Penny Slot Machine

Submitted by:
Richard Hopkins and Jen Harmel

Mechatronics and Measurement Systems
Colorado State University
December 7, 2004

Design Summary

This project is a billiards-themed slot machine (see Figure 1). This slot machine is activated by a penny and then a button or handle. This spins three reels, each of which have eight numbers on them, 1 through 8, in the form of billiard balls. As the reels spin, music is played, accompanied by flashing LED's. If these three reels stop on a winning combination of numbers, then the machine will pay out coins to the user, accompanied by a chime sound for each coin paid out to the winner. During the slot machine operation, messages are displayed on an LCD screen, instructing the user on what to do and what is happening. A video demonstration of the slot machine can be viewed at the following link:

http://www.engr.colostate.edu/~dga/video_demos/mechatronics/PIC_student_projects/PIC_slot_machine.wmv

Other student projects can be viewed at the following link:

http://www.engr.colostate.edu/~dga/video_demos/mechatronics/index.html#PIC_PROJECTS

The heart of this device is the circuit, which controls all of the electrical devices contained inside the slot machine (Figure 5). When plugged in, the slot machine prompts the user to 'insert coin' through a LCD display. Once a coin is inserted, a photo cell will detect it and send a signal to the master microcontroller telling it that a coin was inserted. The LCD will now read 'spin reels' and the user can then activate the stepper motors to spin the reels by either pressing the spin button or by pulling the handle.

Once the button or handle is used the master microcontroller will detect this and tell the three stepper motor microcontrollers to initiate their programs (found in Appendix A), along with starting the music that is output to a speaker, and a sequence of flashing LED lights. The stepper motor microcontrollers each generate a random number between 0 and 7 and use this number to spin their assigned reel from their original positions plus a specific number of positions, determined by the random number, to get to their new position. Each time, all three microcontrollers store their new position into memory to be

used the next time the slot machine is activated. Each microcontroller sends pluses to their individual stepper motor circuit which then rotates the steeper motors (Figure 7). The amount of rotation is determined by the number of pulses sent by the microcontroller. The stepper motors are each directly connected to a reel with eight numbers on them. For visual effects, the reels are spun through multiple rotations before stopping at their new position.

All three microcontrollers then report back to the main microcontroller their new position. If together the three positions produce a winning combination, the main microcontroller then sends pluses to a relay that runs a set of solenoids that pushes out the number of coins that were won, and produces a chiming sound. If 3-8's are produced, the LCD displays 'jackpot' and 10 coins are paid out. For 3 of a kind other than 3-8's 'winner' is displayed on the LCD and three coins are released. For two of a kind, the payout is 1 coin and 'credit' is displayed on the LCD. If any other set of numbers is produced, nothing is paid out and the LCD displays 'Try Again'. In all cases the program now starts over and the LCD displays 'insert coin' and the slot machine is ready to be played again.

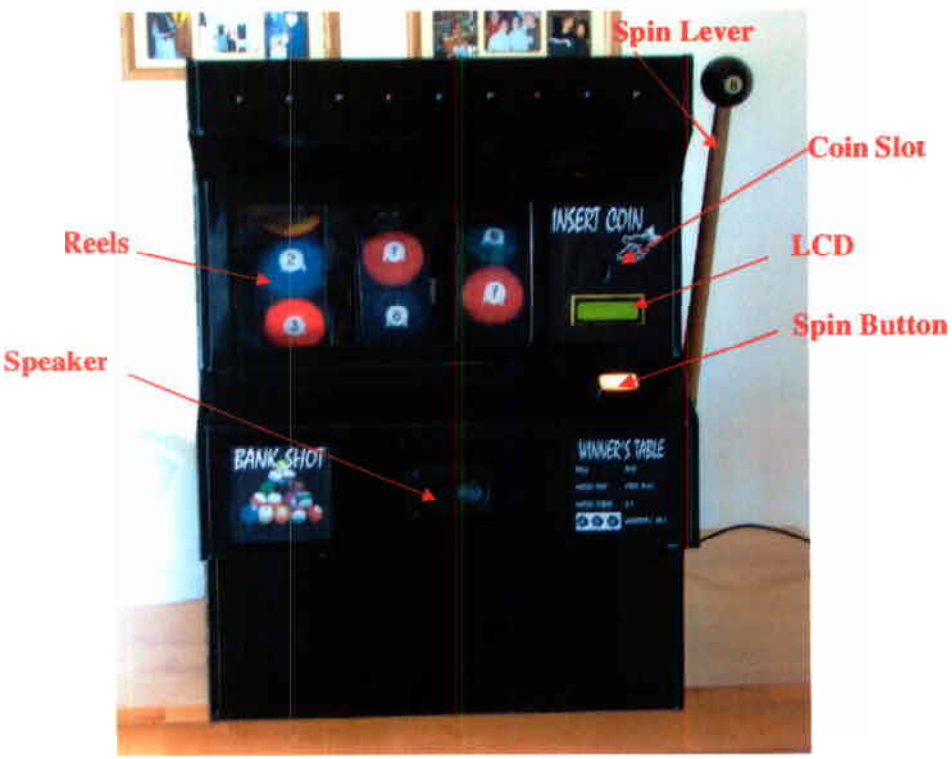

Figure 1: External View of the Slot Machine

Design Details

As indicated in the functional diagram (see Figures 2 and 3),there are two user inputs, the spin button and the spin lever, as well as one automatic sensor input, the photo cell in the coin detector, which input information to the master PIC microcontroller. The master PIC is interfaced with 3 motor PICs, which communicate with the master PIC using both serial and handshake communication, and an LED PIC, which communicates with the master PIC via handshake communication. The master PIC outputs signals to the motor PICs and the LED PIC to start their respective programs, and the master PIC also controls the LCD display and outputs pulses to the solenoid for the payout mechanism. Each motor PIC controls one stepper motor and reel, outputting pulses to the bipolar stepper motor driver to move the reel to its new location, calculated from a random number generator in the PIC software. The B motor PIC also controls the speaker, signaling it to play music as the reels spin. The LED pic controls nine LED's mounted on the top of the slot machine, scrolling with different patterns before and during the spin cycle.

The major components of the slot machine are all housed inside a sheet metal case, supported with an iron frame. The lever is a pool cue and an 8-ball, attached to an emergency break handle from a car to produce the desired ratcheting action. The reels (see Figure 7) are run by bipolar stepper motors, and are cut out of 1 ½ inch thick styrofoam, mounted on an all-thread axle. The payout mechanism (see Fig 6) consists of 1-inch PVC pipe, a solenoid, and an aluminum base. The circuit consists of the following major components:

- (5) 16F84A PIC Microcontrollers
- (3) EDE1204 Bipolar Stepper Motor Controllers
- (3) L293D Dual H-Bridges with Flyback Diodes
- (3) Bipolar Stepper Motors
- Solid State Relay
- Digital Sound Module with Amplified Speaker
- Doorbell (2 solenoids with chime)

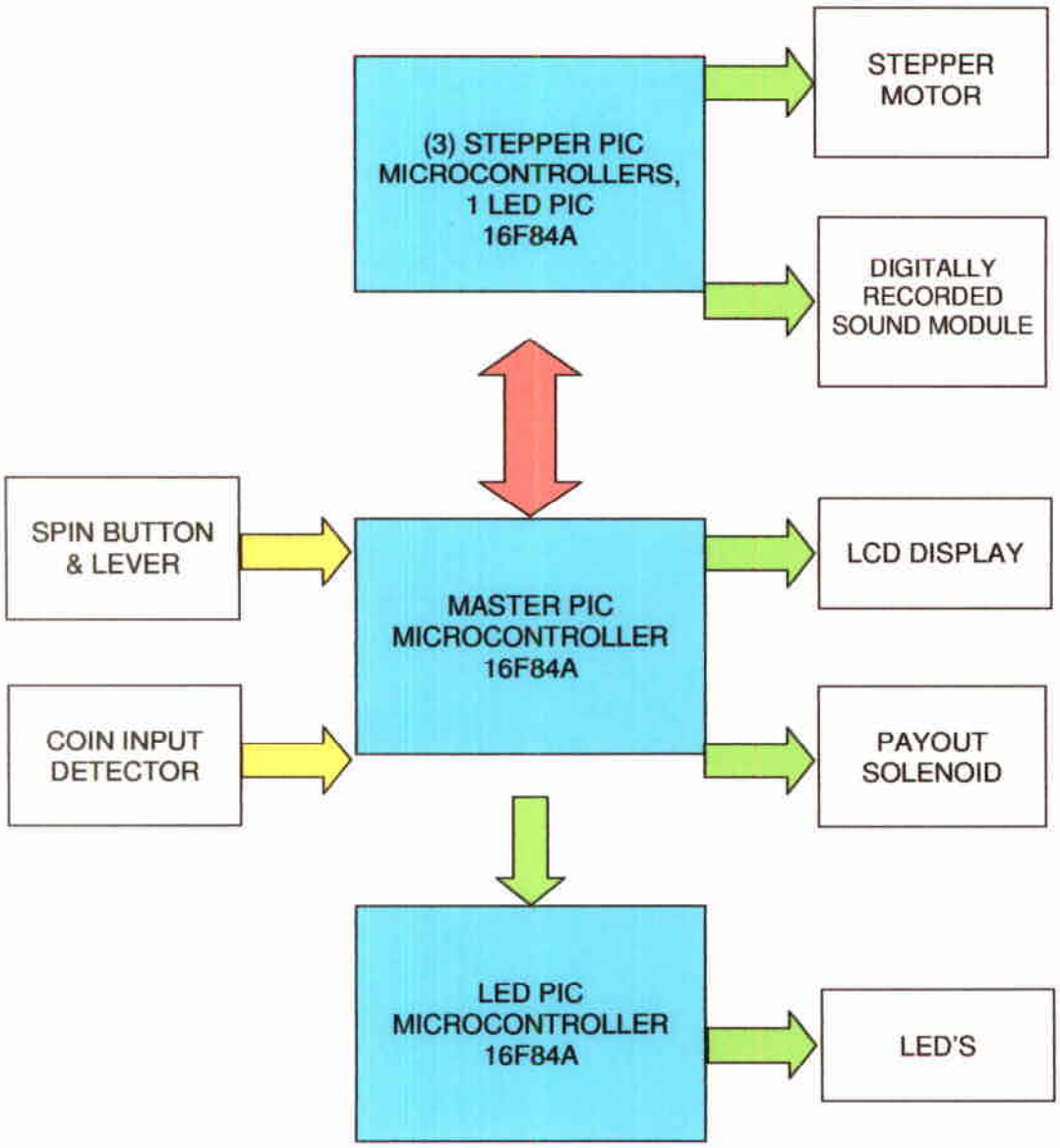

Figure 2: Functional Diagram

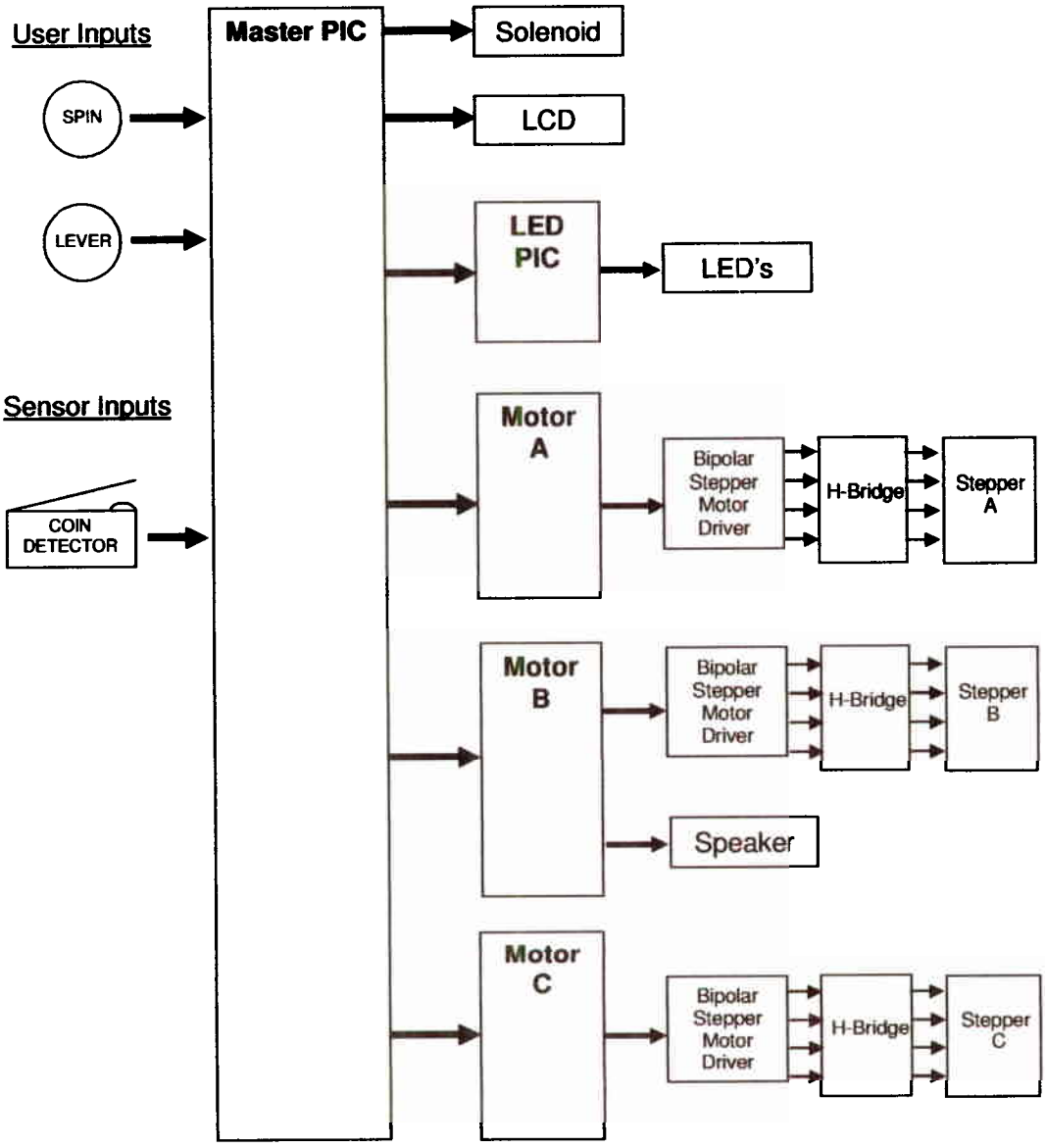

Figure 3: Functional Wiring Diagram

Device Pictures

Figure 4: Internal view of the slot machine

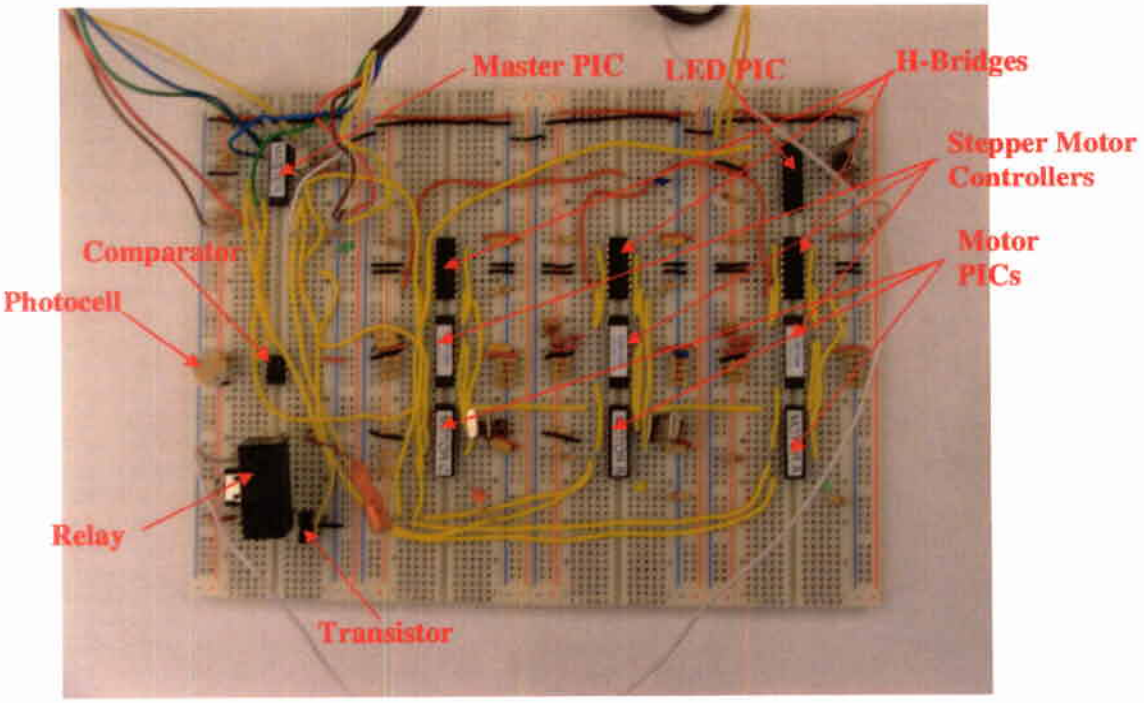

Figure 5: Circuit Board

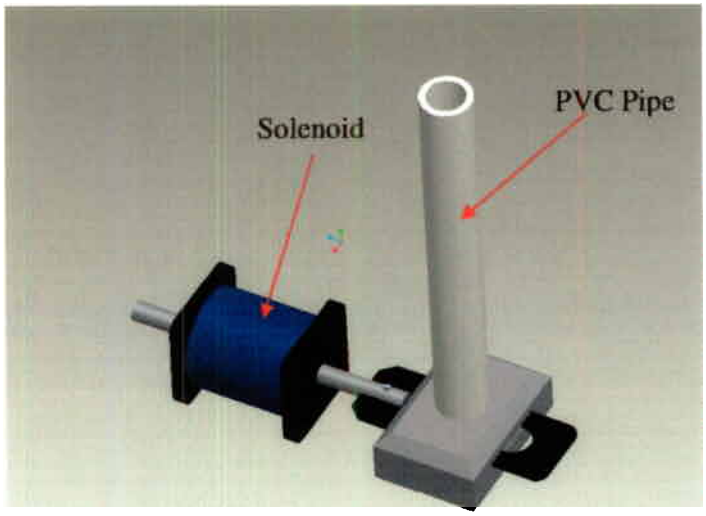

Figure 6: Payout Mechanism

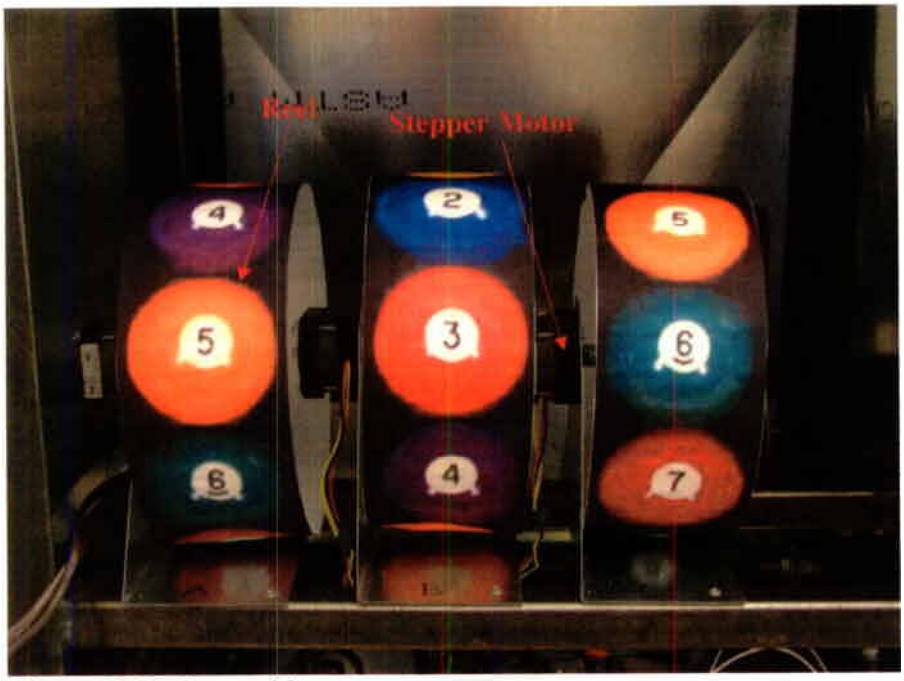

Figure 7: Reel assembly

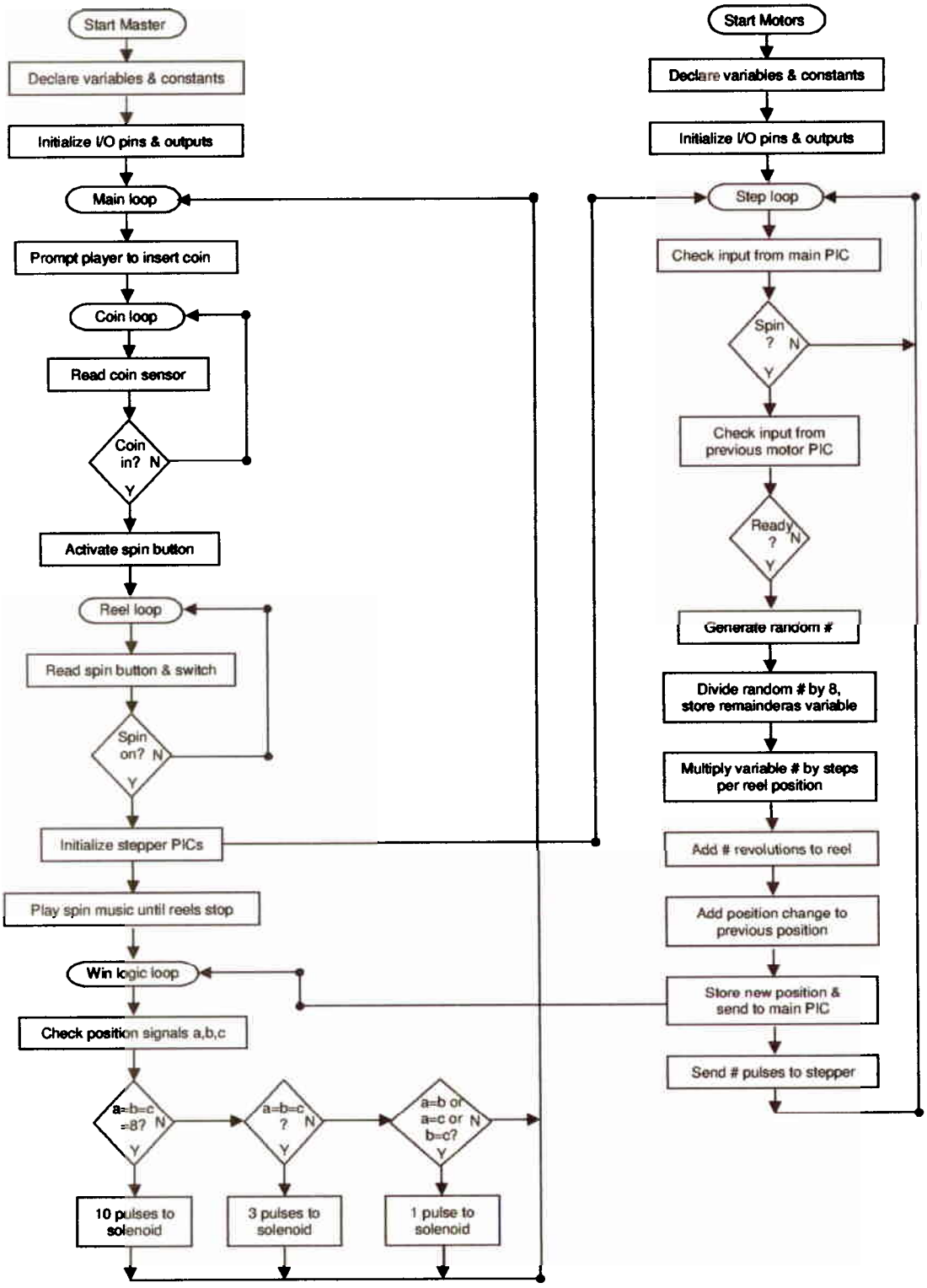

Figure 8: Software Flowchart

Table 1: Itemized Project Cost

Item	Units	Unit Price	Cost	Donation Value	Paid	
Parallel 16x2 LCD Display	1	$11.85	$11.85		$11.85	
Bipolar Stepper Motor	3	$4.95	$14.85		$14.85	
EDE 1204 Bipolar Stepper Motor Driver	3	$8.30	$24.90		$24.90	
Microchip PIC16F84A Microcontroller	5	$5.00	$25.00	$5.00	$20.00	
H-Bridge with Flyback Diode	3	$2.00	$6.00		$6.00	
Computer Power Supply	1	$2.50	$2.50	$0.00	$2.50	
5V DC Relay	1	$2.00	$2.00	$2.00	$0.00	
Digital Sound Module	1	$10.00	$10.00		$10.00	
LM2903N Comparator	1	$0.25	$0.25		$0.25	
Photocell	1	$0.89	$0.89		$0.89	
LED grab bag	1	$3.00	$3.00		$3.00	
Protoboard	**4**	**$5.00**	**$20.00**	**$5.00**	**$15.00**	
1KΩ Resistor	**25**	**$0.01**	**$0.25**	**$0.25**	**$0.00**	
10Ω Resistor	**2**	**$0.01**	**$0.02**	**$0.02**	**$0.00**	
4.7KΩ Resistor	**20**	**$0.01**	**$0.20**	**$0.20**	**$0.00**	
2KΩ Resistor	**1**	**$0.01**	**$0.01**	**$0.01**	**$0.00**	
10KΩ Resistor	**5**	**$0.01**	**$0.05**	**$0.05**	**$0.00**	
22 pF Capacitor	**10**	**$0.07**	**$0.70**	**$0.70**	**$0.00**	
0.1 µF Capacitor	**15**	**$0.08**	**$1.20**	**$1.20**	**$0.00**	
NPN Power Transistor	1	$0.37	$0.37		$0.37	
Slot Machine Spin Button	1	$1.00	$1.00		$1.00	
4 MHz Oscillator Crystal	5	$0.59	$2.95	$0.59	$2.36	
Aluminum Stock	2	$5.00	$10.00	$10.00	$0.00	
Cast Iron Pipe	1	$1.00	$1.00	$1.00	$0.00	
Styrofoam Insulation	1	$3.25	$3.25		$3.25	
Fasteners	20	$1.00	$20.00		$20.00	
8 Ball	1	$6.00	$6.00		$6.00	
Pool Cue	1	$5.00	$5.00	$5.00	$0.00	
Amplified Computer Speaker	1	$5.00	$5.00	$5.00	$0.00	
Sheet Metal	1	$50.00	$50.00	$50.00	$0.00	
Doorbell	1	$10.00	$10.00	$10.00	$0.00	
Car Emergency Break Handle	1	$2.00	$2.00	$2.00	$0.00	
All-Thread	1	$5.00	$5.00		$5.00	
Total			**$245.24**		**$147.22**	

Appendix A-Software

CODE FOR MASTER PIC

This program waits for a coin to be inserted, and then waits for the spin button to be pushed or the lever to be pulled, prompting the user to insert a coin and then to spin the reels. Then, this program sends a signal to the motor PICs to have them determine their new position and report back serially. The program then waits for a signal from motor PIC C that it is done spinning before performing the logic on the reported reel positions to determine whether the result is a winning combination. If a winning combination is reached, the program displays a message on the LCD screen and sends pulses to the solenoid to pay out the winnings. If it is not a winning combination, the program displays a message prompting the user to try again. Then, in either case, the program returns to the beginning.

```
Include "modedefs.bas"                  'include mode definitions
start Var PORTB.7                       'B7 starts motor pics
coin Var PORTB.0                        'coin input sensor
spin Var PORTB.1                        'spin button/lever input
solenoid Var PORTB.2                    'solenoid coin payout
a_motor Var PORTB.4                     'a motor serial in
b_motor Var PORTB.5                     'b motor serial in
c_motor Var PORTB.6                     'c motor serial in
a Var word                              'a position
b Var byte                              'b position
c Var byte                              'c position
i Var byte                              'counting variable

TRISB = %01110011                       'Set B7, B3, B2 as outputs, rest as inputs
LOW start : LOW solenoid : LOW PORTB.3  'Start outputs low
Pause 500

ready:
Lcdout $fe, 1                           'Clear LCD screen
Lcdout "Insert Coin"                    'Prompt Player to insert coin
While (coin==0) : Wend                  'Wait for coin to be inserted
        Goto begin                      'Go to begin subroutine
begin:

Lcdout $fe, 1                           'Clear LCD screen
Lcdout "Spin Reels"                     'Prompt player to spin reels
While (spin==0) : Wend                  'Wait for player to spin reels
High start                              'Send high signal to motors

SERIN a_motor,N300, a                   'Read position of reel A, store as a
SERIN b_motor,N300, b                   'Read position of reel B, store as b
SERIN c_motor,N300, c                   'Read position of reel C, store as c
Low start
Goto motor                              'Go to motor subroutine
```

```
motor:

While (spin == 0) : Wend                        'Wait for signal from motor to start win loop
If (a==b) AND (b==c) AND (c==8) Then            'If 3 8's, then jackpot...
        Pause 500
        LCDOUT $fe, 1
        LCDOUT "Jackpot"
        For i=1 TO 10                           'Pay out 10 coins
                HIGH solenoid
                Pause 500
                LOW solenoid
                Pause 500
        NEXT i
        Pause 1000
        Goto ready                              'Go back to start of program
Else
        If (a==b) and (b==c) Then               'If 3 of a kind, then winner...
        Pause 500
        LCDOUT $fe, 1
        LCDOUT "Winner"
        For i=1 TO 3                            'Payout is 3 coins
                HIGH solenoid
                Pause 500
                LOW solenoid
                Pause 500
        NEXT i
        Pause 5000
        Goto ready                              'Go back to start of program
        Else
                If (a==b) or (a==c) or (b==c) Then   'If 2 of a kind, then credit...
                Pause 500
                LCDOUT $fe, 1
                LCDOUT "Credit"
                HIGH solenoid                   'Payout is 1 coin
                Pause 500
                LOW solenoid
                Else                            'If not a winner, try again

Pause 500
                LCDOUT $fe, 1
                LCDOUT "Try Again"
                Pause 5000
                goto ready
Endif: Endif: Endif

goto ready                                      'Go back to start of program
End
```

PIC CODE FOR STEPPER A

This program waits for a signal from the master PIC to run its reel. When it receives a high signal from the master PIC, the program turns on the enable on the bipolar stepper motor driver, and then performs mathematical operations to determine how far to spin the reel. These operations include the generation of a random number, followed by division by 8, using the remainder value to determine how many reel positions the motor needs to move. The program then determines what the reel's new position will be by taking the original stored position and adding the change in position. Then, the program will report this position back to the master PIC and send pulses to the stepper motor controller in order to move the reel 12 revolutions plus the change in position.

```
Include "modedefs.bas"                                  'Include mode definitions
bser Var PORTA.2                                         'port A2 to b motor pic
motor Var PORTB.0                                       'port B0 to motor
switch Var PORTB.1                                      'port B1 from main PIC
pic Var     PORTB.3                                     'port B3 to main PIC
LED Var PORTB.4                                         'port B4 to LED
current Var PORTB.6                                     'port B6 to motor enable
num Var word                                            'seed number
i Var word                                              'counting variable
steps Var word                                          'number of steps to move
move Var word                                           'change in position on reel
original Var word                                       'reel position prior to move
pos Var word                                            'current reel position
num=23                                                  'seed number is 23

DATA @5, 8                                              'store original pos (8) in EEPROM at pos
5
TRISB = %00001010                                       'set B1 and B3 as inputs, rest as outputs
LOW motor : LOW pic : LOW LED : LOW current : LOW PORTB.7  'start outputs low
loop:
While (switch == 0) : Wend                              'if high signal from pic
        HIGH current                                    'turn on enable for more torque
        Random num                                      'generate a random #
        move = num // 8                                 'divide by 8, move remainder positions
        steps = (move * 25) + 2400                      'move 12 revs plus remainder positions
        READ 5, original                                'read current position from EEPROM
        If ((original + move) >=8) Then                 'determine new reel position
                pos = ((original + move) - 8)
        Else
                pos = (original + move)
        Endif
        SEROUT pic, N300, [#pos]                        'report new position to master PIC
        HIGH bser : HIGH LED                            'Signal b stepper to serout to main PIC
        Pause 50

        LOW bser
        WRITE 5, pos                                    'write position on EEPROM
        For i=1 TO steps                                'send pulses to move desired steps
                HIGH motor
                pause 1
                LOW motor
                pause 1
        NEXT i
        Pause 100                                       'pause .1 sec
        LOW current                                     'turn off enable to decrease current
End
```

PIC CODE FOR STEPPER B

This program waits for a signal from the master PIC to run its reel. When it receives a high signal from the master PIC, the program then waits for a signal from motor PIC A, and then turns on the enable on the bipolar stepper motor driver, and turns on the speaker to play music as the reels spin. Then, the porgram performs mathematical operations to determine how far to spin the reel. These operations include the generation of a random number, followed by division by 8, using the remainder value to determine how many reel positions the motor needs to move. The program then determines what the reel's new position will be by taking the original stored position and adding the change in position. Then, the program will report this position back to the master PIC and send pulses to the stepper motor controller in order to move the reel 12 revolutions plus the change in position.

```
Include "modedefs.bas"                                          'Include mode definitions
bser Var PORTA.1                                                'Port A1 from A stepper
cser Var PORTA.2                                                'Port A2 to C stepper
motor Var PORTB.0                                               'Port B0 to motor
switch Var PORTB.1                                              'Port B1 input from main pic
pic Var PORTB.3                                                 'Port B3 output to main pic
music Var PORTB.4                                               'Port B4 to speaker
LED Var PORTB.5                                                 'Port B5 to LED
current Var PORTB.6                                             'Port B6 to motor enable
num Var word                                                    'Seed number
i Var word                                                      'Counting variable
steps Var word                                                  'Number of steps to move
move Var word                                                   'Change in reel position
original Var word                                               'Reel position prior to move
pos Var word                                                    'Current reel position
num=312                                                         'Seed number is 312

DATA @5, 8                                                      'Store original pos(8) in EEPROM at pos5
TRISB = %00001010                                               'Make pins B1 and B3 inputs, rest outputs
LOW motor : LOW pic : HIGH music : HIGH LED : LOW current : LOW PORTB.7   'Start outputs off
loop:
While (switch == 0) : Wend                                      'Wait  for high signal from master PIC
        While (bser == 0) : Wend                                'Wait for high signal from a stepper PIC
        HIGH current                                            'Enable the motor for more torque
        HIGH music                                              'Turn on the spin music
        Random num                                              'Generate a random number
        move = num // 8                                         'Divide by 8, move remainder positions
        steps = (move * 25) + 2200                              'Move 11 revs plus remainder positions
        READ 5, original                                        'Read original position from EEPROM
        If ((original + move) >=8) Then                         'Calculate the new reel position
                pos = ((original + move) - 8)
        Else
                pos = (original + move)
        Endif
        SEROUT pic, N300, [#pos]                                'Report new position to master PIC
        HIGH cser : HIGH LED                                    'Signal c stepper to serout to main PIC
        Pause 50
        LOW cser
        WRITE 5, pos                                            'Store new position in EEPROM
        For i=1 to steps                                        'Send pulses to move desired # steps
                HIGH motor
                pause 1
                LOW motor
                pause 1
        NEXT i
        LOW current : HIGH music                                'Turn off enable to decrease current,
music off
End
```

PIC CODE FOR STEPPER C

This program waits for a signal from the master PIC to run its reel. When it receives a high signal from the master PIC, the program then waits for a signal from motor PIC A, and then turns on the enable on the bipolar stepper motor driver, and turns on the speaker to play music as the reels spin. Then, the porgram performs mathematical operations to determine how far to spin the reel. These operations include the generation of a random number, followed by division by 8, using the remainder value to determine how many reel positions the motor needs to move. The program then determines what the reel's new position will be by taking the original stored position and adding the change in position. Then, the program will report this position back to the master PIC and send pulses to the stepper motor controller in order to move the reel 12 revolutions plus the change in position.

```
Include "modedefs.bas"                                  'Include mode definitions
cser Var PORTA.1                                        'Port A1 from stepper B
win_time Var PORTA.2                                    'Port A2 to master PIC
switch Var PORTB.1                                      'Port B1 from master PIC
pic Var PORTB.3                                         'Port B3 to master PIC
LED Var PORTB.4                                         'Port B4 to LED
current Var PORTB.6                                     'Port B6 to motor enable
num Var word                                            'Seed number
i Var word                                              'Counting variable
steps Var word                                          'Number of steps to move
move Var word                                           'Change in reel position
original Var word                                       'Reel position prior to move
pos Var word                                            'Current reel position
num=187                                                 'Seed number is 187

DATA @5, 8                                              'Store original pos in EEPROM at pos 5
TRISB = %00001010                                       'Set B1 and B3 as inputs, rest as outputs
LOW motor : LOW pic : LOW LED : LOW current : LOW win_time   'Start outputs low

loop:
While (switch == 0) : Wend                              'Wait for high signal from master PIC
        HIGH current                                    'Turn on enable for more torque
        Random num                                      'Generate a random #
        move = num // 8                                 'Divide by 8, move remainder steps
        steps = (move * 25) + 2400                      'Move 12 revs plus remainder positions
        READ 5, original                                'Read original position from EEPROM
        If ((original + move) >= 8) Then                'Determine new reel position
                pos = ((original + move) - 8)
        Else
                pos = (original + move)
                original = pos
        Endif
        SEROUT pic, N300, [#pos]                        'Report new position to main pic
        HIGH LED
        WRITE 5, pos                                    'Write current position on EEPROM
        For i=1 TO steps                                'Send pulses to move desired steps
                HIGH motor
                pause 1
                LOW motor
                pause 1
        NEXT i

        HIGH win_time
        Pause 100
        LOW win_time :  LOW current : LOW LED           'Turn off enable to decrease current, LED off
        goto loop
End
```

PIC CODE FOR LED BLINK PIC

This program blinks the LED's in a walking pattern while the reels are not spinning. When the program recieves a signal that the reels are going to spin, the program then blinks all of the LED's simultaneously.

```
Spin Var PORTA.2                    'Port A2 from main PIC to signal spin
Red Var PORTB.3                     'Red LEDs on Port B3
Green Var PORTB.4                   'Green LEDs on Port B4
Yellow Var PORTB.5                  'Yellow LEDs on Port B5

HIGH Red : HIGH Green : HIGH Yellow 'Start with LEDs off (wired to be reverse)

Loop:
While (spin==0)                     'If not spinning, then...
        LOW Red                     'Turn on red LEDs
        Pause 300                   'Wait 0.3 sec
        HIGH Red                    'Turn off red LEDs
        LOW Yellow                  'Turn on yellow LEDs
        Pause 300                   'Wait 0.3 sec
        HIGH Yellow                 'Turn off yellow LEDs
        LOW Green                   'Turn on green LEDs
        Pause 300                   'Wait 0.3 sec
        HIGH Green                  'Turn off green LEDs
        Goto loop                   'Repeat forever
Wend                                'If reels are spinning, then...
        LOW Red                     'Turn on red LEDs
        LOW Yellow                  'Turn on yellow LEDs
        LOW Green                   'Turn on green LEDs
        Pause 300                   'Wait 0.3 sec
        HIGH Red                    'Turn off red LEDs
        HIGH Yellow                 'Turn off yellow LEDs
        HIGH Green                  'Turn off green LEDs
        Goto loop                   'Repeat forever
End
```