Carlo Pascoe

University of Florida
Department of Electrical Engineering
EEL 5666
Intelligent Machines Design Laboratory

Final Report: Automated Poker Dealer

Table of Contents

Abstract

This automated poker dealing robot will deal 8 different poker games to up to 8 different players in accordance with the rules of the game selected. The 8 games I have selected are Texas Hold'Em, Omaha, 5 Card Draw, 7 Card Stud, 7's Take All, Super Hold'Em, Cool Hand Luke, and Triple Flop Hold'Em.

Executive Summary

This automated poker dealing robot deals 8 different poker games to up to 8 different players around a poker table. The 8 games are Texas Hold'Em, Omaha, 5 Card Draw, 7 Card Stud, 7's Take All, Super Hold'Em, Cool Hand Luke, and Triple Flop Hold'Em. There are 8 player and 3 game select switches. These switches can be continuously adjusted until a deck of cards is placed in the deck loader, activating the weight sensors, and initiating game play. Based on the values of the input switches the robot will then begin to deal cards in accordance to the rules of the game selected. Most games require the robot to move forward and deal cards to each individual player. The robot does this by sending a PWM signal to a motor driver causing the motors to propel the robot forward, sending a PWM to a mega servo causing the rotating platform to rotate the position of the player waiting for cards, and activating a number of output ports for small periods of time that causes current to travel through MOSFET's connected to several deck manipulation motors. When the robot moves forward it watches for objects in its way with an IR proximity sensor and bump switches. An object in the way causes the robot to stop and wait for it to be removed. During periods of time that the players have to bet the robot will wait. When the players are ready for game play to continue they will communicate this through the special voice recognition sensor by shouting the command "continue." Speaking the word "continue" causes the values sent to an input port to change indicating that "continue" has been spoken. In games that require specific inputs from the different players like 5 card draw, the players will speak these commands into the headset so that the voice recognition circuit can recognize them as well. The robot will continue in this fashion until the game is over at which time the robot will eject the rest of the deck and go into a wait state until the players are ready to begin again.

Introduction

My friends and I play poker all the time but there is this one kid who we do not let deal because he is notorious for dealing from the bottom of the deck. He often comments that it is unfair that everyone else is allowed to deal while only his dealing is held in scrutiny. An automated poker dealer would take the doubt out of human dealing, alleviating this problem once and for all. The rest of this report discusses how such a robot would be constructed.

Integrated System

When turned on the robot would do nothing, waiting for inputs. 8 switches corresponding to 8 player positions would need to be set to determine the number of players and there positions on the table. 8 LED's pointing to the player position will light up when the corresponding switch is asserted. Another set of 3 switches corresponding to 8 different types of poker games would need to be set to specify which game you want to play. The name of the game selected would be displayed on the LCD. If the switches are set to invalid inputs (like you can't have 8 players play 5 card draw with 1 deck of cards because there is too much of a chance that the deck will run out) the LCD will say that you have entered invalid inputs. The robot would continue waiting until a deck of cards is placed in the deck loader. A weight sensor will indicate that a deck of cards has been placed in the loader initiating a shuffling sequence. When the shuffling is complete the robot will begin dealing the cards to the positions selected by the 8 input switches in accordance with the game selected. If a hidden switch is asserted the robot will still deal the cards in accordance with the game selected but will deal from a rigged deck that is hidden inside the robot. When the Robot moves, IR and bump sensors detect if obstacles like poker chips or beer bottles are in the way, waiting for the object to be removed if an object is detected. In games that require human action like betting before more cards are dealt (or in the case of games like 5 card draw, specifying how many more cards you want) a special voice recognition sensor will convey to the robot that game play is ready to continue. When the round is complete and a winner has been determined the robot will eject the rest of the cards and wait for the process to begin again.

The brains of this robot will be an ATmega128 MCU contained in a Mavrik-IB development board from BDMicro. The microcontroller will have to control:

- 2 drive motors
- 1 servo for platform rotation
- 2 motors for shuffling
- 1 motor for loading the regular cards into the dealer
- 1 motor for loading the rigged cards into the dealer
- 1 motor for dealing to people
- 1 servo for weighting down the cards to be dealt
- 1 servo for flipping cards face-up or face-down
- 1 LCD screen

While receiving inputs from:

- 12 input switches
- 1 IR detection sensor
- 2 bump sensors
- 2 weight sensors
- 1 voice recognition sensor

Mobile Platform

Most of the platform is constructed from a medium grade, half inch plywood. The base is a 16 inch diameter circle with two rectangular holes near the sides for the drive wheels and one hole on the back so that a 14.4V drill battery can be inserted for power. The rear of the platform extends up from the base about 6 inches and is basically a crescent with the radius of the outer semi-circle being 16 inches and the radius of the inner semi-circle being 14 inches. This rear part of the platform houses most of the electronics on the inside and the 12 input switches on the top. The front part of the platform is a 14 inch diameter circle that connects the base with a lazy Susan allowing the circle to rotate. This 14 inch diameter circle is the base for the dealing mechanism. The top is where the

cards are placed. The cards are dealt through a hole at the front of the robot that is positioned under the LCD screen. Extending from the base of the robot are two little arm like pieces that house the bump switches. The IR detector is located between these two arms. For aesthetic reasons all of the visible plywood is stained with a dark walnut stain and then covered with a coat of polyurethane. The dark walnut stain was selected to match the stain of the beer rail on the poker table that I have also built. Several T-Tech'ed pieces of Balsa wood are placed around my robot to cover unsightly cuts on the plywood that could not be done accurately with the tools I have.

Actuation

As mentioned before the Robot will need to control:

- 2 drive motors
- 1 servo for platform rotation
- 2 motors for shuffling
- 1 motor for loading the regular cards into the dealer
- 1 motor for loading the rigged cards into the dealer
- 1 motor for dealing to people
- 1 servo for weighting down the cards to be dealt
- 1 servo for flipping cards face-up or face-down
- 1 LCD screen

These are discussed in more detail below.

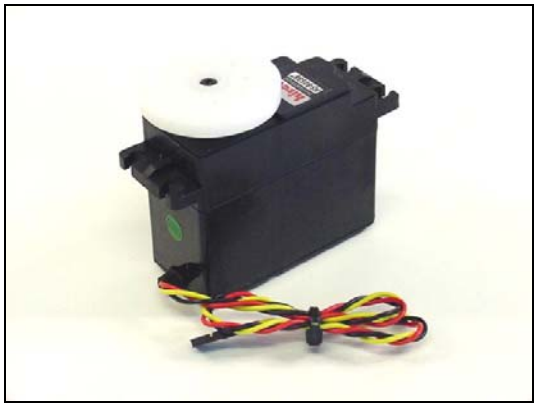
2 Drive Motors

Vendor: Lynxmotion, Inc.
PO Box 818
Pekin, IL 61555-0818
Tel: 309-382-1816 (Sales)
Tel: 309-382-2760 (Support)
Fax: 309-382-1254
E-m: sales@lynxmotion.com
E-m: tech@lynxmotion.com
Web: http://www.lynxmotion.com

Part#: PGHM-18

These 6mm OD RS-540 motors are 12VDC with a 369:1 gear reduction causing 13RPM and a stall torque of 4382.22 oz-in. They weigh 12.42 oz. These motors are placed at the bottom of the robot and connected to 5 inch diameter wheels that, with the aid of two casters located near the rear of the robot, allow the robot to move around the table. In order to control these motors, a pair of PWM signals are sent to a standard motor driver. The driving algorithm can be viewed in the code provided in the appendices.

1 Platform Rotation Servo



Vendor: Lynxmotion, Inc.
PO Box 818
Pekin, IL 61555-0818
Tel: 309-382-1816 (Sales)
Tel: 309-382-2760 (Support)
Fax: 309-382-1254
E-m: sales@lynxmotion.com
E-m: tech@lynxmotion.com
Web: http://www.lynxmotion.com

Part#: S805BB

The robot utilizes a HS-805BB Mega Servo to rotate the 14 inch diameter platform that houses the dealing mechanism. This servo provides 343 oz.-inches of torque which is more than enough to rotate the platform. This servo was tested by varying the PWM signal and determining the pulse size needed to rotate the platform to various positions. The positions and corresponding pulse sizes needed for the dealing platform are listed in the table below.

Position	**Pulse Size (ms)**
Player1	1.0
Player2	1.14
Player3	1.29
Player4	1.43
Player5	1.57
Player6	1.72
Player7	1.85
Player8	2.0
Center	1.5
Burn1	1.27
Flop1	1.34
Flop2	1.4
Flop3	1.47
Burn2	1.53
Turn	1.60
Burn3	1.66
River	1.73
SARiver	1.79
Eject	2.0

How these values are used can be viewed in the code provided in the appendices.

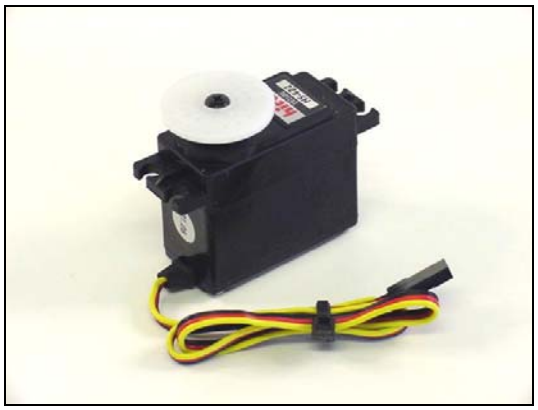
5 small DC motors

Vendor: various household items

Part#: unknown

These are simple DC motors used in unison to manipulate a deck of cards. Two motors are used to simulate shuffling. Half the deck is placed on a platform with a motor underneath it while the other half is placed on an identical platform right across from it. When turned on the motors move the cards from the two platforms into a single chamber located between the two platforms. In order to move one card at a time a 0.4 second pulse from an output port is sent to the gate of an N channel MOSFET which allows a simple power ground circuit to complete, activating the motor for a short period of time— just enough time to allow one and only one card to pass into the center chamber. The center chamber is located over another platform just like the other two recently mentioned. Across from this platform is another platform that houses a rigged deck of cards. These two platforms launch cards into another center chamber that is located above another platform that finally launches cards at the players. The platform that houses the rigged deck of cards only moves cards into the center if the hidden cheat switch is asserted. The deal() method provided with the code in the appendices shows how these motors are activated.

2 standard servos



Vendor: Lynxmotion, Inc.
PO Box 818
Pekin, IL 61555-0818
Tel: 309-382-1816 (Sales)
Tel: 309-382-2760 (Support)
Fax: 309-382-1254
E-m: sales@lynxmotion.com
E-m: tech@lynxmotion.com
Web: http://www.lynxmotion.com

Part#: S422

The robot uses two HS-422 Standard Servos, one for flipping cards face up or face down and another for weighing down cards in the dealing mechanism so that the rubber grabbers can have a better grip. The servos only provide 57 oz.-inches of torque which is plenty for the applications they are being used for. These servos only need two positions to be effective— full on at 2.0ms and full off at 1.0ms. The card flipping servo rotates a piece of wood into the path of falling cards causing it to flip over. The weigh down servo lowers a weight onto a card that is ready to be dealt. How these servos are used can be viewed in the code provided in the appendices.

1 LCD

Vendor: Lab

Part#: unknown

This is the LCD from EEL4744. Though I guess not actually actuation, it is an output of the system like all of the other items in the actuation section. The LCD is used to deliver messages to players throughout the game.

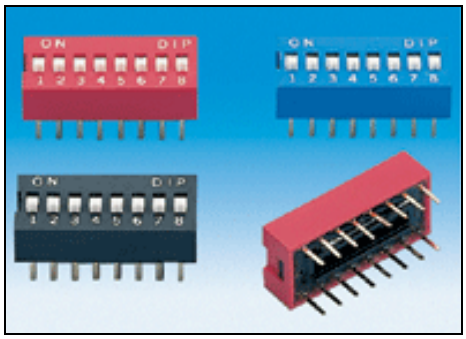
Sensors

The Automated Poker Dealer is able to perform its dealing tasks through the aid of several sensors. The first type of sensor is 12 standard SPST switches-- 8 switches corresponding to 8 players and their positions on the table, 3 switches corresponding to 8 different types of poker games, and a single switch that indicates that a rigged deck of cards hidden inside the robot should be put in play. These switches are active before game play begins and become inactive when the next sensor is activated, the next sensor being a set of weight sensors that determine when a deck of cards has been placed in the card loader. Once the deck has been loaded into the robot game play commences and only stops again for four different reasons:

1. The round is complete
2. The IRPD-01 Proximity Sensor has determined that an obstacle impedes the robots movement
3. 1 of 2 standard bump switches has determined that an obstacle impedes the robots movement
4. Some aspect of the current game requires human input communicated through the SR-07 voice recognition system

These sensors are discussed in detail below:

Sensor 1: SPST Switches

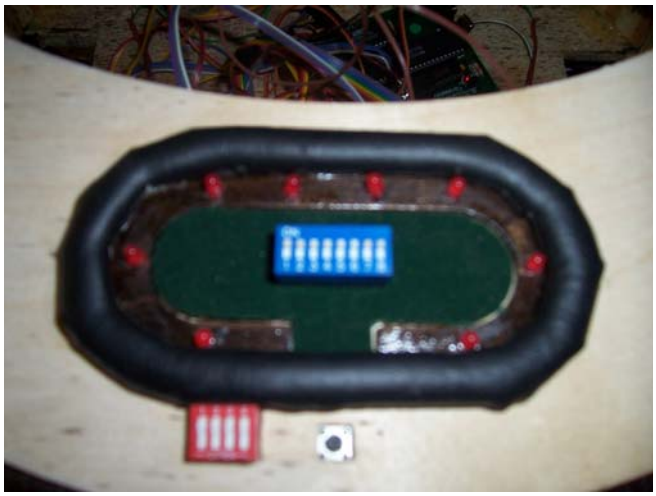


Vendor: Lab

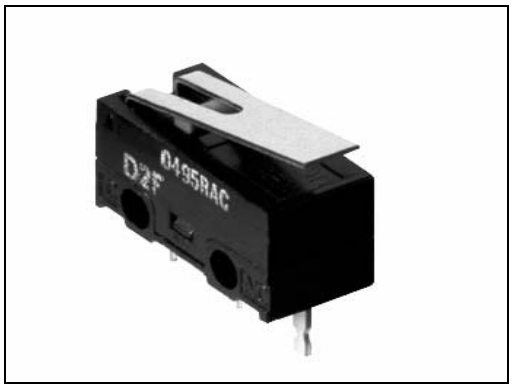
Part#: unknown

These are your standard SPST switches. One set of switches is an 8 switch DIP pack and the other is a 4 switch DIP pack for a total of 12 SPST switches. As stated above these switches will allow the processor to determine the number of players and their positions on the table (8s DIP pack), what game the users want to play, and if the user wants to use the rigged deck of cards (4s DIP pack). These switches will be connected with a pull-up resistor and signal line on one terminal and GRND on the other terminal so that +5V and GRND will be on the signal line when the switch is open or closed respectively. I wired up 3 different switches and with a volt meter proceeded to test the reliability of these switches. I performed 25 tests on each switch and recorded 100% reliability on each switch.

Images from: www.emulation.com/catalog/



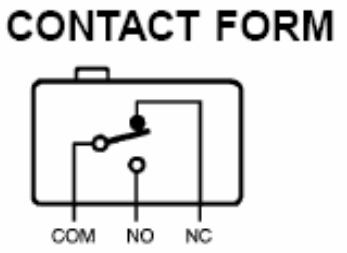
Sensor 2: Weight Sensors



Vendor: Spark Fun Electronics
2500 Central Ave.
Suite Q
Boulder, CO 80301
Tel: 1-303-284-0979
Fax: 1-303-443-0048
E-m: fun@sparkfun.com
Web: www.sparkfun.com/

Part#: D2F-L

Since the deck of cards is either placed in the loader or it's not I used a Subminiature Snap Action Switch to act as a digital weight sensor rather than using its analog counterpart. Two of these switches, one for each card slot, will be used to determine if a cut deck of cards has been placed in the loader. The microcontroller will wait until it receives signals from both of these switches before it commences game play. This switch has three terminals as depicted below (COM, NC, NO).

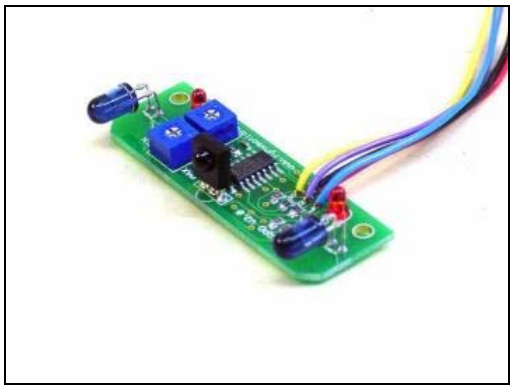


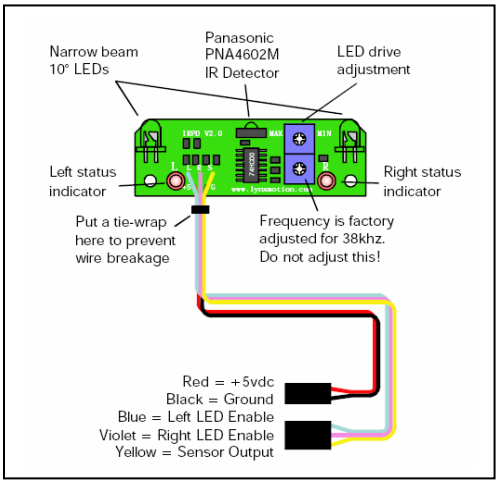
The signal line that is sent to the microcontroller will be connected to the COM terminal with a resister in series, the NC terminal will be connected to GRND, and the NO terminal will be connected to VCC. This connection will ensure that when the cards are placed in the loader +5V will be sent to the microcontroller. I wired up the 2 sensors and

with a volt meter proceeded to test the reliability of these weight sensors. I performed 25 tests on each and recorded 100% reliability.

Images From: www.sparkfun.com

Sensor 3: IRPD-01 Proximity Sensor

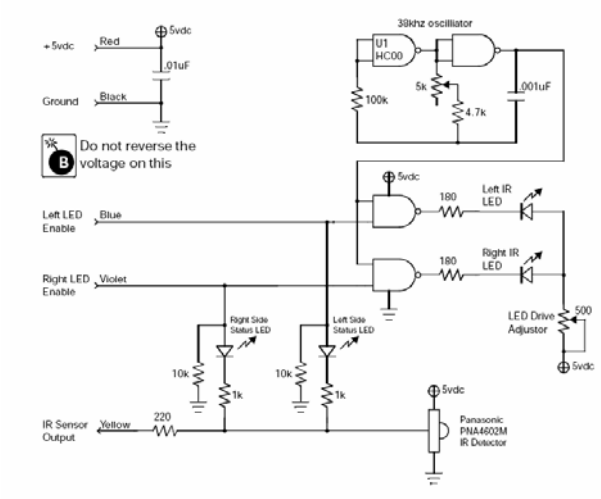




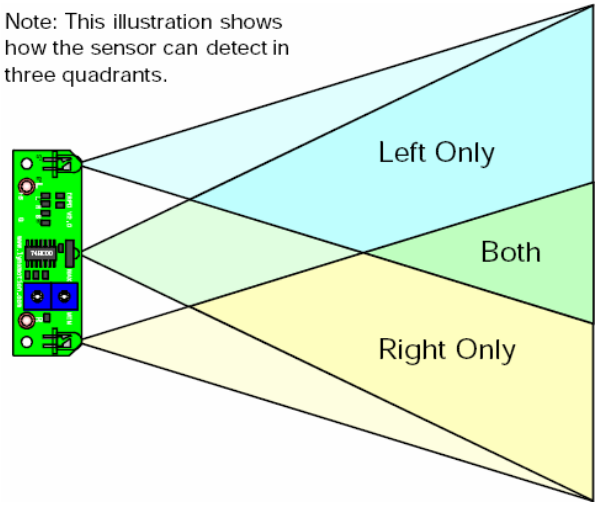
Vendor: Lynxmotion, Inc.
PO Box 818
Pekin, IL 61555-0818
Tel: 309-382-1816 (Sales)
Tel: 309-382-2760 (Support)
Fax: 309-382-1254
E-m: sales@lynxmotion.com
E-m: tech@lynxmotion.com
Web: http://www.lynxmotion.com

Part#: IRPD-01

This sensor detects small objects that pass in front of it. Through the use of 2 LED's transmitting at 38kHz and a Panasonic PNA4602M IR Detector, the sensor is able to detect when the pattern of reflecting light changes thereby deducing that some object in front of the sensor is causing the altered pattern. This device has many other components that make it easy to interface with a microcontroller— when wired up as the image above depicts one only needs 1 input pin to determine if an object is in front of the device. The other components are connected as the image below depicts.



Though the device can detect objects in 3 different cones as the picture below shows, there is only 1 sensor output signal that goes low when an object is detected and is high otherwise. Alternating the LED enable signals will allow the microcontroller to distinguish which cone the object is in.



The truth table below shows the appropriate IR detector output given the various inputs.

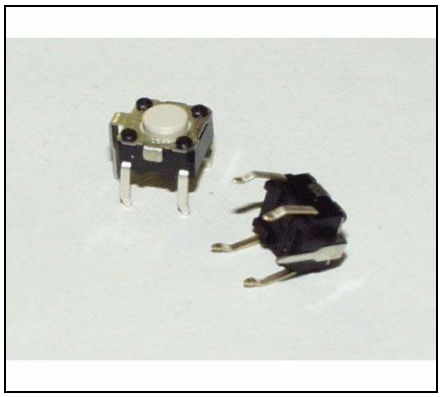
IRPD Truth Table

IRPD I/O pins / LEDs	Not Allowed	With obstacle			Without obstacle		
Left enable (blue)	~~High~~	Low	High	Low	Low	High	Low
Right enable (violet)	~~High~~	Low	Low	High	Low	Low	High
Left status LED	????	Off	On	Off	Off	Off	Off
Right status LED	????	Off	Off	On	Off	Off	Off
IR detector (yellow)	????	High	Low	Low	High	High	High

My testing shows that this sensor will reliably detect object that are within 5 to 7 inches of the IR Detector. This relatively large range is due to that fact that dark and light colors reflect light differently. The sensor will detect a white object from about 7 inches and a black object from about 5 inches.

Images From: www.lynxmotion.com

Sensor 4: Bump Switches



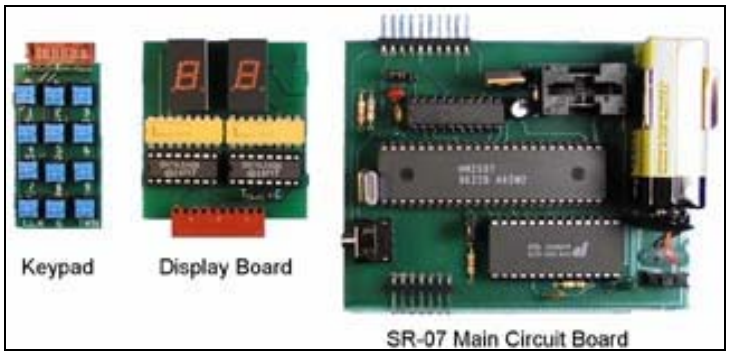
Vendor: Lab

Part#: unknown

Four standard bump switches are used as a fail safe when the IRPD-01 Proximity Sensor does not detect an object. The object will have to come in contact with a wooden strip that closes the bump switch indicating to the processor that the robot has hit something. When a bump switch is activated the robot immediately stops and waits for the object that has triggered the bump switch to be removed. These bump switches are wired up just like the SPST switches from above. I wired up 3 different bump switches and with a volt meter proceeded to test their reliability. I performed 25 tests on each and recorded 100% reliability.

Images From: www.sparkfun.com

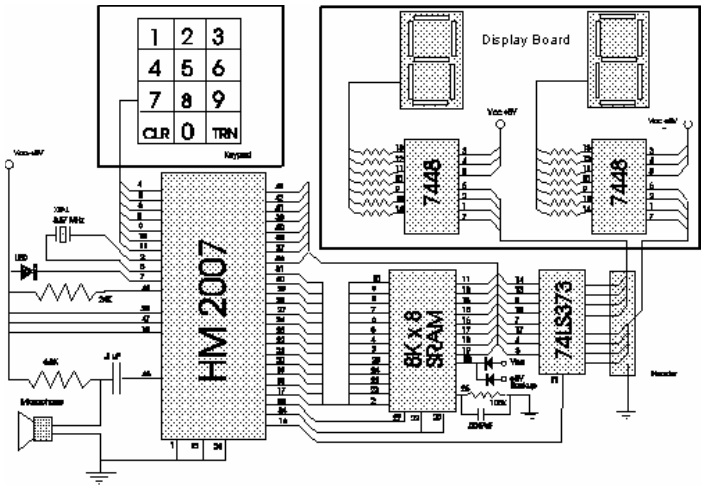
Sensor 5: SR-07 Voice Recognition System



Vendor: Images Scientific Instruments INC.
109 Woods of Arden Road
Staten Island NY 10312
Tel: (718) 966-3694
Fax: (718) 966-3695
E-M: Imagesco@bellatlantic.net

Part#: SR-07

The SR-07 voice recognition system is a standalone development board that allows a user to train up to 40 different words then indicate through a BCD output signal that a word has been recognized. This system works by utilizing a HM2007 chip which is a speech recognition chip that provides the options of recognizing either 40 .96 second words or 20 1.92 second words. When turned on, the HM2007 checks the static 8K X 8 static RAM for instructions. When the board is ready for use it will output a "00" BCD signal. Regardless if there are any words stored in memory, you can train a new word by entering a number 1-40 on the keypad then pressing the train button. The circuit will then record the next word you say and then blink an LED to indicate that the word has been recorded. When recording the board will output a "55" when a word is to long or a "66" when a word is to short, prompting the user to try and train the word again. Once words are stored in the device, it is continually monitoring speech for a potential match to a word. When a mach is found the board will output the number of the word that it matched. If there is not a match to words spoken the board will output a "77." The way this circuit works makes it very easy to interface with a microcontroller. The jobs of listening and recognition don't occupy any microcontroller clock cycles. The output of the board can be continually fed into an input port on the mavrik board and only worry about the data when it needs to. A diagram of the entire SR-07 voice recognition system is provided below.



In order to test this circuit I trained the device with the different words I will need my robot to recognize:

- "Continue"- needed when the players want to convey that game play is ready to continue
- "Single"- needed when a player needs 1 card
- "apair"- needed when a player needs 2 cards
- "three"- needed when a player needs 3 cards
- "four"- needed when a player needs 4 cards
- "misdeal"- needed when there was a misdeal and another card needs to be dealt
- "finish"- needed to signal that the game is over

The table below shows the results of my tests

Word	#of success out of 50 (no noise)	Percentage	#of success out of 50 (noise)	Percentage
Continue	47	94	44	88
Single	47	94	46	92
Apair	45	90	42	84
Three	41	82	37	74
Four	38	76	33	66
Misdeal	46	92	44	88
Finish	47	94	45	90

As you can see the results are satisfactory for multi-syllable words but are not so great for one syllable words.

Images From: www. imagesco.com/catalog/

Behaviors

When the number of players, there position, and the game has been selected the robot will wait till someone places a deck of cards into the deck shuffler. The deck will trigger a wait sensor that will initiate a shuffle. The shuffled cards will be loaded into the dealing mechanism that deals to all the positions that were specified by the players. It does this by rotating to the position then launching a card in the air that should land relatively close to the player. The robot will then drive to the designated dealing position avoiding obstacles and then go into a wait state that allows the players to place bets. When the players are ready for game play to continue they will communicate this through the special voice recognition sensor by shouting a command like "continue." In a game that requires specific inputs from the different players like 5 card draw, the players will speak these commands into the headset. The robot will continue in this fashion until the game is over at which time the robot will eject the rest of the deck and go into a wait state until the players are ready to begin again.

Conclusion

I learned a lot during the course of this project. I came in knowing a little bit about robot construction but not really enough to start from nothing and after a period of time have a fully functioning robot. This course helped me link knowledge I already possessed with new knowledge so that I understand the whole picture. I was limited in what I could do due to the lack of my mechanical engineering skills and my final product suffered because of it. All aspects of the project performed to specification but the dealing mechanism could be more accurate and would be if a mechanical engineer built it. If I were to do the project all over again I would spend more time planning. I had to rebuild my platform several times because of simple flaws that I could have avoided if I spent more time in the beginning thinking about functionality. If I were to take the class again I would make sure and work harder at the beginning of the course so as to avoid the mad dash to get everything done in the end— my final product has suffered because I just didn't have enough time to prefect it. All in all it was a good experience.

Appendices

```
#include <mega128.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>
#include <delay.h>
#include <stdlib.h>

// Global variables
bit P1,P2,P3,P4,P5,P6,P7,P8,up;
unsigned char PN, Dealer;

// Timer 1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void){
	TCNT1H=0x00;
	TCNT1L=0x00;
	TCCR1A=0xFC;
	TCCR1C=0xE0;
	TCCR1A=0xA8;
	}

// Timer 3 overflow interrupt service routine
interrupt [TIM3_OVF] void timer3_ovf_isr(void){
	TCNT3H=0x00;
	TCNT3L=0x00;
	TCCR3A=0xF0;
	TCCR3C=0xC0;
	TCCR3A=0xA0;
	}

void invalidInput(void){
	lcd_clear();
	lcd_gotoxy(0,0);
	lcd_putsf(" Invalid Inputs");
	}

void driveForward(){
	int i;
	lcd_clear();
	lcd_gotoxy(0,0);
	lcd_putsf("Driving Forward");
	i = 0;
	OCR3AH=0x0E;
	OCR3AL=0x66;

	while(i < 4000){
		delay_ms(1);
		i = i + 1;
		if(!PIND.6 || !PINE.5){
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("Remove Obstacle");
			OCR3AH=0x0A;
			OCR3AL=0xCD;
			while(!PIND.6 || !PINE.5){ delay_ms(3000); }
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("Driving Forward");
			OCR3AH=0x0E;
			OCR3AL=0x66;
			delay_ms(100);
			}
		}
		OCR3AH=0x0A;
		OCR3AL=0xCD;
	}
```

```
void driveBackward(){
	int i;
	lcd_clear();
	lcd_gotoxy(0,0);
	lcd_putsf("Driving Backward");
	i = 0;
	OCR3AH=0x07;
	OCR3AL=0x30;

	while(i < 4000){
		delay_ms(1);
		i = i + 1;
		if(!PIND.6){
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("Remove Obstacle");
			OCR3AH=0x0A;
			OCR3AL=0xCD;
			while(!PIND.6){ delay_ms(3000); }
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("Driving Backward");
			OCR3AH=0x07;
			OCR3AL=0x30;
			delay_ms(100);
			}
		}
		OCR3AH=0x0A;
		OCR3AL=0xCD;
	}

void deal(int Position){
	int randnum;
	OCR1A=Position;
	randnum = rand();
	if(up){
		OCR1CH=0x0E;
		OCR1CL=0x66;
		}
	delay_ms(2000);
	if(randnum < 16384){
		PORTB.4=1;
		delay_ms(400);
		PORTB.4=0;
	}else{
		PORTB.3=1;
		delay_ms(400);
		PORTB.3=0;
		}
	delay_ms(2000);
	if(PIND.3){
		PORTB.2=1;
		delay_ms(400);
		PORTB.2=0;
	}else{
		PORTB.1=1;
		delay_ms(400);
		PORTB.1=0;
		}
	delay_ms(1000);
	OCR1BH=0x0E;
	OCR1BL=0x66;
	delay_ms(2000);
	PORTB.0=1;
	delay_ms(1000);
	PORTB.0=0;
	OCR1BH=0x07;
	OCR1BL=0x30;
	OCR1CH=0x07;
```

```
	OCR1CL=0x30;
	}

void drawcards(int Position){
	OCR1A=Position;
	lcd_clear();
	lcd_gotoxy(0,0);
	lcd_putsf("How Many Cards?");
	while(1){
		delay_ms(3000);
		if(!PINF.3 && PINF.2 && !PINF.1 && !PINF.0){
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("1 Card");
			deal(Position);
			break;
		}else if(!PINF.3 && !PINF.2 && PINF.1 && PINF.0){
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("2 Cards");
			deal(Position);
			deal(Position);
			break;
		}else if(!PINF.3 && !PINF.2 && PINF.1 && !PINF.0){
			lcd_clear();
			lcd_gotoxy(0,0);
			lcd_putsf("3 Cards");
			deal(Position);
			deal(Position);
			deal(Position);
			break;
			}
		}

	}

void dealOneEP(void){
	if(Dealer == 8){
		if(P1){ deal(1840); }
		if(P2){ deal(2100); }
		if(P3){ deal(2370); }
		if(P4){ deal(2635); }
		if(P5){ deal(2900); }
		if(P6){ deal(3165); }
		if(P7){ deal(3400); }
		if(P8){ deal(3690); }
		}
	if(Dealer == 1){
		if(P2){ deal(2100); }
		if(P3){ deal(2370); }
		if(P4){ deal(2635); }
		if(P5){ deal(2900); }
		if(P6){ deal(3165); }
		if(P7){ deal(3400); }
		if(P8){ deal(3690); }
		if(P1){ deal(1840); }
		}
	if(Dealer == 2){
		if(P3){ deal(2370); }
		if(P4){ deal(2635); }
		if(P5){ deal(2900); }
		if(P6){ deal(3165); }
		if(P7){ deal(3400); }
		if(P8){ deal(3690); }
		if(P1){ deal(1840); }
		if(P2){ deal(2100); }
		}
	if(Dealer == 3){
		if(P4){ deal(2635); }
		if(P5){ deal(2900); }
```

```
            if(P6){ deal(3165); }
            if(P7){ deal(3400); }
            if(P8){ deal(3690); }
            if(P1){ deal(1840); }
            if(P2){ deal(2100); }
            if(P3){ deal(2370); }
            }
        if(Dealer == 4){
            if(P5){ deal(2900); }
            if(P6){ deal(3165); }
            if(P7){ deal(3400); }
            if(P8){ deal(3690); }
            if(P1){ deal(1840); }
            if(P2){ deal(2100); }
            if(P3){ deal(2370); }
            if(P4){ deal(2635); }
            }
        if(Dealer == 5){
            if(P6){ deal(3165); }
            if(P7){ deal(3400); }
            if(P8){ deal(3690); }
            if(P1){ deal(1840); }
            if(P2){ deal(2100); }
            if(P3){ deal(2370); }
            if(P4){ deal(2635); }
            if(P5){ deal(2900); }
            }
        if(Dealer == 6){
            if(P7){ deal(3400); }
            if(P8){ deal(3690); }
            if(P1){ deal(1840); }
            if(P2){ deal(2100); }
            if(P3){ deal(2370); }
            if(P4){ deal(2635); }
            if(P5){ deal(2900); }
            if(P6){ deal(3165); }
            }
        if(Dealer == 7){
            if(P8){ deal(3690); }
            if(P1){ deal(1840); }
            if(P2){ deal(2100); }
            if(P3){ deal(2370); }
            if(P4){ deal(2635); }
            if(P5){ deal(2900); }
            if(P6){ deal(3165); }
            if(P7){ deal(3400); }
            }
        }

    void passButton(void){
    while(1){
        if(Dealer == 8){
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            }
        if(Dealer == 1){
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            }
        if(Dealer == 2){
```

```
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            }
        if(Dealer == 3){
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            }
        if(Dealer == 4){
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            }
        if(Dealer == 5){
            if(P6){ Dealer = 6; break;  }
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            }
        if(Dealer == 6){
            if(P7){ Dealer = 7; break;  }
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            }
        if(Dealer == 7){
            if(P8){ Dealer = 8; break;  }
            if(P1){ Dealer = 1; break;  }
            if(P2){ Dealer = 2; break;  }
            if(P3){ Dealer = 3; break;  }
            if(P4){ Dealer = 4; break;  }
            if(P5){ Dealer = 5; break;  }
            if(P6){ Dealer = 6; break;  }
            }
        };
        }


void Holdaha(void){
        delay_ms(1000);
        driveBackward();
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2100); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Here's The Flop");
```

```
        deal(2340);
        up=1;
        deal(2460);
        deal(2580);
        deal(2700);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2100); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("    The Turn");
        deal(2820);
        up=1;
        deal(2940);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2100); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("And the River");
        deal(3060);
        up=1;
        deal(3180);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(!PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2100); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("We Have A Winner");
        OCR1A= 3300;
        PORTB.4=1;
        PORTB.3=1;
        delay_ms(10000);
        PORTB.4=0;
        PORTB.3=0;
        delay_ms(1500);
        OCR1BH=0x0E;
        OCR1BL=0x6A;
        delay_ms(2000);
        PORTB.0=1;
        delay_ms(10000);
        PORTB.0=0;
        OCR1BH=0x07;
        OCR1BL=0x30;
        passButton();
        }


void main(void){
// Local variables
bit G1,G2,G3,G4,G5,G6,G7,G8;

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
```

```
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTA=0x00;
DDRA=0x00;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=Out Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=0 State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x80;

// Port E initialization
// Func7=Out Func6=Out Func5=In Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=1 State6=1 State5=T State4=0 State3=0 State2=0 State1=0 State0=0
PORTE=0xC0;
DDRE=0xDF;

// Port F initialization
// Func7=Out Func6=Out Func5=In Func4=Out Func3=In Func2=In Func1=In Func0=In
// State7=1 State6=1 State5=T State4=0 State3=T State2=T State1=T State0=T
PORTF=0xC0;
DDRF=0xD0;

// Port G initialization
// Func4=In Func3=In Func2=In Func1=Out Func0=Out
// State4=T State3=T State2=T State1=1 State0=1
PORTG=0x00;
DDRG=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: 1843.200 kHz
// Mode: Normal top=FFFFh
// OC1A output: Clear
// OC1B output: Clear
// OC1C output: Clear
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR1A=0xA8;
TCCR1B=0x02;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x0a;
OCR1AL=0xcd;
OCR1BH=0x07;
OCR1BL=0x33;
OCR1CH=0x07;
OCR1CL=0x33;

// Timer/Counter 3 initialization
// Clock source: System Clock
// Clock value: 1843.200 kHz
```

```
// Mode: Normal top=FFFFh
// Noise Canceler: Off
// Input Capture on Falling Edge
// OC3A output: Clear
// OC3B output: Clear
// OC3C output: Discon.
// Timer 3 Overflow Interrupt: On
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
// Compare C Match Interrupt: Off
TCCR3A=0xA0;
TCCR3B=0x02;
TCNT3H=0x00;
TCNT3L=0x00;
ICR3H=0x00;
ICR3L=0x00;
OCR3AH=0x0a;
OCR3AL=0xcd;
OCR3BH=0x0a;
OCR3BL=0xcd;
OCR3CH=0x00;
OCR3CL=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;
ETIMSK=0x04;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);

// Global enable interrupts
#asm("sei")

Dealer = 9;

while(1){

do{     delay_ms(200);
        P1 = PINA.0;
        P2 = PINA.1;
        P3 = PINA.2;
        P4 = PINA.3;
        P5 = PINA.4;
        P6 = PINA.5;
        P7 = PINA.6;
        P8 = PINA.7;
        G1 = (!PIND.2 && !PIND.1 && !PIND.0);
        G2 = (!PIND.2 && !PIND.1 && PIND.0);
        G3 = (!PIND.2 && PIND.1 && !PIND.0);
        G4 = (!PIND.2 && PIND.1 && PIND.0);
        G5 = (PIND.2 && !PIND.1 && !PIND.0);
        G6 = (PIND.2 && !PIND.1 && PIND.0);
        G7 = (PIND.2 && PIND.1 && !PIND.0);
        G8 = (PIND.2 && PIND.1 && PIND.0);
        PN = 0;
        if(P1){PN++;}
        if(P2){PN++;}
        if(P3){PN++;}
        if(P4){PN++;}
        if(P5){PN++;}
        if(P6){PN++;}
        if(P7){PN++;}
        if(P8){PN++;}
```

```
        if(PN < 2){
                invalidInput();
                continue;
                }
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("  Player");
        lcd_gotoxy(0,0);
        if(PN == 2){lcd_putsf("2");}
        if(PN == 3){lcd_putsf("3");}
        if(PN == 4){lcd_putsf("4");}
        if(PN == 5){lcd_putsf("5");}
        if(PN == 6){lcd_putsf("6");}
        if(PN == 7){lcd_putsf("7");}
        if(PN == 8){lcd_putsf("8");}
        lcd_gotoxy(0,1);
        if(G1){ lcd_putsf(" Texas Hold'Em");}
        if(G2){ lcd_putsf("     Omaha");}
        if(G3){
                lcd_putsf("  5 Card Draw");
                if(PN > 6){ invalidInput();}
                }
        if(G4){
                lcd_putsf("  7 Card Stud");
                if(PN == 8){ invalidInput();}
                }
        if(G5){
                lcd_putsf("  7's Take All");
                if(PN == 8){ invalidInput();}
                }
        if(G6){ lcd_putsf(" Super Hold'Em");}
        if(G7){ lcd_putsf(" Cool Hand Luke");}
        if(G8){ lcd_putsf("TripFlop Hold'Em");}

        }while(!(PIND.5 && PIND.4));

if(Dealer == 9){
                if(P8){ Dealer = 8;   }
        if(P7){ Dealer = 7;   }
        if(P6){ Dealer = 6;   }
        if(P5){ Dealer = 5;   }
        if(P4){ Dealer = 4;   }
        if(P3){ Dealer = 3;   }
        if(P2){ Dealer = 2;   }
                if(P1){ Dealer = 1;   }
        }

delay_ms(3000);
driveForward();
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("    Dealing");

if(G1 || G7){
        up=0;
        dealOneEP();
        dealOneEP();
        Holdaha();
        }

if(G2){
        up=0;
        dealOneEP();
        dealOneEP();
        dealOneEP();
        dealOneEP();
        Holdaha();
        }

if(G3){
```

```
up=0;
dealOneEP();
dealOneEP();
dealOneEP();
dealOneEP();
dealOneEP();
driveBackward();
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("Comence Betting");
while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
      delay_ms(1000);
      if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
      };
if(Dealer == 8){
      if(P1){ drawcards(1840); }
      if(P2){ drawcards(2100); }
      if(P3){ drawcards(2370); }
      if(P4){ drawcards(2635); }
      if(P5){ drawcards(2900); }
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      }
if(Dealer == 1){
      if(P2){ drawcards(2100); }
      if(P3){ drawcards(2370); }
      if(P4){ drawcards(2635); }
      if(P5){ drawcards(2900); }
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      if(P1){ drawcards(1840); }
      }
if(Dealer == 2){
      if(P3){ drawcards(2370); }
      if(P4){ drawcards(2635); }
      if(P5){ drawcards(2900); }
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      if(P1){ drawcards(1840); }
      if(P2){ drawcards(2100); }
      }
if(Dealer == 3){
      if(P4){ drawcards(2635); }
      if(P5){ drawcards(2900); }
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      if(P1){ drawcards(1840); }
      if(P2){ drawcards(2100); }
      if(P3){ drawcards(2370); }
      }
if(Dealer == 4){
      if(P5){ drawcards(2900); }
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      if(P1){ drawcards(1840); }
      if(P2){ drawcards(2100); }
      if(P3){ drawcards(2370); }
      if(P4){ drawcards(2635); }
      }
if(Dealer == 5){
      if(P6){ drawcards(3165); }
      if(P7){ drawcards(3400); }
      if(P8){ drawcards(3690); }
      if(P1){ drawcards(1840); }
      if(P2){ drawcards(2100); }
```

```
            if(P3){ drawcards(2370); }
            if(P4){ drawcards(2635); }
            if(P5){ drawcards(2900); }
            }
        if(Dealer == 6){
            if(P7){ drawcards(3400); }
            if(P8){ drawcards(3690); }
            if(P1){ drawcards(1840); }
            if(P2){ drawcards(2100); }
            if(P3){ drawcards(2370); }
            if(P4){ drawcards(2635); }
            if(P5){ drawcards(2900); }
            if(P6){ drawcards(3165); }
            }
        if(Dealer == 7){
            if(P8){ drawcards(3690); }
            if(P1){ drawcards(1840); }
            if(P2){ drawcards(2100); }
            if(P3){ drawcards(2370); }
            if(P4){ drawcards(2635); }
            if(P5){ drawcards(2900); }
            if(P6){ drawcards(3165); }
            if(P7){ drawcards(3400); }
            }
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(!PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("We Have A Winner");
        OCR1A= 2760;
        PORTB.4=1;
        PORTB.3=1;
        delay_ms(10000);
        PORTB.4=0;
        PORTB.3=0;
        delay_ms(1500);
        OCR1BH=0x0E;
        OCR1BL=0x6A;
        delay_ms(2000);
        PORTB.0=1;
        delay_ms(10000);
        PORTB.0=0;
        OCR1BH=0x07;
        OCR1BL=0x30;
        passButton();
        }

    if(G4 || G5){
        up=0;
        dealOneEP();
        dealOneEP();
        up=1;
        dealOneEP();
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("2nd Street");
        dealOneEP();
        lcd_clear();
```

```
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("3rd Street");
        dealOneEP();
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("4th Street");
        dealOneEP();
        up=0;
        dealOneEP();
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(!PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
              delay_ms(1000);
              if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2760); }
              };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("We Have A Winner");
        OCR1A= 2760;
        PORTB.4=1;
        PORTB.3=1;
        delay_ms(10000);
        PORTB.4=0;
        PORTB.3=0;
        delay_ms(1500);
        OCR1BH=0x0E;
        OCR1BL=0x6A;
        delay_ms(2000);
        PORTB.0=1;
        delay_ms(10000);
        PORTB.0=0;
        OCR1BH=0x07;
        OCR1BL=0x30;
        passButton();
        }

if(G6){
        up=0;
        dealOneEP();
        dealOneEP();
        dealOneEP();
        Holdaha();
        }


if(G8){
        up=0;
        dealOneEP();
        dealOneEP();
        delay_ms(1000);
        driveBackward();
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
```

```
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2000); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Here's The 1st Flop");
        deal(2220);
        up=1;
        deal(2340);
        deal(2460);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2000); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Here's The 2nd Flop");
        deal(2580);
        up=1;
        deal(2700);
        deal(2820);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2000); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("And The 3rd Flop");
        deal(2940);
        up=1;
        deal(3060);
        deal(3180);
        up=0;
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("Comence Betting");
        while(!(!PINF.3 && !PINF.2 && !PINF.1 && PINF.0)){
            delay_ms(1000);
            if(PINF.3 && !PINF.2 && !PINF.1 && !PINF.0){ deal(2000); }
            };
        lcd_clear();
        lcd_gotoxy(0,0);
        lcd_putsf("We Have A Winner");
        OCR1A= 3300;
        PORTB.4=1;
        PORTB.3=1;
        delay_ms(10000);
        PORTB.4=0;
        PORTB.3=0;
        delay_ms(1500);
        OCR1BH=0x0E;
        OCR1BL=0x6A;
        delay_ms(2000);
        PORTB.0=1;
        delay_ms(10000);
        PORTB.0=0;
        OCR1BH=0x07;
        OCR1BL=0x30;
        passButton();
        }
};
}
```