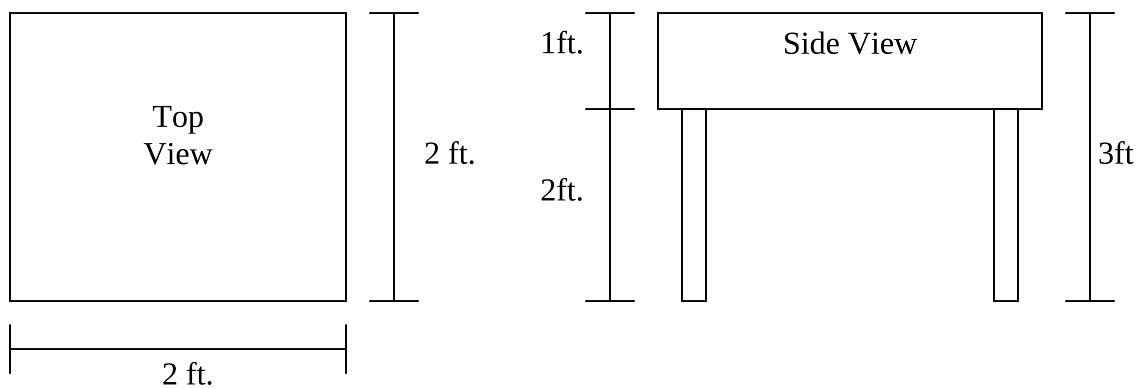


Figure 4. Block diagram of operating solenoids (paddles).

Mega pinball consists of various working components that of which encompass our knowledge acquired in Mechatronics. The initial development stages of the game included the construction of the game board. The game board was originally fabricated from plywood and finishing boards. In order to accommodate two players, the game board was 2 ft. by 2 ft. The height of the board was chosen to be 3 ft. as to accommodate any player's height.



Game Board

Figure 5. Dimensions of constructed game board for Mega Pinball. The game board was build as to allow everybody to play.

The design of the playing board required accurate designing and precision measurements. The idea of continuous play led the group to design a playing board unlike that of an existing pinball machine. Pinball machines use flat angled playing surfaces in the design. With two players, the group decided to incorporate a similar design. The playing surface was initially picked so that it can be bent easily. A bent playing surface allows the ball to roll freely on either side of the board. The playing board was fitted with side rails so that the action of play is concentrated in the center of the board. The playing board has fitted holes so that the ball has a place to pass. After each ball passes through the goal, it goes through a pipe in which the ball can be picked up and placed again on the playing board. The signal goal lights go off after each scored goal.

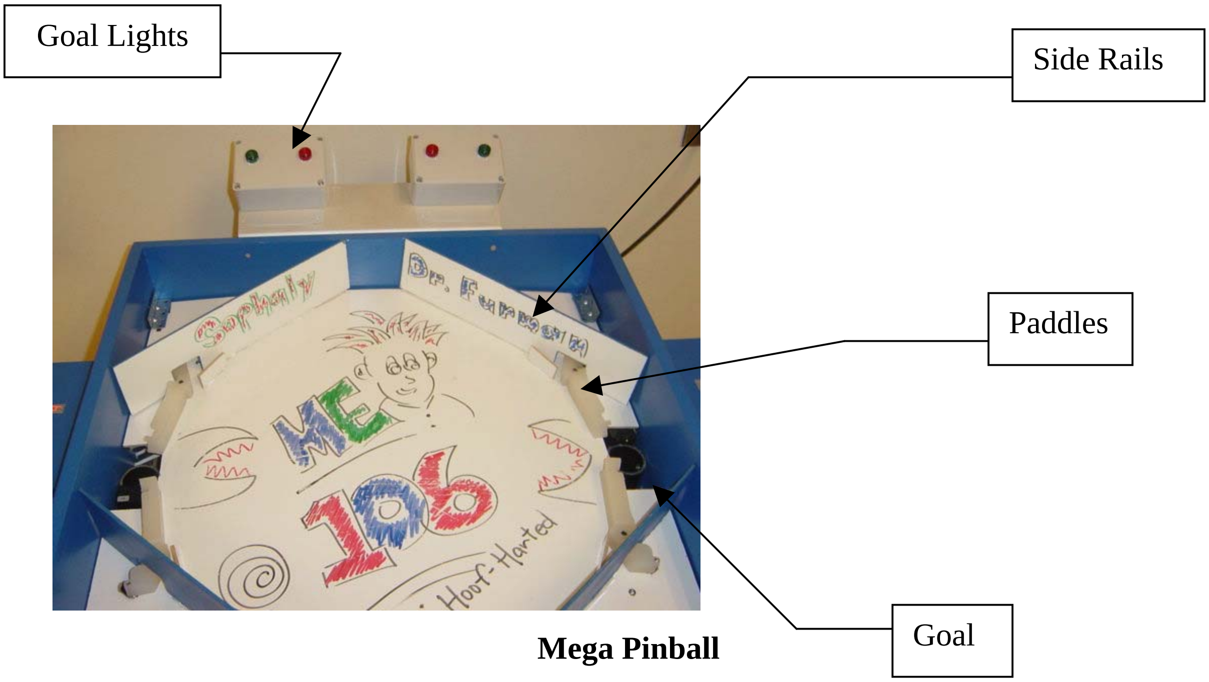


Figure 6. Top view of actual game board with components.

Each player has two paddles that can be simply operated with the push of a button. The paddles are spaced evenly so that the ball can freely drop into the goal. The operation of the paddles involves the use of electric solenoids that require a separate power supply. The solenoids are rated at 24 volts DC. We connected the solenoids to a 24 volt power supply unit. When the player pushes the button, the solenoid engages. The paddle swings forward and then returns to its position by the spring attachment according to paddle mechanism design in Figure 3 and Figure 4.

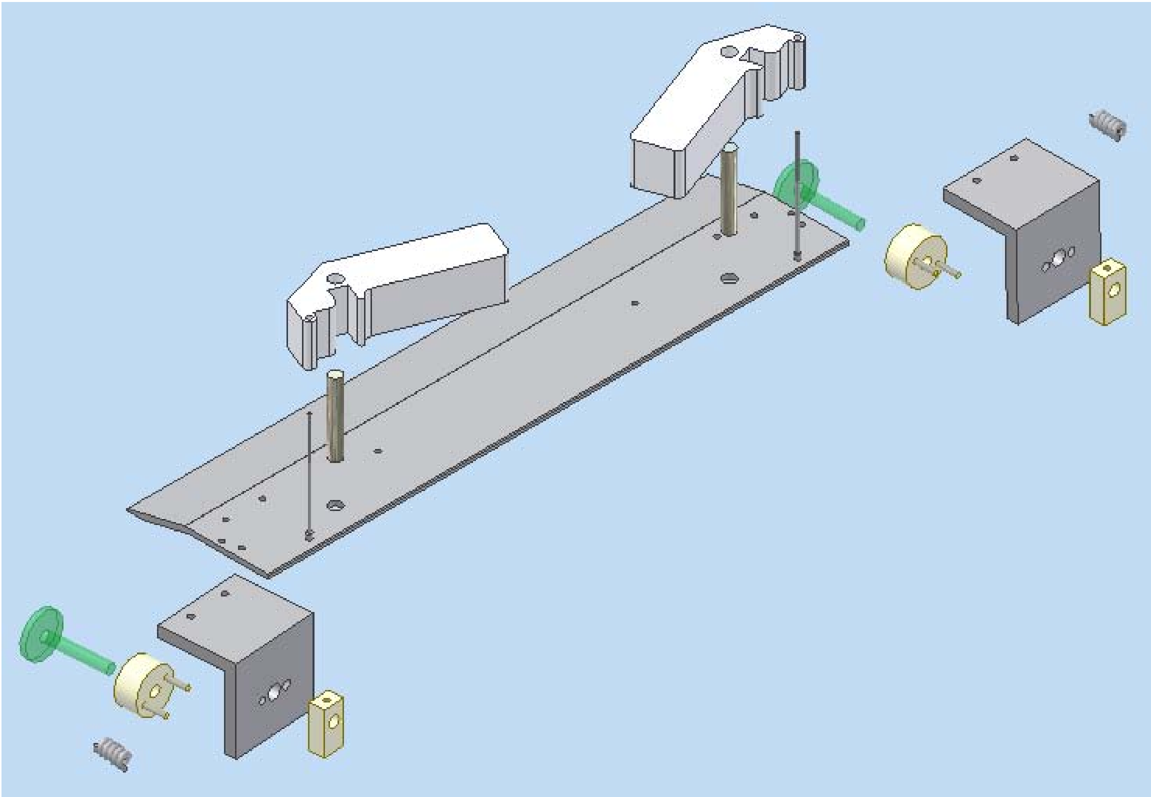


Figure 7. Exploded View of Solenoid Paddle Mechanism

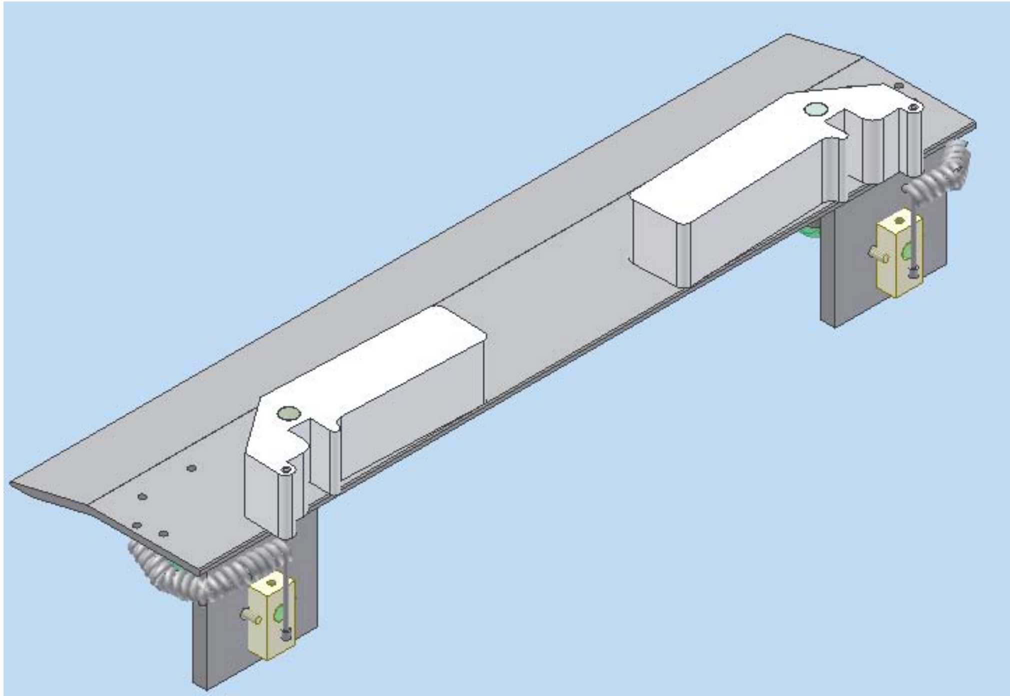


Figure 8. Assembled Drawing of Paddle Mechanism

When 24 volt DC is applied, the solenoid is actuated. We connected the solenoid for the paddle mechanism according to the schematic in Figure 4.

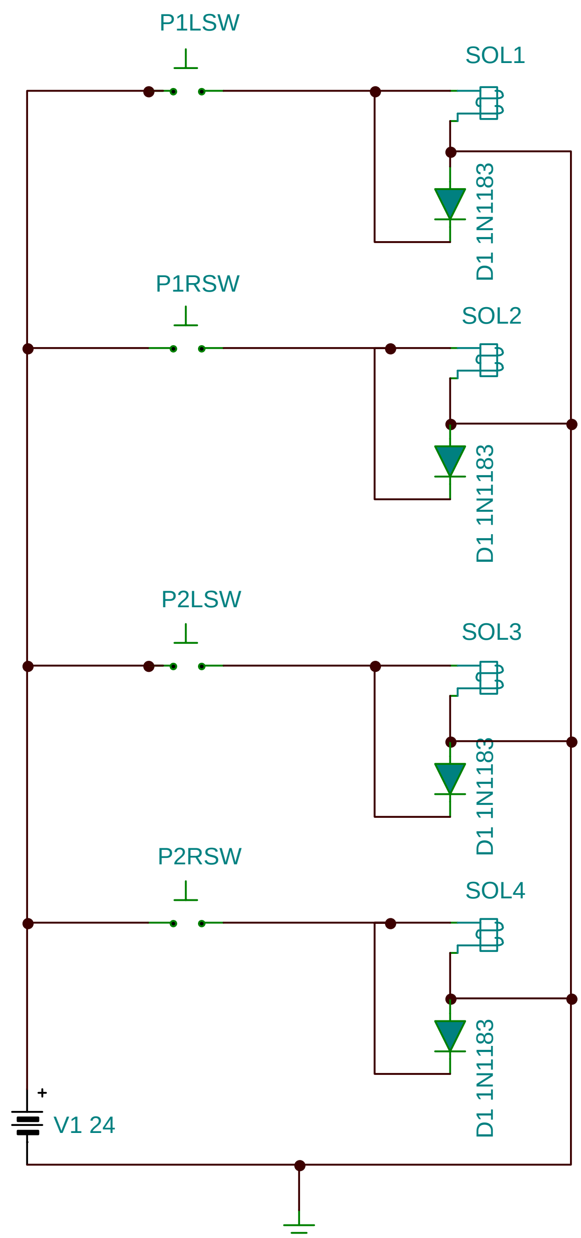


Figure 9. Connection Circuitry for the paddle electric solenoid.

The feedback system consisted of a timer and a score that would be updated constantly throughout the play. Scores and times would be shown by way of 7-segment counters at each end of the board. We designed the circuit for the game board by applying all the hands on experience that we acquired from the laboratory experiments. We built our circuit according to the schematic diagram in Figure 6. We wired our circuit according to the pin-out diagram from the chip manufacturer as shown in Figure 7 and Figure 8.

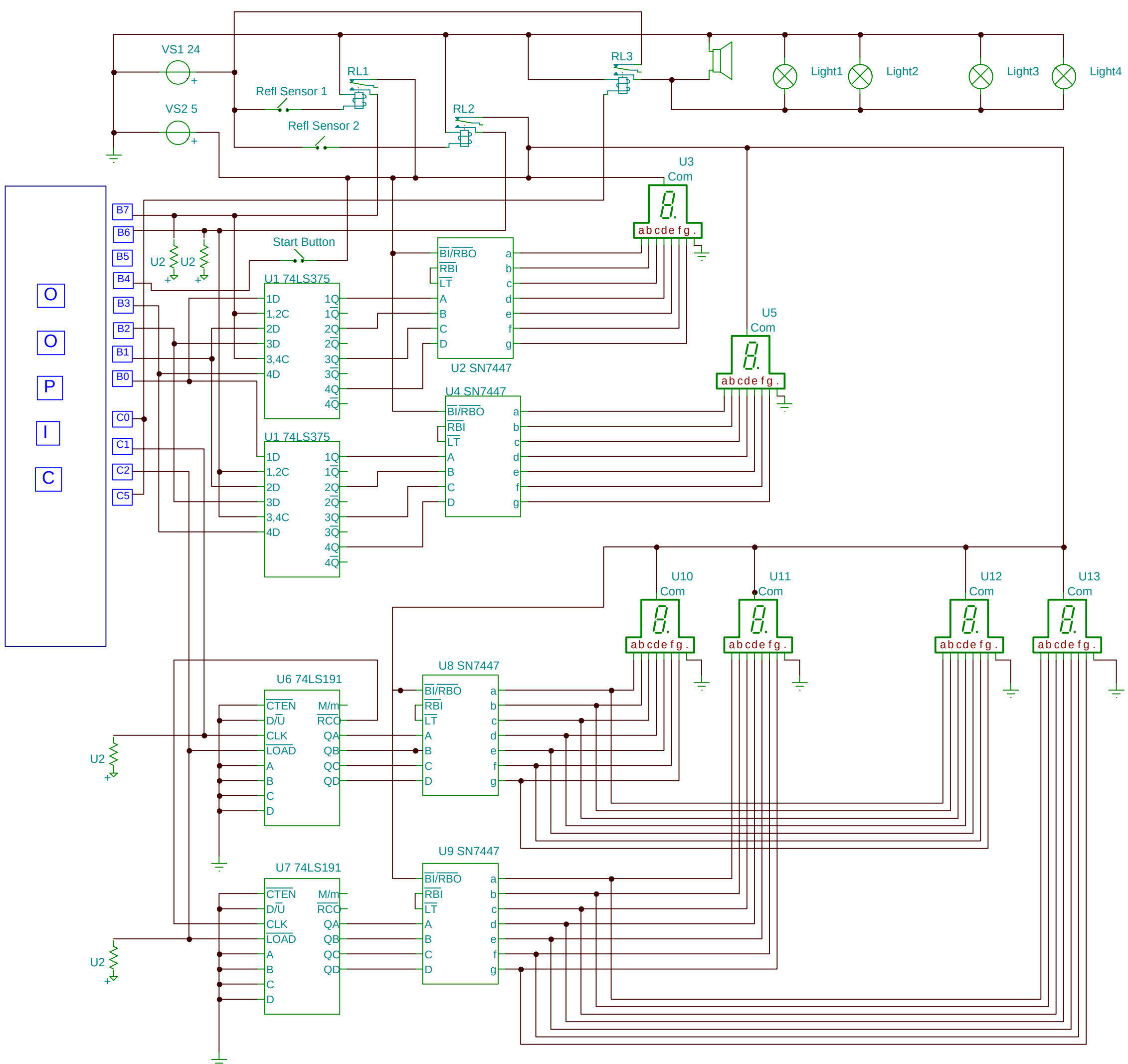
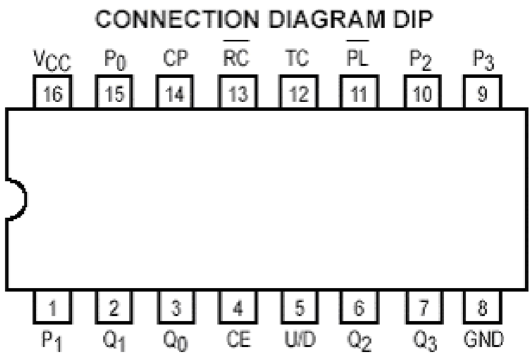


Figure 10. Logic Circuitry, time display, and player 1 and player 2 score



MODE SELECT TABLE

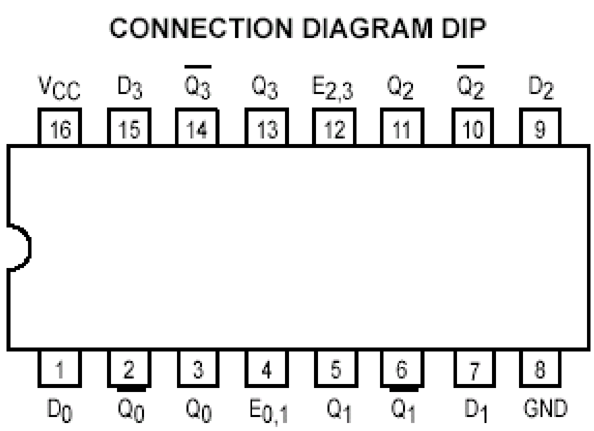
INPUTS				MODE
PL	CE	U/D	CP	
H	L	L	⌐	Count Up
H	L	H	⌐	Count Down
L	X	X	X	Preset (Asyn.)
H	H	X	X	No Change (Hold)

PIN NAMES

$\overline{CE}$	Count Enable (Active LOW) Input
CP	Clock Pulse (Active HIGH going edge) Input
U/D	Up/Down Count Control Input
PL	Parallel Load Control (Active LOW) Input
P_n	Parallel Data Inputs
Q_n	Flip-Flop Outputs (Note b)
RC	Ripple Clock Output (Note b)
TC	Terminal Count Output (Note b)

L = LOW Voltage Level
H = HIGH Voltage Level
X = Don't Care
⌐ = LOW-to-HIGH Clock Transition
⌐⌐ = LOW Pulse

Figure 11. Chip 74LS190, Up/Down BCD Decade Counter Connection Diagram



PIN NAMES

$D_1 - D_4$	Data Inputs
E_{0-1}	Enable Input Latches 0, 1
E_{2-3}	Enable Input Latches 2, 3
$Q_1 - Q_4$	Latch Outputs (Note b)
$Q_1 - Q_4$	Complimentary Latch Outputs (

TRUTH TABLE
(Each latch)

t_n	t_{n+1}
D	Q
H	H
L	L

Figure 12. IC chip 74LS375, 4Bit D-latch, temporary storage for binary input/output

Project Outcome

Despite extensive preparation time, and several hours of planning; we have seen first hand how everything does not work according to plan. We were able to make the solenoids function properly. These however had an unanticipated side effect on the counters.

The counters kept track of both the time and score faultlessly when the solenoids were not actuated. Once the solenoids were actuated however, the counter values on the display fluctuated, and the score tracker set both values equal to the value being scored. Initially, we assumed that this was because the solenoids dragged the voltage down. To counter the aforementioned problem, we decided to isolate the power supply between the circuits and the solenoids. Isolating the circuitry and the solenoids did not achieve the desired results either. By accident, we found out that the same thing happened if we used a drill instead of the solenoids.

This led us to the conclusion that the magnetic field produced by the solenoid had a negative impact on the D-type flip-flop and the 7-segment BCD up/down counter. This disturbance in the D-type flip-flop and 7 segment BCD up/down counters in turn is assumed to affect the output of the counters.

We know by observing the OOPic while it is running that the initial program does not change and so we eliminated this as being the cause of the fluctuations. If we had more time, and a bigger budget, we would implement several changes to our project. We would include enough paddles for four players at the same time. We would also include another power supply in order to power the additional solenoid. Given more time, we would have aggressively tried to find the root source of the fluctuations, and to try and correct it to minimize any anomalies. Instead of simply curving the playing surface so the ball doesn’t get stuck, we would have applied an actuator so that the playing surface rotates.

References

Introduction to Mechatronics and Measurement Systems 2nd ed., Histand, M. B., Alciatore, D. G., WCB/McGraw-Hill, Boston, 2003.

Programming and Customizing the OOPic Microcontroller, Clark, D., McGraw-Hill (TAB Robotics), New York, 2003.

Debco Online. Retrieved 3 Nov 2003 from: http://debcoelectronics.com/catpages/product74LS190. html

Motorola. Retrieved 8 Nov 2003 from: http://ac.gul.pt/Documentacao/Praticas/MotorolaTTL/sn74ls375rev5.pdf

Appendix

Component List:

84	22 Ohm resistor (1/4 Watts 5% tolerance)
4	1 K Ohm resistor (1/4 Watts 5% tolerance)
4	7 segment BCD display decoder
2	4-bit D-type latch (74LS375) temporary binary input/output storage
2	Resetable BCD/decade up/down counters (74LS190)
4	Push button (Monentary)
4	24 Vdc electronic solenoid
2	Diode
2	24 Vdc Refective sensor
3	5 – 30 Vdc changerover relay
4	24 Vdc lamps
1	Speaker

Game Program for the OOPic

```
// Display score For hockey game
// Sophaly Hiep, Bryan, Israel
// 12/02/03
oDio4 Display = New oDio4; // Declare a 4-bit digital I/O object For
447 interface
oDio4 Switches = New oDio4; // Declare a 4-bit digital I/O object For
interface to the switches
oNibble Score1 = New oNibble;
oNibble Score2 = New oNibble;
oWord i = New oWord;
oWord j = New oWord;
oWord TotalTime = New oWord; // Declare a 8-bit digital I/O object for
time display
oFreq ScoreTone = New oFreq; // Declare frequency object for the
audible alert
oDio1 StartTone = New oDio1; // Declare a digital I/O object for
audible alert
oDio1 Timer = New oDio1;
oDio1 ResetTimer = New oDio1;

Sub Void main(void)
{
// Initialize Timer
Timer.ioline=17; // Connect timer trigger to C1
Timer.Direction = cvOutput;

ResetTimer.ioline=18; // Connect reset timer trigger to C2
ResetTimer.Direction = cvOutput;
```

```
// Initialize alarm
// ScoreTone.ioline = 21;//Connect scoretone to C5 only use For oTone
StartTone.IOline=16;//connect startup tone to pin C0
StartTone.direction = cvOutput;

// Initialize Display object
Display.IOGroup = 1; // Use pins in IO Group 1 (B0 – B7)
Display.Nibble = cvLow; // but only use the lower 4 in the IO Group (B0
– B3)
Display.Direction = cvOutput; // Make pins digital outputs
// Initialize Switches object
Switches.IOGroup = 1; // Use pins in IO Group 1 (B0 – B7)
Switches.Nibble = cvHigh; // but only use the upper 4 pins in the IO
Group (B4 – B7)
Switches.Direction = cvInput; // Make pins digital inputs


While(1)
{
   Switch (Switches.Value)
      {
         Case 2: // Start the game If the start button is pressed
                  // Initializing all the display to zero
                  Score1 = 0;
                  Score2 = 0;
                  Display = 0;
                  StartTone = 1;
                  OOpic.delay = 100;
                  StartTone = 0;

                  StartGame; // call StartGame Subroutine


}
      }

} // main

Sub Void StartGame(void)
{
OOpic.Delay = 100;
   ResetTimer = 0; // signal to reset timer to zero
   ResetTimer = 1;
   TotalTime = 99; // set total time to display 99 seconds

For (j = 0; j < TotalTime; j++) // loop For the total time display
{

   Timer = cvTrue;  // Signal For timer
For (i = 0; i < 50; i++) // value For i < 50
{

// Test For Switch to activate and write value to the display
Switch (Switches.Value)
{
Case 8: // (B7 is high) when Reflective Sensor for player 1 is
activated
```

```
Score1++; // Increment player 1 score 1 time

If (score1 == 9) // End the game If player score is 9
   {

       ToTalTime = j; // change value of total time to stop the game

   }

Display = Score1; // Write a value store in Score1 to the 7447
ScoreTone.operate = 1;
ScoreTone.Value = 5000; // play tone at frequency value
oopic.delay = 50;
ScoreTone.Value = 30000;
oopic.delay = 50;
ScoreTone.Value = 100;
oopic.delay = 150;
ScoreTone.Operate = 0;

Break;
Case 4: // (pin B6 is high) when reflective Sensor for player 2 is
activated

Score2++; Increment player 2 score 1 time

If (score2 == 9) // End the game If player score is 9
   {
      TotalTime = j;

   }

Display = Score2; // Write a value store in Score2 to the 7447

ScoreTone.operate = 1;
ScoreTone.Value = 5000; // play tone at frequency value
oopic.delay = 50;
ScoreTone.Value = 30000;
oopic.delay = 50;
ScoreTone.Value = 100;
oopic.delay = 150;
ScoreTone.Operate = 0;

Break;

} // Switch

 } // For i loop to monitor For which player just score

   Timer = CVFalse;

 } // For j loop For the counter


 // End of Game play tone and flash light

// Loop the audible alert when the game is over
```

```
For (i = 0; i < 5; i++)

 {
   ScoreTone.operate = 1;
   ScoreTone.Value = 30000;
   oopic.delay = 65;
   ScoreTone.Operate = 0;
   oopic.delay = 10;
   ScoreTone.operate = 1;
   ScoreTone.Value = 5000;
   oopic.delay = 65;
   ScoreTone.Operate = 0;
   oopic.delay = 10;
   ScoreTone.operate = 1;
   ScoreTone.Value = 30000;
   oopic.delay = 65;
   ScoreTone.Operate = 0;
   oopic.delay = 10;
 } // End of playing tone and flashing light


Return; // Return to Main routine to restart the game

} // startgame
```