Frog Jump

Design Documentation

Trevor Brown

ECE 3710 Final Project

Table of Contents

Images

Introduction

Why did the chicken cross the road? To get to the other side. So then why did the frog cross the road? Because he knew he was much braver than some chicken and if the chicken could do it, so could he. Frog Jump is just that, reaching the other side of the road amongst a busy highway and avoiding getting killed by these fast paced cars! Work hard and get as many wins as you can. Try to get the fastest time as well! If you think the game is too easy, then feel free to step up the difficulty and make it harder than ever! With all that you can do in this game, it will be hard to put it down!

Purpose of the Game

The purpose of this game is to be able to reach the finish position while jumping through an obstacle of cars. When you turn on the game, you will be taken to a menu screen. You are able to see the wins/losses as well as the high score. You can change the color of the frog in the menu screen as well. When ready, press up or down to get you to the gameplay screen. You must wait for the screen to setup and for the cars to get going. After waiting a few seconds you can go. You can then jump around and make your way through a never ending maze of cars. Time your moves right and navigate your frog to the finish position at the end. Upon completion you will add to your wins and if your current time is faster than the older time then you obtain the new high score. If your frog gets hit by a car then you lose and will be taken back to the menu screen.

Design Overview

Requirements:

1. **Display the menu screen.**
2. **Be able to see the high score and the wins/losses.**
3. **Change the color of the frog in the menu screen**
4. **Enter game play mode.**
5. **By pushing the buttons, have the frog jump to the desired square.**
6. **Have cars travel in patterns automatically.**
7. **When the frog gets hit by the car, the player loses.**
8. **When the frog gets to the finish position, the player wins.**
9. **Enable music for the menu and during game play.**
10. **Enable sounds for frog jumps, winning and losing.**

Hardware:

1. **Tiva C series TM4C123GH6PMI Microcontroller**
2. **3.2” TFT LCD Display Module**
3. **3.2” LCD adapter, 16-bit to 8-bit**

4. Analog test board
5. 5 volt power supply
6. 4 push buttons

Software:

Things that need to get programmed:

1. Menu screen
    a. Display the Title
    b. Display the High Score
    c. Display the Wins/Loses
    d. Display instructions for buttons for screen
    e. Display the frog
    f. Display two cars, one moving and one staying still.
    g. Play music
    h. Enable button pushes to trigger events
        i. Up and down start game
        ii. Left and right change the frog's color
2. Gameplay Screen
    a. Display background
    b. Display boundaries
    c. Display Start and finish position
    d. Play music
3. The Frog
    a. Update position of the frog
        i. Up button jumps frog up
        ii. Down button jumps frog down
        iii. Right button jumps frog right
        iv. Left button jumps frog left
    b. Create frog image
    c. Create position for frog
4. The Cars
    a. Enable automatic updates of car paths
    b. Create car image
    c. Create positions for each car
    d. Create patterns for cars
5. Winning State
    a. Display Win
    b. Return to menu
6. Losing State
    a. Display Lose
    b. Return to menu

Design Details

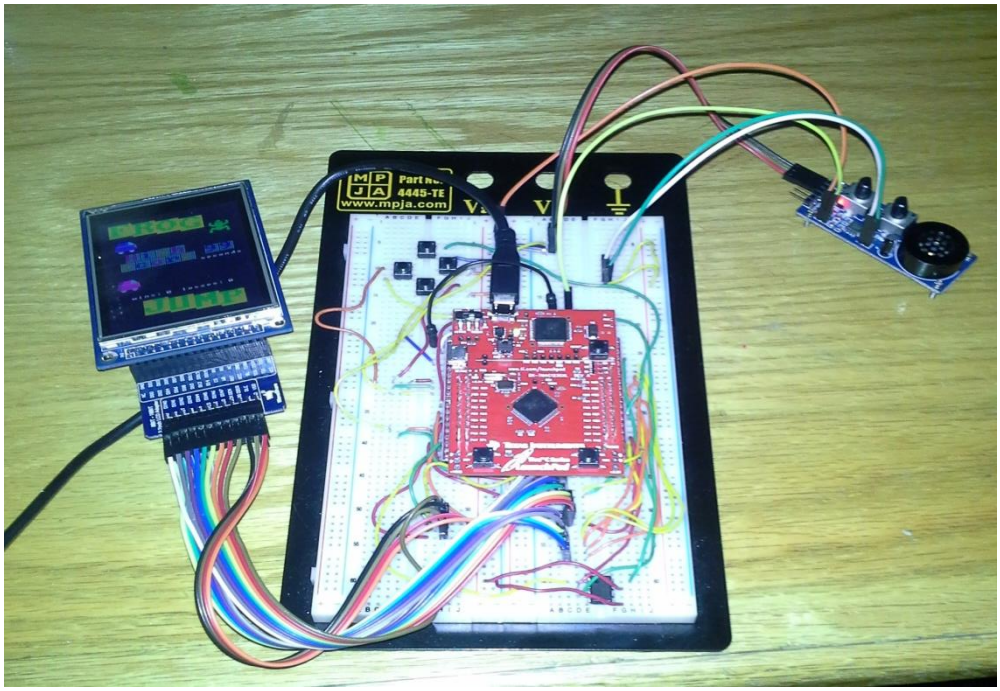
Hardware:

Port Configuration:

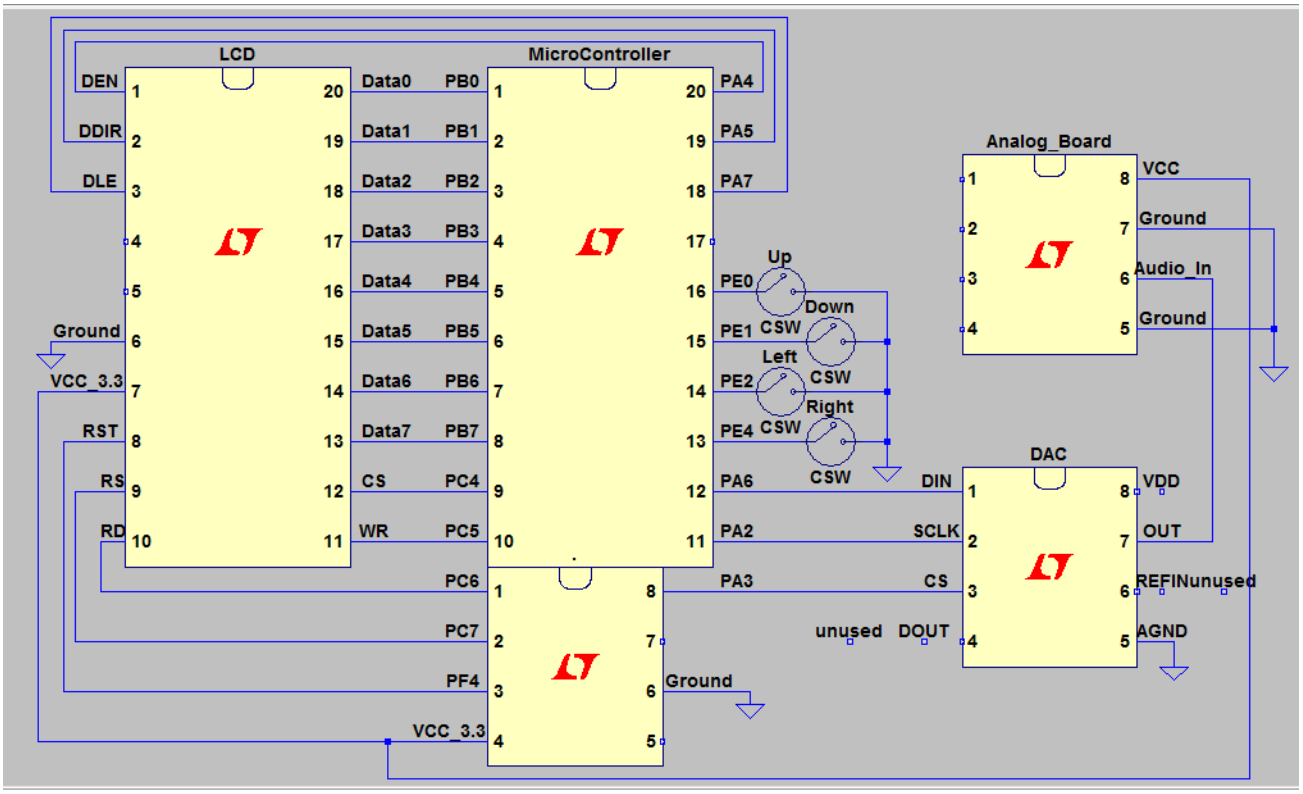
PA2 – SCLK on DAC
PA3 – CS on DAC
PA4 – DEN on LCD module
PA5 -- DDIR on LCD module
PA6 – DLE on LCD module
PB0 – Data0 on LCD module
PB1 – Data1 on LCD module
PB2 – Data2 on LCD module
PB3 – Data3 on LCD module
PB4 – Data4 on LCD module
PB5 – Data5 on LCD module
PB6 – Data6 on LCD module
PB7 – Data7 on LCD module
PC4 – CS on LCD module
PC5 – WR on LCD module
PC6 -- RD on LCD module
PC7—RS on LCD module
PE0 – Push button (up)
PE1 – Push button (down)
PE2 – Push button (left)
PE4 – Push button (right)
PE3 – ACD2 on Analog board
PF4 – RST on LCD module
UBUS – VDD on DAC
VCC (3.3V) – VCC on LCD module, VCC on analog board, VCC for speaker

Hardware Setup:

Actual Setup:



Hardware Schematic:

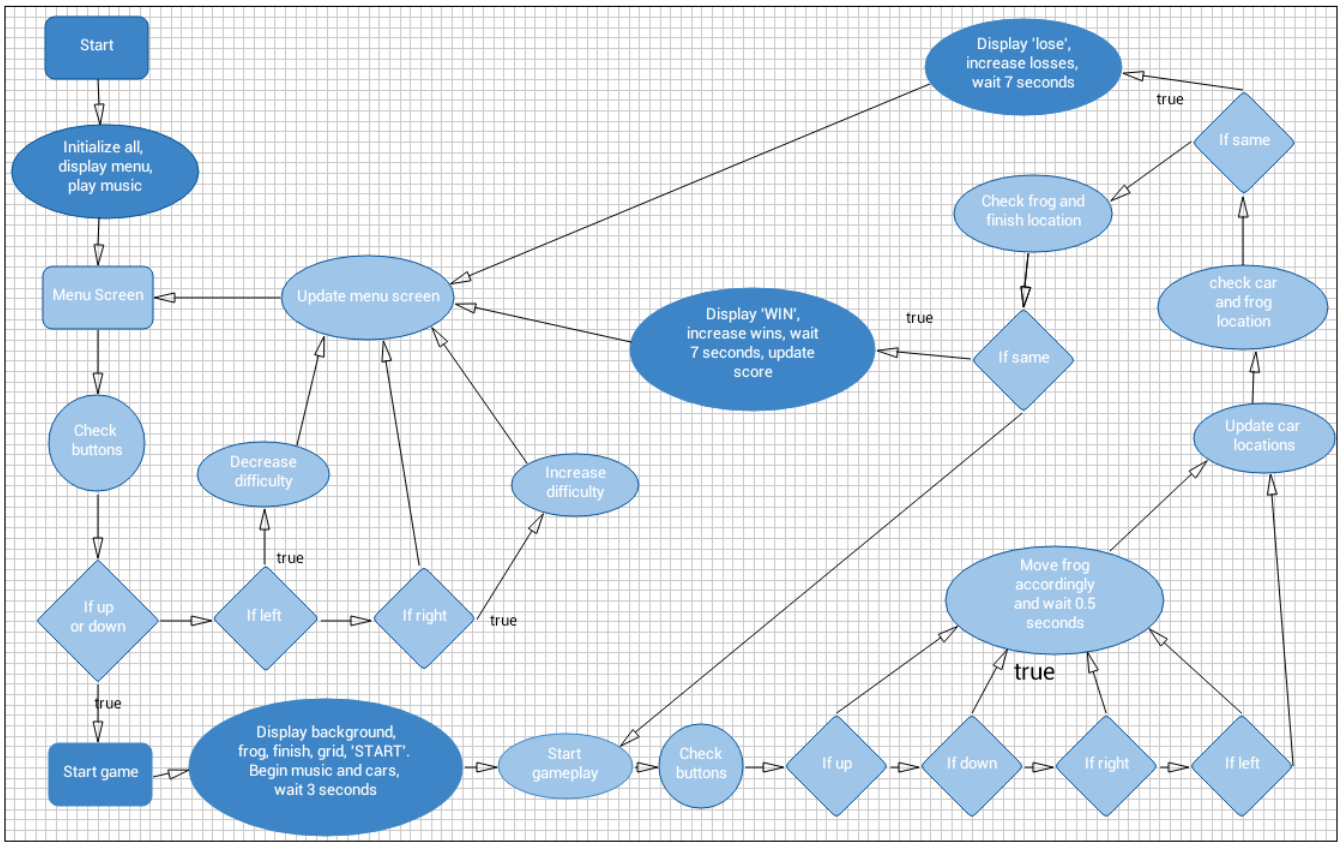


Explanation of the Setup:

I decided to try to follow what we did in the labs. From our final lab we were able to to use the LCD screen with the DAC and analog board. I tried to keep these settings the same. On top of that I needed to add 4 buttons. I struggled getting the buttons to work so I had to change some things from the last lab's setup. I kept all of port B to be the data pins for the LCD screen as well as all of port C4-C7 for the CS, RS, RD, and WR, with port PF4 as the RST. From the last lab we used ports E0-E2 for the DDIR, DEN, DLE but I had to switch them to PA4, PA5, and PA7. I then kept the DAC ports the same, being PA2 for SCLK, PA3 for CS for the DAC, and PA6 for DIN. I kept PE3 being used for the Analog board and used the pins around it for the buttons, those pins being PE0, PE1, PE2, PE4. I then tied the VCC and GND as needed from each piece of equipment to the micro controller. I also needed to use a power supply to power the analog board.

Software:

The Flow Chart:



Explanation of the Flow Chart:

Upon turning on the game or reset, we must initialize everything that we are using. We first initialize the Tiva C Series Microcontroller with its various timers, GPIO ports, interrupts, and the DAC. For more information regarding the setup see the Hardware section under Design Details. After that we initialize the LCD module for displaying data only. We do not make use of the touchscreen. Once we have everything initialized, we display the menu on the LCD screen and play some background music. We wait in this state until one of the buttons is pushed. When the left or right button is pushed, we change the color of the frog. When either the up or down button is pushed, we continue with the game.

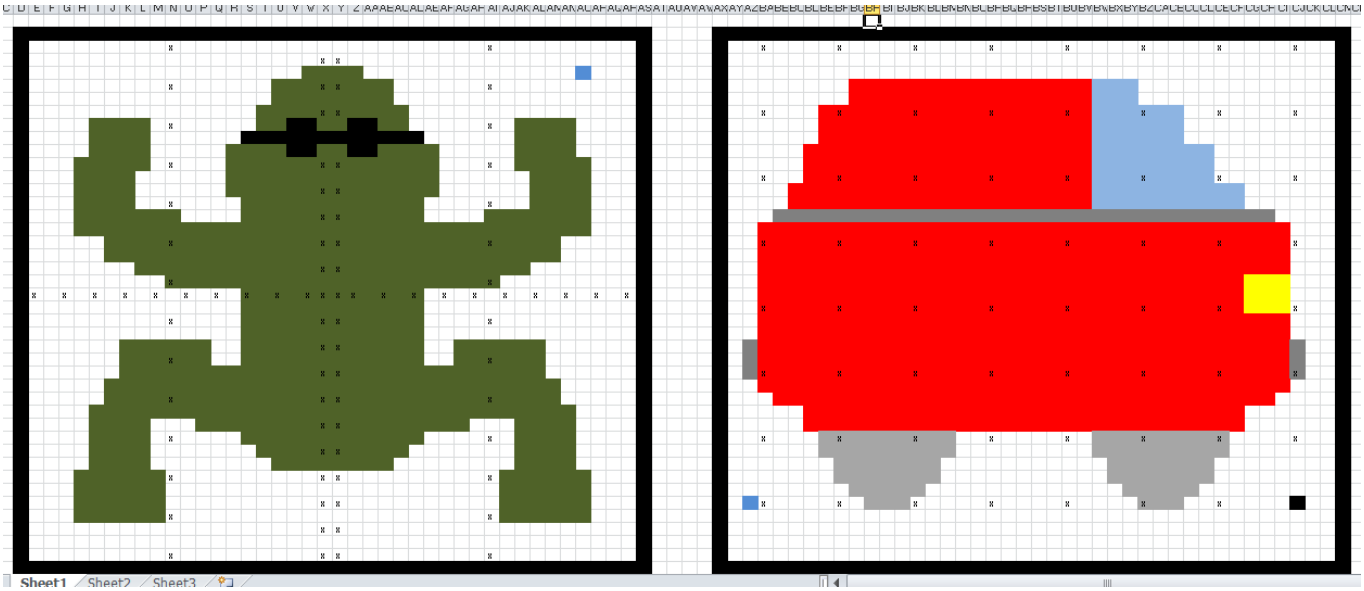
Before we start the gameplay, we must setup how everything is displayed. We display the background, followed by the finish position, the start position, the frog, and the boundaries. We enable the cars to start their patterns. We also display 'START' for 3 seconds and we do not allow the player to start playing until the 'START' is gone. Once the "START" is gone, we display a "GO" around the frog and set a flag that will enable the buttons to work.

After the 'START' is gone, the player is then free to jump around and avoid the cars. To do this we must check to see which buttons are being pushed. If the up button is pushed then we have the frog jump up and face up. We mimic this with the down jumping the frog down and facing down, with the right, and with the left. The frog limited to jump within the boundaries set, which are the top and bottom of the screen, as well as the middle portion of the screen. After the frog either jumps in any direction or sits there, the cars get updated. Each track gets updated and the car moves to its new location. After the frog and the cars are updated, we must check to see if the car has run over the frog. If this has happened we immediately display 'LOSE' and wait a little bit before we move back to the menu screen. We then must update the screen and increase the number of losses. If the frog doesn't get hit and makes his way to the finish position, then we immediately display 'WIN' and wait a little bit before we go back to the menu screen. We update the menu screen to include the new amount of wins as well as the high score. We then repeat the menu screen state.

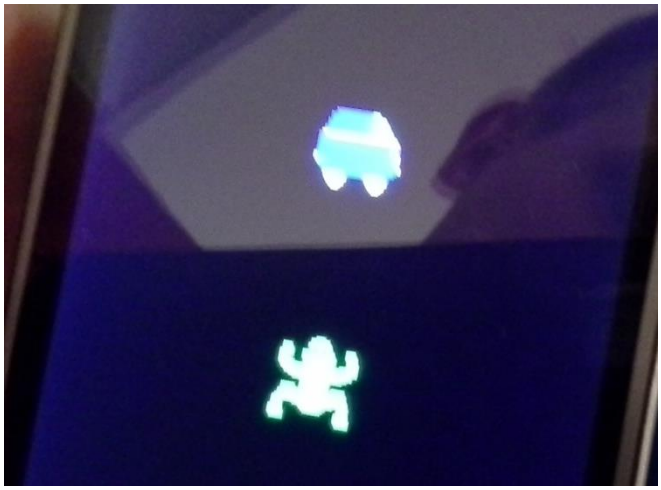
Testing

Creating images:

When I thought about making the images, I started out using Excel. I created a 40X40 area that would be designated for the frog and another for the car. I made each square about the same size. I then started filling in areas to design the frog. I made it symmetric. I did something similar with the car. I designed it as I wanted. Here is the excel picture of it:



From this I was able to create code that would create a lot of boxes for each part of the frog or car. I would feed the function the coordinate where I wanted the box, and then I would pass in how long in the x-direction as well as how long in the y-direction. We would also throw in the color that we wanted for most of the frog or car. After doing so this is how I got the car and frog to appear on the screen:



Moving the Frog:

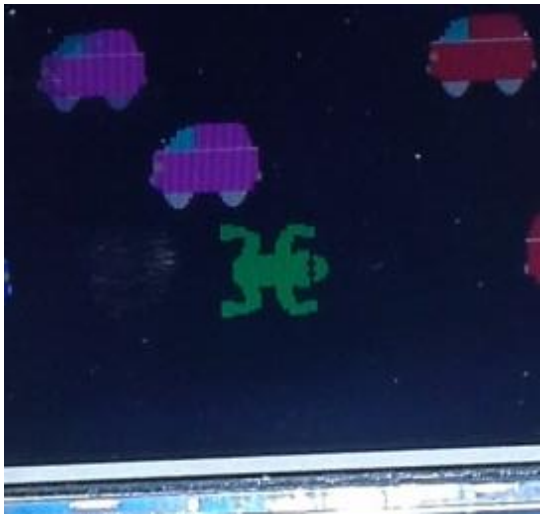
Once I got the buttons working, I started working on the frog's movement. I created 4 different directions for the frog so I could keep the position of the frog relative for each movement. For each button I either added or subtracted from either the X coordinate or Y coordinate and saved this value as the new X and Y coordinate. After we push a direction, we clear the frog completely at its current position, we increment or decrement the X or Y depending on which button is pressed. We then draw the specific direction of the frog. Here are a pictures of each direction:

Frog facing up:

Frog facing down:

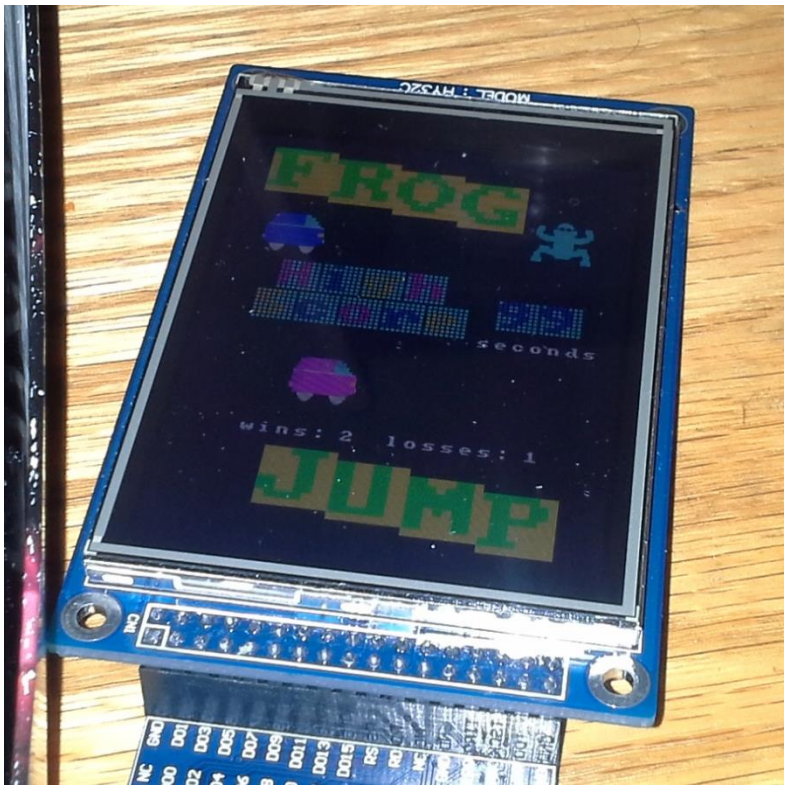
Frog facing left:

Frog facing right:



Creating the Menu Screen:

There were a handful of things I wanted in the screen. I needed to put the title. I needed to display the "High Score" followed by the actual high score. For design I put a picture of a car. I also decided it would be fun to have a car drive around, so I was able to get that working. I decided to put a frog on the screen. I made the frog be able to change color and that the color that the frog is when starting the game is the color of the frog during the game play. I also decided it would be fun to denote how many wins and losses the player has and I would display "wins" followed by how many and also "losses" followed by how many. Here is how it looks:



Creating the Car paths:

The goal of this was to try and get different patterns and different speeds of the cars going in different directions. I was able to get half of the cars going one way and the other half going the other direction. I also was able to get the cars in different colors. I had trouble with the speed so I wasn't able to get the cars moving at different speeds. But I was able to get the patterns of the cars to be different. I had the cars regenerate at different rates which makes the game change as time goes by. These are a few pictures of the screen:

Win and Lose State:

Win State: For the design I decided to do a couple of things. Once the frog's coordinates are the same as the finish position, we enter this state. We display a "You Win" signifying that the player won. We also must display the time it took the player to win. So we display "Score" followed by the time it took. I decided to add a couple of frogs in the screen because the frog successfully crossed the road. We also wait a couple of seconds before we go back to the menu screen. I organized them as shown:

Lose State: We enter this state when the frog's coordinates match up with any of the car's coordinates. When this happens we display a "Game Over", signifying that the player has lost. We also display a bunch of cars because the cars ran the frog over. We wait a few seconds before we go back to the menu screen. I organized it as follows:

Conclusion

Some of the problems I faced:

1) I had a problem with the speed of the cars. Because of all of the code that I was running, I couldn't update the cars as fast as I wanted to. I tried to make it faster but it wasn't going as fast as I hoped. So I tried to update the cars faster by reducing the amount of cars to update. When I did this the cars moved twice as fast, but that was still slower than I wanted. I then tried to change the way I update the cars. I drew the first eight pixels of the car and let them drag so I was only really updating an 8 x 40 pixel area instead of a 40 x 40 pixel area. Sadly this didn't increase the speed by that much, but it made them slightly faster.
2) I struggled with getting the ports set up correctly for the push buttons. I initially tried to set up port D. After I tried this and tried to get them working, I realized this port doesn't work as I had hoped. I later found out that port D0 and port D1 are linked with some of the port B pins. I then tried to switch to port A but I wasn't having luck with that. I then switched to port E and tried to get it working. It was working but I struggled to get the right pins. The simulator didn't help and threw me off a lot because it wouldn't show the data on port E correctly and it changed almost every time I reran the debugger. After a lot of changing I found out that the simulator doesn't show the button pushes but that they work regardless of what the simulator shows. I ended up putting the buttons on port E0, E1, E2, and E4, reserving E3 for the Analog board. Once I realized this fact, then things worked out really well and I implemented functions accordingly for the menu screen and the game screen.

Besides these problems, most of the other stuff went normal and just took time to get things rolling. I enjoyed working on the game and working on some of the bugs that came up and getting everything in a working order. I enjoyed this project and look forward to creating more and more projects like this.

Code

My Main.c

```
#include "LCDfinal.H"

int main(void)
{
  unsigned short bg = black;
  lcdPortConfig();
  lcdInit();
  clearLCD(bg);
  //SSI0Init();
        //initial values
        while(1)
        {
                menuScreen();
        }
}
```

My FrogGame.h

```
/* function prototypes for SSD1289 LCD on Tiva C Launchpad */

// 16-bit RGB values for some common colors
#define white       0xFFFF
#define black       0x0001
#define gray        0xF7DE
#define blue        0x001F
#define red         0xF800
#define magenta     0xF81F
#define green       0x07E0
#define cyan        0x7FFF
#define yellow      0xFFE0

// functions to init LCD and GPIO
void lcdPortConfig(void); //setup GPIO pins
void lcdInit(void); //base LCD config

// functions to send cmd/data to LCD
void writeCmd(unsigned short cmd); //command: where we write data to
void writeDat(unsigned short dat); //data: what gets written to address given by cmd
void writeReg(unsigned short cmd,unsigned short dat); //either wrapper for above or combine above to avoid
function calls

// helper functions
```

```
void clearLCD(unsigned short rgb); //set entire LCD to the color rgb (useful for debugging above)
void setCursor(unsigned short x,unsigned short y); //set current pixel to x,y
void delayMS(char ms); //generate ms milliseconds of delay (timer or loop, your choice)
void B8_B16(unsigned short value);
void setTimer(unsigned int value);
char timerExpired(void);

//menu screen functions
void menuScreen(void);
void checkBtnsMenu(void);


//game play functions
void clearCarLtoR(unsigned int tempX, unsigned int tempY);
void clearCarRtoL(unsigned int tempX, unsigned int tempY);
unsigned int clearObject(unsigned int obPosition);
void gameScreen(void);
void gameInit(void);
void gameStart(void);
void checkAll(void);
void checkStart(void);
void checkWin(void);
void checkLoss(void);
void initAll(void);
void checkCar(int carX, unsigned int carY);
void lossState(void);
void checkBtnsGame(void);
void longDelay(unsigned int number);
void calcHighScore(void);

//car path updates
void updateAllCars(void);
void firstRow(void);
void secondRow(void);
void thirdRow(void);
void fourthRow(void);
void fifthRow(void);
void sixthRow(void);

//image functions

//frog images
void createFrogUp(unsigned int x, unsigned int y, unsigned int color);
void createFrogDown(unsigned int x, unsigned int y, unsigned int color);
void createFrogLeft(unsigned int x, unsigned int y, unsigned int color);
void createFrogRight(unsigned int x, unsigned int y, unsigned int color);
//car images
void createCarFrongLtoR(unsigned int x, unsigned int y, unsigned int color);
void createCarFrontRtoL(unsigned int x, unsigned int y, unsigned int color);
void createCarLtoR(unsigned int x, unsigned int y, unsigned int color);
```

```
void createCarRtoL(unsigned int x, unsigned int y, unsigned int color);
//letter images
void writeLetter(unsigned short x, unsigned short y, unsigned int letter);
void writeLetterBig(unsigned short x, unsigned short y, unsigned int letter);
void writeLetterMagenta(unsigned short x, unsigned short y, unsigned int letter);
void writeLetterBlue(unsigned short x, unsigned short y, unsigned int letter);
void writeLetterYellow(unsigned short x, unsigned short y, unsigned int letter);
//helper functions for drawing images and automatically updating images
void drawParts(unsigned int startX, unsigned int startY, unsigned int quantityX, unsigned int quantityY,
unsigned int color);
int updateCarLtoR(int positionX,unsigned int positionY, unsigned int color, int length);
int updateCarRtoL(int positionX,unsigned int positionY, unsigned int color, int length);
```

My FrogGame.c

```
#include "LCDfinal.H"
#include "font8x8_basic.h"

//addresses of everything that we are using
unsigned int CLOCKFREQ = 16000000;
unsigned int *STBASE = (unsigned int *) 0xE000E000;
volatile int ST_CTRL __attribute__((at(0xE000E010)));
volatile char ST_RE __attribute__((at(0xE000E014)));
volatile char ST_CURR __attribute__((at(0xE000E018)));
unsigned char * PA = (unsigned char *) 0x40004000;
unsigned char * PB = (unsigned char *) 0x40005000;
unsigned char * PC = (unsigned char *) 0x40006000;
unsigned char * PD = (unsigned char *) 0x40007000;
unsigned char * PE = (unsigned char *) 0x40024000;
volatile char   PE_T __attribute__((at(0x42487F8C)));
unsigned char * PF = (unsigned char *) 0x40025000;
volatile int ALOCK __attribute__((at(0x40004520)));
volatile int RCGC2 __attribute__((at(0x400FE108)));
volatile int RCGC1 __attribute__((at(0x400FE104)));
volatile char csLCD __attribute__((at(0x420C7F90)));
volatile char wrLCD __attribute__((at(0x420C7F94)));
volatile char rdLCD __attribute__((at(0x420C7F98)));
volatile char dcLCD __attribute__((at(0x420C7F9C)));
volatile char DAC_SCLK __attribute__((at(0x420C788)));
volatile char DAC_CS __attribute__((at(0x420C7F8C)));
volatile char denAD       __attribute__((at(0x42087F90)));
volatile char DAC_DIN __attribute__((at(0x42087F94)));
volatile char ddirAD       __attribute__((at(0x42087F98)));
volatile char dleAD __attribute__((at(0x42087F9C)));

unsigned short value = 0x0;
unsigned int MAX_X = 320;                                   //size of screen in Y direction
unsigned int MAX_Y = 240;                                   //size of screen in X direction
```

```
unsigned int scoreOne = 0;                          //high score one's value
unsigned int scoreTen = 0;                          //high score ten's value
unsigned int finalScoreOne = 0;          //final high score one's value
unsigned int finalScoreTen = 0;          //final high score ten's value
int menuCar = 250;                                                          //menu car's x=coordinate

unsigned int FROGX = 100;                                   //frog's x-coordinate
unsigned int FROGY = 0;                                     //frog's y-coordinate
unsigned int FROGCOLOR = green;                //frog's color
unsigned int countForStart = 0;          //counter for start number
unsigned int startFlag = 0;                         //flag for start
unsigned int countEnd = 0;                          //counter for end wait
unsigned int loseFlag = 0;                          //flag for losing
unsigned int newColor = 0x1;                   //tracker for assigning color of the frog
unsigned int movesScore;
unsigned int timeScore;
unsigned int scoreCounter;
unsigned int scoreCountDown;
unsigned int finalScore;
unsigned int oldScore;

unsigned int numOfWins = 0x0;                  //tracks number of wins
unsigned int numOfLosses =0x0;                 //tracks number of losses

//global variables for the cars
//these are the y-coordinates of the cars in their respective lanes
unsigned int car1Y =40;
unsigned int car2Y = 80;
unsigned int car3Y = 120;
unsigned int car4Y = 160;
unsigned int car5Y = 200;
unsigned int car6Y = 240;
//these are the starting x-coordinates for each car in their respective lanes
int car1A =260;
int car1B =320;
int car2A =-80;
int car2B =-200;
int car3A =-40;
int car3B = -220;
int car4A = 280;
int car4B = 380;
int car5A = 240;
int car5B = 420;
int car6A = -60;
int car6B = -160;
int car6C = -210;


//basic LCD initialization
void lcdInit(void)
```

```
{
//It may be a good idea to reset the LCD and turn on the backlight here (can be done using GPIO)
//LCD reset
//backlight
  PF[0x3FC] = 0x00;
  PF[0x3FC] = 0x10;

//see the datasheet for an explanation of the following
//  (shouldn't need to change anything below)
writeReg(0x0000,0x0001);    delayMS(50);   /* Enable LCD Oscillator */
writeReg(0x0003,0xA8A4);    delayMS(50);   // Power control(1)
writeReg(0x000C,0x0000);    delayMS(50);   // Power control(2)
writeReg(0x000D,0x080C);    delayMS(50);   // Power control(3)
writeReg(0x000E,0x2B00);    delayMS(50);   // Power control(4)
writeReg(0x001E,0x00B0);    delayMS(50);   // Power control(5)
writeReg(0x0001,0x2B3F);    delayMS(50);   // Driver Output Control /* 320*240 0x2B3F */
writeReg(0x0002,0x0600);    delayMS(50);   // LCD Drive AC Control
writeReg(0x0010,0x0000);    delayMS(50);   // Sleep Mode off
writeReg(0x0011,0x6070);    delayMS(50);   // Entry Mode
writeReg(0x0005,0x0000);    delayMS(50);   // Compare register(1)
writeReg(0x0006,0x0000);    delayMS(50);   // Compare register(2)
writeReg(0x0016,0xEF1C);    delayMS(50);   // Horizontal Porch
writeReg(0x0017,0x0003);    delayMS(50);   // Vertical Porch
writeReg(0x0007,0x0133);    delayMS(50);   // Display Control
writeReg(0x000B,0x0000);    delayMS(50);   // Frame Cycle control
writeReg(0x000F,0x0000);    delayMS(50);   // Gate scan start position
writeReg(0x0041,0x0000);    delayMS(50);   // Vertical scroll control(1)
writeReg(0x0042,0x0000);    delayMS(50);   // Vertical scroll control(2)
writeReg(0x0048,0x0000);    delayMS(50);   // First window start
writeReg(0x0049,0x013F);    delayMS(50);   // First window end
writeReg(0x004A,0x0000);    delayMS(50);   // Second window start
writeReg(0x004B,0x0000);    delayMS(50);   // Second window end
writeReg(0x0044,0xEF00);    delayMS(50);   // Horizontal RAM address position
writeReg(0x0045,0x0000);    delayMS(50);   // Vertical RAM address start position
writeReg(0x0046,0x013F);    delayMS(50);   // Vertical RAM address end position
writeReg(0x0030,0x0707);    delayMS(50);   // gamma control(1)
writeReg(0x0031,0x0204);    delayMS(50);   // gamma control(2)
writeReg(0x0032,0x0204);    delayMS(50);   // gamma control(3)
writeReg(0x0033,0x0502);    delayMS(50);   // gamma control(4)
writeReg(0x0034,0x0507);    delayMS(50);   // gamma control(5)
writeReg(0x0035,0x0204);    delayMS(50);   // gamma control(6)
writeReg(0x0036,0x0204);    delayMS(50);   // gamma control(7)
writeReg(0x0037,0x0502);    delayMS(50);   // gamma control(8)
writeReg(0x003A,0x0302);    delayMS(50);   // gamma control(9)
writeReg(0x003B,0x0302);    delayMS(50);   // gamma control(10)
writeReg(0x0023,0x0000);    delayMS(50);   // RAM write data mask(1)
writeReg(0x0024,0x0000);    delayMS(50);   // RAM write data mask(2)
writeReg(0x0025,0x8000);    delayMS(50);   // Frame Frequency
writeReg(0x004f,0);                        // Set GDDRAM Y address counter
writeReg(0x004e,0);                        // Set GDDRAM X address counter
```

```
delayMS(50);
}

// functions to init LCD and GPIO
void lcdPortConfig(void){
RCGC1 = 0x00030000;
// setupPorts();

RCGC2 = 0x37; //0b0011 1111
__asm volatile ("nop");
__asm volatile ("nop");

//config port B DATA PORT 8
PB[0x420] = 0x0;//AF
PB[0x400] = 0xFF; //DIR
PB[0x50C] = 0x00;//ODR
PB[0x510] = 0xFF; //PUR
PB[0x51C] = 0xFF; //DEN

//config port A TOUCH PORT UPPER 2
ALOCK = 0x4C4F434B;
PA[0x420] = 0x06; //AF
PA[0X52D] = 0X22;
PA[0X52E] = 0X22;
PA[0x400] = 0xFC; //DIR
PA[0x50C] = 0x00; //ODR
PA[0x510] = 0xF3; //PUR
PA[0x51C] = 0xFC; //DEN

//config port C CTRL PINS
//PC4 = CS
//PC5 = WR
//PC6 = RD
//PC7 = (RS) D/C
PC[0x420] &= 0x0;//AF
PC[0x400] |= 0xF0; //DIR
PC[0x50C] |= 0x00;//ODR
PC[0x510] |= 0xF0; //PUR
PC[0x51C] |= 0xF0; //DEN

//buttons
PE[0x420] = 0x0;//AF
PE[0x400] |= 0x00; //DIR
PE[0x50C] = 0x00;//ODR
PE[0x510] = 0xFF; //PUR
PE[0x51C] = 0xFF; //DEN

//config port F DATA PORT (PF.4 = RESET)
PF[0x420] = 0x0;//AF
```

```
PF[0x400] = 0xFF; //DIR
PF[0x50C] = 0xE7;//ODR
PF[0x510] = 0xFF; //PUR
PF[0x51C] = 0x18; //DEN

PC[0x3FC] = 0xFF;
PF[0x3FC] = 0x18;
PA[0x3FC] = 0XFF;
PB[0x3FC] = 0xFF;
PE[0x3FC] = 0xFF;
//PD[0x3FC] = 0xFF;
} //setup GPIO pins
// functions to send cmd/data to LCD
void writeCmd(unsigned short cmd){
  csLCD = 0;
  rdLCD = 1;
  dcLCD = 0;
  wrLCD = 0;

  B8_B16(cmd);

  wrLCD = 1;
  csLCD = 1;
} //command: where we write data to
void writeDat(unsigned short dat){
  csLCD = 0;
  rdLCD = 1;
  dcLCD = 1;
  wrLCD = 0;

  B8_B16(dat);

  wrLCD = 1;
  csLCD = 1;
} //data: what gets written to address given by cmd
void writeReg(unsigned short cmd,unsigned short dat){
// chipSelect(1);
  writeCmd(cmd);
  writeDat(dat);
} //either wrapper for above or combine above to avoid function calls

// helper functions
void clearLCD(unsigned short rgb){
  unsigned int i;
  setCursor(0,0);
  writeCmd(0x22);
  for (i = 0; i < (MAX_X * MAX_Y); i++)
  {
    writeDat(rgb);
  }
```

```
} //set entire LCD to the color rgb (useful for debugging above)
void setCursor(unsigned short x,unsigned short y){
writeReg(0x004E,x);    /* 0-239 */
writeReg(0x004F,y);    /* 0-319 */
} //set current pixel to x,y

void delayMS(char ms){
  unsigned int dTime = 0;
  dTime = (.001 * CLOCKFREQ) -1;
  setTimer(dTime);
  while(!timerExpired()){};
} //generate ms milliseconds of delay (timer or loop, your choice)
void B8_B16(unsigned short value){

  denAD = 0x0;
  ddirAD = 0x1;
  dleAD = 0x1;
  PB[0x3FC] = value;
  dleAD = 0x0;
  PB[0x3FC] = (value>>8);
}

void setTimer(unsigned int value){
  ST_CTRL = ST_CTRL & 0xFE;
  STBASE[0x14/4] = value;
  ST_CURR = 0x0;
ST_CTRL = ST_CTRL | 0x4; //(CLK_SRC, INTEN, ENABLE)
ST_CTRL = ST_CTRL & 0xFD; //(CLK_SRC, INTEN, ENABLE)
ST_CTRL = ST_CTRL | 0x1;
}

char timerExpired(void){
  if((ST_CTRL >> 16) == 0x1){
    return 1;
  }
  return 0;
}

//this writes the letters for the title, winning screen and winning area
void writeLetterBig(unsigned short x, unsigned short y, unsigned int letter){
  int i = 0;
  int j = 0;
            unsigned int tempX = x;
            unsigned int tempY = y;
            char line = 0x0;
  setCursor(tempX, tempY);
  for (i = 0; i < 8; i++)
  {
    line = font8x8_basic[letter][i];
    for (j = 0x1; j <= 0x80; j=j*0x2)
```

```
    {
     if((line & ((unsigned char) j)) != 0x0)
                                {
                                        drawParts(tempX, tempY, 5, 5 , green);
                                }
     else
                                {
                                        drawParts(tempX,tempY, 5, 5, yellow);
     }
                                tempX -= 0x5;
    }
    tempY -= 0x5;
                        tempX = x;
  }
}
//this function writes a stylish letter in the color of magenta
void writeLetterMagenta(unsigned short x, unsigned short y, unsigned int letter){
  int i = 0;
  int j = 0;
            unsigned int tempX = x;
            unsigned int tempY = y;
            char line = 0x0;
  setCursor(tempX, tempY);
  for (i = 0; i < 8; i++)
  {
    line = font8x8_basic[letter][i];
    for (j = 0x1; j <= 0x80; j=j*0x2)
    {
     if((line & ((unsigned char) j)) != 0x0)
                                {       drawParts(tempX, tempY, 2,2, magenta);          }
     else
                                {       drawParts(tempX,tempY,2,2, cyan); }
                                tempX -= 0x3;
    }
    tempY -= 0x3;
                        tempX = x;
  }
}
//this function writes a stylish letter in the color of blue
void writeLetterBlue(unsigned short x, unsigned short y, unsigned int letter){
  int i = 0;
  int j = 0;
            unsigned int tempX = x;
            unsigned int tempY = y;
            char line = 0x0;
  setCursor(tempX, tempY);
  for (i = 0; i < 8; i++)
  {
    line = font8x8_basic[letter][i];
    for (j = 0x1; j <= 0x80; j=j*0x2)
```

```
    {
     if((line & ((unsigned char) j)) != 0x0)
                                {           drawParts(tempX, tempY, 2,2, blue); }
     else
                                {           drawParts(tempX,tempY,2,2, cyan); }
                                tempX -= 0x3;
    }
    tempY -= 0x3;
                        tempX = x;
  }
}
//this function writes a stylish letter in the color of yellow
void writeLetterYellow(unsigned short x, unsigned short y, unsigned int letter){
  int i = 0;
  int j = 0;
            unsigned int tempX = x;
            unsigned int tempY = y;
            char line = 0x0;
  setCursor(tempX, tempY);
  for (i = 0; i < 8; i++)
  {
    line = font8x8_basic[letter][i];
    for (j = 0x1; j <= 0x80; j=j*0x2)
    {
     if((line & ((unsigned char) j)) != 0x0)
                                {           drawParts(tempX, tempY, 2,2, yellow); }
     else
                                {           drawParts(tempX,tempY,2,2, cyan); }
                                tempX -= 0x3;
    }
    tempY -= 0x3;
                        tempX = x;
  }
}
//this function writes a normal letter in white
void writeLetter(unsigned short x, unsigned short y, unsigned int letter){
  int i = 0;
  int j = 0;
            char line = 0x0;
  setCursor(x,y);
  for (i = 0; i < 8; i++)
  {
    line = font8x8_basic[letter][i];
    for (j = 0x1; j <= 0x80; j=j*0x2)
    {
     if((line & ((unsigned char) j)) != 0x0)
                                {
                        writeCmd(0x22);
                                                    writeDat(white);
                                }
```

```
    else
                    {
                            writeCmd(0x22);
      writeDat(black);
    }
    setCursor(--x,y);
   }
   x=x+8;
   setCursor(x,--y);
  }
}
//this function creates the frog in the up position
void createFrogUp(unsigned int x, unsigned int y, unsigned int color) //these are the coordinates
{
            unsigned int startX;
            unsigned int startY;

            startX = x+3;
            startY = y+3;

            drawParts(startX,startY,6,4,color); //draws 1st part of foot1
            drawParts(startX+29,startY,6,4,color);    //draws 1st part of foot2
            drawParts(startX+1,startY+4,4,4,color);  //draws 2nd part of foot1
            drawParts(startX+30,startY+4,4,4,color); //draws 2nd part of foot2
            drawParts(startX+1,startY+8,32,1,color); //draws 1st foot to foot
            drawParts(startX+2,startY+9,30,2,color);          //draws 2nd foot to foot
            drawParts(startX+3,startY+11,28,1,color);         //draws 3rd foot to foot
            drawParts(startX+3,startY+12,6,2,color);          //draws knee1
            drawParts(startX+25,startY+12,6,2,color);         //draws knee2
            drawParts(startX+8,startY+7,18,1,color);          //draws 1st bum
            drawParts(startX+11,startY+6,12,1,color);         //draws 2nd bum
            drawParts(startX+12,startY+5,10,1,color); //draws 3rd bum
            drawParts(startX+13,startY+4,8,1,color);          //draws 4th bum
            drawParts(startX+11,startY+12,12,6,color);        //draws main body
            drawParts(startX+6,startY+18,22,1,color);         //1st arm to arm
            drawParts(startX+4,startY+19,26,1,color);         //2nd arm to arm
            drawParts(startX+2,startY+20,30,2,color);         //3rd arm to arm
            drawParts(startX+0,startY+22,34,1,color);         //4th arm to arm
            drawParts(startX+0,startY+23,7,1,color);          //1st arm1
            drawParts(startX+27,startY+23,7,1,color);         //1st arm2
            drawParts(startX+0,startY+24,4,3,color);          //2nd arm1
            drawParts(startX+30,startY+24,4,3,color);         //2nd arm2
            drawParts(startX+0,startY+27,5,1,color);          //3rd arm1
            drawParts(startX+29,startY+27,5,1,color);         //3rd arm2
            drawParts(startX+1,startY+28,4,3,color);          //4th arm1
            drawParts(startX+29,startY+28,4,3,color);         //4th arm2
            drawParts(startX+11,startY+23,12,2,color);        //1st neck
            drawParts(startX+10,startY+25,14,4,color);        //2nd neck
            drawParts(startX+11,startY+29,12,1,black);        //1st head and glasses
            drawParts(startX+12,startY+30,10,2,color);        //2nd head
```

```
		drawParts(startX+13,startY+32,8,2,color);		//3rd head
		drawParts(startX+15,startY+34,4,1,color);		//4th head
		drawParts(startX+14,startY+28,2,3,black);		//eye1
		drawParts(startX+18,startY+28,2,3,black);		//eye2
}
//this function creates the frog in the down position
void createFrogDown(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+3;
		startY = y+2;

		drawParts(startX,startY+34,6,4,color); //draws 1st part of foot1
		drawParts(startX+29,startY+34,6,4,color);	//draws 1st part of foot2
		drawParts(startX+1,startY+30,4,4,color);	//draws 2nd part of foot1
		drawParts(startX+30,startY+30,4,4,color); //draws 2nd part of foot2
		drawParts(startX+1,startY+29,32,1,color); //draws 1st foot to foot
		drawParts(startX+2,startY+27,30,2,color);	//draws 2nd foot to foot
		drawParts(startX+3,startY+26,28,1,color);	//draws 3rd foot to foot
		drawParts(startX+3,startY+25,6,2,color);	//draws knee1
		drawParts(startX+25,startY+25,6,2,color);	//draws knee2
		drawParts(startX+8,startY+30,18,1,color);	//draws 1st bum
		drawParts(startX+11,startY+31,12,1,color);	//draws 2nd bum
		drawParts(startX+12,startY+32,10,1,color); //draws 3rd bum
		drawParts(startX+13,startY+33,8,1,color);	//draws 4th bum
		drawParts(startX+11,startY+20,12,6,color);	//draws main body
		drawParts(startX+6,startY+19,22,1,color);	//1st arm to arm
		drawParts(startX+4,startY+18,26,1,color);	//2nd arm to arm
		drawParts(startX+2,startY+16,30,2,color);	//3rd arm to arm
		drawParts(startX+0,startY+15,34,1,color);	//4th arm to arm
		drawParts(startX+0,startY+14,7,1,color);	//1st arm1
		drawParts(startX+27,startY+14,7,1,color);	//1st arm2
		drawParts(startX+0,startY+11,4,3,color);	//2nd arm1
		drawParts(startX+30,startY+11,4,3,color);	//2nd arm2
		drawParts(startX+0,startY+10,5,1,color);	//3rd arm1
		drawParts(startX+29,startY+10,5,1,color);	//3rd arm2
		drawParts(startX+1,startY+7,4,3,color);  //4th arm1
		drawParts(startX+29,startY+7,4,3,color);	//4th arm2
		drawParts(startX+11,startY+13,12,2,color);	//1st neck
		drawParts(startX+10,startY+9,14,4,color);	//2nd neck
		drawParts(startX+11,startY+8,12,1,black);	//1st head and glasses
		drawParts(startX+12,startY+6,10,2,color);	//2nd head
		drawParts(startX+13,startY+4,8,2,color);		//3rd head
		drawParts(startX+15,startY+3,4,1,color);		//4th head
		drawParts(startX+14,startY+7,2,3,black);		//eye1
		drawParts(startX+18,startY+7,2,3,black);		//eye2
}
```

```
//this function creates the frog in the left position
void createFrogLeft(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+3;
		startY = y+3;

		drawParts(startX,startY,4,6,color); //draws 1st part of foot1
		drawParts(startX,startY+29,4,6,color);	//draws 1st part of foot2
		drawParts(startX+4,startY+1,4,4,color);	//draws 2nd part of foot1
		drawParts(startX+4,startY+30,4,4,color); //draws 2nd part of foot2
		drawParts(startX+8,startY+1,1,32,color); //draws 1st foot to foot
		drawParts(startX+9,startY+2,2,30,color);		//draws 2nd foot to foot
		drawParts(startX+11,startY+3,1,28,color);		//draws 3rd foot to foot
		drawParts(startX+12,startY+3,2,6,color);		//draws knee1
		drawParts(startX+12,startY+25,2,6,color);		//draws knee2
		drawParts(startX+7,startY+8,1,18,color);		//draws 1st bum
		drawParts(startX+6,startY+11,1,12,color);		//draws 2nd bum
		drawParts(startX+5,startY+12,1,10,color); //draws 3rd bum
		drawParts(startX+4,startY+13,1,8,color);		//draws 4th bum
		drawParts(startX+12,startY+11,6,12,color);		//draws main body
		drawParts(startX+18,startY+6,1,22,color);		//1st arm to arm
		drawParts(startX+19,startY+4,1,26,color);		//2nd arm to arm
		drawParts(startX+20,startY+2,2,30,color);		//3rd arm to arm
		drawParts(startX+22,startY+0,1,34,color);		//4th arm to arm
		drawParts(startX+23,startY+0,1,7,color);		//1st arm1
		drawParts(startX+23,startY+27,1,7,color);		//1st arm2
		drawParts(startX+24,startY+0,3,4,color);		//2nd arm1
		drawParts(startX+24,startY+30,3,4,color);		//2nd arm2
		drawParts(startX+27,startY+0,1,5,color);		//3rd arm1
		drawParts(startX+27,startY+29,1,5,color);		//3rd arm2
		drawParts(startX+28,startY+1,3,4,color);		//4th arm1
		drawParts(startX+28,startY+29,3,4,color);		//4th arm2
		drawParts(startX+23,startY+11,2,12,color);		//1st neck
		drawParts(startX+25,startY+10,4,14,color);		//2nd neck
		drawParts(startX+29,startY+11,1,12,black);		//1st head and glasses
		drawParts(startX+30,startY+12,2,10,color);		//2nd head
		drawParts(startX+32,startY+13,2,8,color);			//3rd head
		drawParts(startX+34,startY+15,1,4,color);			//4th head
		drawParts(startX+28,startY+14,3,2,black);			//eye1
		drawParts(startX+28,startY+18,3,2,black);			//eye2
}

//this function creates the frog in the right position
void createFrogRight(unsigned int x, unsigned int y, unsigned int color)
{
	unsigned int startX;
		unsigned int startY;
```

```
		startX = x+3;
		startY = y+2;

		drawParts(startX+34,startY+0,4,6,color); //draws 1st part of foot1
		drawParts(startX+34,startY+29,4,6,color);	//draws 1st part of foot2
		drawParts(startX+30,startY+1,4,4,color);	//draws 2nd part of foot1
		drawParts(startX+30,startY+30,4,4,color); //draws 2nd part of foot2
		drawParts(startX+29,startY+1,1,32,color); //draws 1st foot to foot
		drawParts(startX+27,startY+2,2,30,color);	//draws 2nd foot to foot
		drawParts(startX+26,startY+3,1,28,color);	//draws 3rd foot to foot
		drawParts(startX+25,startY+3,2,6,color);	//draws knee1
		drawParts(startX+25,startY+25,2,6,color);	//draws knee2
		drawParts(startX+30,startY+8,1,18,color);	//draws 1st bum
		drawParts(startX+31,startY+11,1,12,color);	//draws 2nd bum
		drawParts(startX+32,startY+12,1,10,color); //draws 3rd bum
		drawParts(startX+33,startY+13,1,8,color);	//draws 4th bum
		drawParts(startX+20,startY+11,6,12,color);	//draws main body
		drawParts(startX+19,startY+6,1,22,color);	//1st arm to arm
		drawParts(startX+18,startY+4,1,26,color);	//2nd arm to arm
		drawParts(startX+16,startY+2,2,30,color);	//3rd arm to arm
		drawParts(startX+15,startY+0,1,34,color);	//4th arm to arm
		drawParts(startX+14,startY+0,1,7,color);	//1st arm1
		drawParts(startX+14,startY+27,1,7,color);	//1st arm2
		drawParts(startX+11,startY+0,3,4,color);	//2nd arm1
		drawParts(startX+11,startY+30,3,4,color);	//2nd arm2
		drawParts(startX+10,startY+0,1,5,color);	//3rd arm1
		drawParts(startX+10,startY+29,1,5,color);	//3rd arm2
		drawParts(startX+7,startY+1,3,4,color);  //4th arm1
		drawParts(startX+7,startY+29,3,4,color);	//4th arm2
		drawParts(startX+13,startY+11,2,12,color);	//1st neck
		drawParts(startX+9,startY+10,4,14,color);	//2nd neck
		drawParts(startX+8,startY+11,1,12,black);	//1st head and glasses
		drawParts(startX+6,startY+12,2,10,color);	//2nd head
		drawParts(startX+4,startY+13,2,8,color);		//3rd head
		drawParts(startX+3,startY+15,1,4,color);		//4th head
		drawParts(startX+7,startY+14,3,2,black);		//eye1
		drawParts(startX+7,startY+18,3,2,black);		//eye2
}

//this function draws a box of any size (x and y) in any position (x and y) with any color
void drawParts(unsigned int startX, unsigned int startY, unsigned int quantityX, unsigned int quantityY,
unsigned int color)
{
		unsigned int tempX;
		unsigned int tempY;
		unsigned int i,j;						//counters

		tempX = startX;
		tempY = startY;
```

```
		setCursor(startX, startY);
		for(i=0 ; i < quantityY; i++)
		{
			for(j=0; j < quantityX; j++)
			{
				if(tempX<240)
				{
					if(tempX>0)
					{
					writeCmd(0x22);
					writeDat(color);
					}
				}
				tempX = tempX + 1;
				setCursor(tempX,tempY);
			}
			tempY++;
			tempX = startX;
			setCursor(tempX,tempY);
		}
}

//this function creates the front end of the car which faces to the right
void createCarFrontLtoR(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+1;
		startY = y+4;

		drawParts(startX+8,startY+0,3,1,gray);	//1st wheel1
		drawParts(startX+26,startY+0,3,1,gray);	//1st wheel2
		drawParts(startX+7,startY+1,5,1,gray);	//2nd wheel1
		drawParts(startX+25,startY+1,5,1,gray);	//2nd wheel2
		drawParts(startX+6,startY+2,7,2,gray);	//3rd wheel1
		drawParts(startX+24,startY+2,7,2,gray);	//3rd wheel2
		drawParts(startX+5,startY+4,9,2,gray);	//4th wheel1
		drawParts(startX+23,startY+4,9,2,gray);	//4th wheel2
		drawParts(startX+4,startY+6,6,2,color);	//1st bottom
		drawParts(startX+2,startY+8,6,1,color);	//2nd bottom
		drawParts(startX+1,startY+9,6,13,color);		//body
		drawParts(startX+0,startY+10,1,3,gray);	//front bumper
		drawParts(startX+36,startY+10,1,3,gray);		//back bumper
		drawParts(startX+1,startY+15,4,3,yellow);		//lights
		drawParts(startX+4,startY+15,6,3,color);		//other side of lights
		drawParts(startX+2,startY+22,6,1,gray);		//1st top
		drawParts(startX+4,startY+23,6,2,cyan);	//1st windshield
		drawParts(startX+14,startY+23,6,2,color);		//1st color top
```

```
		drawParts(startX+6,startY+25,6,3,cyan); //2nd windshield
		drawParts(startX+14,startY+25,6,3,color);	//2nd color top
		drawParts(startX+8,startY+28,6,3,cyan);		//3rd windshield
		drawParts(startX+14,startY+28,6,3,color);	//3rd color top
		drawParts(startX+11,startY+31,3,2,cyan);	//4th windshield
		drawParts(startX+14,startY+31,6,2,color);	//4th color top

}
//this function creates the front end of the car which faces to the left
void createCarFrontRtoL(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+1;
		startY = y+4;

		drawParts(startX+8,startY+0,3,1,gray);	//1st wheel1
		drawParts(startX+26,startY+0,3,1,gray);	//1st wheel2
		drawParts(startX+7,startY+1,5,1,gray);	//2nd wheel1
		drawParts(startX+25,startY+1,5,1,gray);	//2nd wheel2
		drawParts(startX+6,startY+2,7,2,gray);	//3rd wheel1
		drawParts(startX+24,startY+2,7,2,gray);	//3rd wheel2
		drawParts(startX+5,startY+4,9,2,gray);	//4th wheel1
		drawParts(startX+23,startY+4,9,2,gray);	//4th wheel2
		drawParts(startX+27,startY+6,6,2,color);	//1st bottom
		drawParts(startX+29,startY+8,6,1,color);	//2nd bottom
		drawParts(startX+0,startY+10,1,3,gray);	//front bumper
		drawParts(startX+36,startY+10,1,3,gray);	//back bumper
		drawParts(startX+30,startY+9,6,13,color);	//body
		drawParts(startX+32,startY+15,4,3,yellow);	//lights
		drawParts(startX+27,startY+15,6,3,color);		//after the lights
		drawParts(startX+28,startY+22,6,1,gray);		//1st top
		drawParts(startX+26,startY+23,6,2,cyan);	//1st windshield
		drawParts(startX+3,startY+23,19,2,color);	//1st color top
		drawParts(startX+22,startY+25,8,3,cyan);	//2nd windshield
		drawParts(startX+16,startY+25,6,3,color);	//2nd color top
		drawParts(startX+22,startY+28,6,3,cyan);		//3rd windshield
		drawParts(startX+16,startY+28,6,3,color);	//3rd color top
		drawParts(startX+22,startY+31,3,1,cyan);	//4th windshield
		drawParts(startX+16,startY+31,6,2,color);	//4th color top
}
//this function creates the entire car that faces to the right
void createCarLtoR(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+1;
		startY = y+4;
```

```
		drawParts(startX+8,startY+0,3,1,gray);	//1st wheel1
		drawParts(startX+26,startY+0,3,1,gray);	//1st wheel2
		drawParts(startX+7,startY+1,5,1,gray);	//2nd wheel1
		drawParts(startX+25,startY+1,5,1,gray);	//2nd wheel2
		drawParts(startX+6,startY+2,7,2,gray);	//3rd wheel1
		drawParts(startX+24,startY+2,7,2,gray);	//3rd wheel2
		drawParts(startX+5,startY+4,9,2,gray);	//4th wheel1
		drawParts(startX+23,startY+4,9,2,gray);	//4th wheel2
		drawParts(startX+4,startY+6,29,2,color);		//1st bottom
		drawParts(startX+2,startY+8,33,1,color);		//2nd bottom
		drawParts(startX+1,startY+9,35,13,color);		//body
		drawParts(startX+0,startY+10,1,3,gray);	//front bumper
		drawParts(startX+36,startY+10,1,3,gray);		//back bumper
		drawParts(startX+1,startY+15,4,3,yellow);		//lights
		drawParts(startX+2,startY+22,33,1,gray);			//1st top
		drawParts(startX+4,startY+23,10,2,cyan);		//1st windshield
		drawParts(startX+14,startY+23,20,2,color);		//1st color top
		drawParts(startX+6,startY+25,8,3,cyan);	//2nd windshield
		drawParts(startX+14,startY+25,19,3,color);		//2nd color top
		drawParts(startX+8,startY+28,6,3,cyan);		//3rd windshield
		drawParts(startX+14,startY+28,18,3,color);		//3rd color top
		drawParts(startX+11,startY+31,3,2,cyan);		//4th windshield
		drawParts(startX+14,startY+31,16,2,color);		//4th color top
}

//this function creates the entire car that faces to the left
void createCarRtoL(unsigned int x, unsigned int y, unsigned int color)
{
		unsigned int startX;
		unsigned int startY;

		startX = x+1;
		startY = y+4;

		drawParts(startX+8,startY+0,3,1,gray);	//1st wheel1
		drawParts(startX+26,startY+0,3,1,gray);	//1st wheel2
		drawParts(startX+7,startY+1,5,1,gray);	//2nd wheel1
		drawParts(startX+25,startY+1,5,1,gray);	//2nd wheel2
		drawParts(startX+6,startY+2,7,2,gray);	//3rd wheel1
		drawParts(startX+24,startY+2,7,2,gray);	//3rd wheel2
		drawParts(startX+5,startY+4,9,2,gray);	//4th wheel1
		drawParts(startX+23,startY+4,9,2,gray);	//4th wheel2
		drawParts(startX+4,startY+6,29,2,color);		//1st bottom
		drawParts(startX+2,startY+8,33,1,color);		//2nd bottom
		drawParts(startX+1,startY+9,36,13,color);		//body
		drawParts(startX+0,startY+10,1,3,gray);	//front bumper
		drawParts(startX+36,startY+10,1,3,gray);		//back bumper
		drawParts(startX+33,startY+15,4,3,yellow);		//lights
		drawParts(startX+2,startY+22,33,1,gray);		//1st top
```

```
                drawParts(startX+22,startY+23,10,2,cyan);       //1st windshield
                drawParts(startX+3,startY+23,20,2,color);       //1st color top
                drawParts(startX+22,startY+25,8,3,cyan);        //2nd windshield
                drawParts(startX+4,startY+25,19,3,color);       //2nd color top
                drawParts(startX+22,startY+28,6,3,cyan);                //3rd windshield
                drawParts(startX+5,startY+28,18,3,color);       //3rd color top
                drawParts(startX+22,startY+31,3,2,cyan);        //4th windshield
                drawParts(startX+7,startY+31,16,2,color);       //4th color top
}


//this function clears the back end of the car that faces to the right
void clearCarLtoR(unsigned int tempX, unsigned int tempY)
{
        drawParts(tempX+32,tempY+36,8,3,black);     //top back1
        drawParts(tempX+34,tempY+33,7,3,black);     //top back2
        drawParts(tempX+35,tempY+30,7,3,black);     //top back3
        drawParts(tempX+36,tempY+28,7,2,black);     //top back4
        drawParts(tempX+37,tempY+27,7,1,black);     //top back5
        drawParts(tempX+37,tempY+18,9,9,black);     //body1
        drawParts(tempX+38,tempY+15,9,3,black);     //bumper1
        drawParts(tempX+38,tempY+14,7,1,black);     //bottom1
        drawParts(tempX+37,tempY+13,7,1,black);     //bottom2
        drawParts(tempX+35,tempY+10,7,3,black);     //bottom3

        drawParts(tempX,tempY,40,10,black);                     //for the wheels
}

//this function clears the back end of the car that faces to the left
void clearCarRtoL(unsigned int tempX, unsigned int tempY)
{
        drawParts(tempX-1,tempY+36,9,3,black);      //top back1
        drawParts(tempX-2,tempY+33,8,3,black);      //top back2
        drawParts(tempX-2,tempY+30,7,3,black);      //top back3
        drawParts(tempX-2,tempY+28,6,2,black);      //top back4
        drawParts(tempX-2,tempY+27,6,1,black);      //top back5
        drawParts(tempX-5,tempY+18,7,9,black);      //body1
        drawParts(tempX-5,tempY+14,7,4,black);      //bumper1
        drawParts(tempX-7,tempY+14,8,2,black);      //bottom1
        drawParts(tempX-4,tempY+13,7,1,black);      //bottom2
        drawParts(tempX-2,tempY+10,6,3,black);      //bottom3

        drawParts(tempX,tempY,40,10,black);                     //for the wheels
}

//this function creates the menu screen and loops to either change the frog color
//or starts the game. We wait until either action is performed. We also update
//a car to continually drive by.
void menuScreen(void)
{
```

```
initAll();
//random image
createCarLtoR(180,215,blue);

//title
writeLetterBig(220, 305, 0x46);  //F
writeLetterBig(180, 300, 0x52);  //R
writeLetterBig(140, 295, 0x4F);          //O
writeLetterBig(100, 290, 0x47);          //G

writeLetterBig(180, 65, 0x4A);           //J
writeLetterBig(140, 60, 0x55);           //U
writeLetterBig(100, 55, 0x4D);           //M
writeLetterBig(60, 50, 0x50);                    //P

//high score
writeLetterMagenta(200,210, 0x48);       //H
writeLetterBlue(175, 210, 0x69);                 //i
writeLetterYellow(150,210,0x67);                 //g
writeLetterMagenta(125,210, 0x68);       //h

writeLetterYellow(210,185, 0x53);                //S
writeLetterBlue(185,185,0x63);                   //c
writeLetterMagenta(160,185,0x6F);                //o
writeLetterBlue(135,185,0x72);                   //r
writeLetterYellow(110,185,0x65);                 //e

writeLetterBlue(70,195,finalScoreTen+0x30);
writeLetterBlue(45,195,finalScoreOne+0x30);

writeLetter(75,165,0x70);                                        //p
writeLetter(65,165,0x6F);                                        //o
writeLetter(55,165,0x69);                                        //i
writeLetter(45,165,0x6E);                                        //n
writeLetter(35,165,0x74);                                        //t
writeLetter(25,165,0x73);                                        //s

writeLetter(200,85,0x77);                                        //w
writeLetter(190,85,0x69);                                        //i
writeLetter(180,85,0x6E);                                        //n
writeLetter(170,85,0x73);                                        //s
writeLetter(160,85,0x3A);                                        //:
writeLetter(145,85,numOfWins+0x30);

writeLetter(120,85,0x6C);                                        //l
writeLetter(110,85,0x6F);                                        //o
writeLetter(100,85,0x73);                                        //s
writeLetter(90,85,0x73);                                         //s
writeLetter(80,85,0x65);                                         //e
writeLetter(70,85,0x73);                                         //s
```

```
	writeLetter(60,85,0x3A);	//:
	writeLetter(45,85,numOfLosses +0x30);

	while(1)
	{
		createFrogUp(20,230,FROGCOLOR);
		menuCar = updateCarLtoR(menuCar,100,magenta,250);
		//menuPosition = updateCarLtoR(menuPosition);
		checkBtnsMenu();
	}
}

//this function checks to see if any buttons are pushed during the menu screen.
void checkBtnsMenu(void)
{
	unsigned int checker;
	unsigned int i=0x0;
	checker = PE[0x3FC];
	checker &= 0x1;
	if(checker == 0x0)
	{
		for(i=0;i>700;i++)
		{
			delayMS(200);
		}
		gameScreen();
	}

	checker = PE[0x3FC];
	checker &= 0x2;
	if(checker == 0x0)
	{
		for(i=0;i>700;i++)
		{
			delayMS(200);
		}
		gameScreen();
	}

	checker = PE[0x3FC];
	checker &= 0x4;
	if(checker == 0x0)
	{
		delayMS(200);
		delayMS(200);
		newColor++;
		longDelay(250);
	}

	checker = PE[0x3FC];
```

```
	checker &= 0x10;
	if(checker == 0x0)
	{
		delayMS(200);
		delayMS(200);
		newColor--;
		longDelay(250);
	}


	if(newColor ==0x0)
		newColor = 0x4;
	if(newColor >=0x5)
		newColor = 0x0;
	if(newColor == 1)
		FROGCOLOR = green;
	if(newColor == 2)
		FROGCOLOR = red;
	if(newColor == 3)
		FROGCOLOR = yellow;
	if(newColor == 4)
		FROGCOLOR = cyan;

	delayMS(200);
	delayMS(200);
}
//this function updates the car that travels to the right
int updateCarLtoR(int positionX, unsigned int positionY, unsigned int color, int length)
{
	unsigned int tempY = positionY;
	if(positionX<-50)
	{
		positionX = length;
	}
	//delayMS(50);
	clearCarLtoR(positionX,tempY);
	createCarFrontLtoR(positionX, tempY, color);
	positionX -= 0x6;

	delayMS(25);
	return positionX;
}


//this function updates the car that travels to the left
int updateCarRtoL(int positionX,unsigned int positionY, unsigned int color, int length)
{
	unsigned int tempY = positionY;
	if(positionX>250)
	{
```

```
		positionX = length;
	}
	//delayMS(50);
	clearCarRtoL(positionX,tempY);
	createCarFrontRtoL(positionX, tempY, color);
	positionX += 0x6;

	delayMS(25);
	return positionX;
}

//this function creates the game screen, updates the cars and checks for button pushes
void gameScreen(void)
{

	clearLCD(black);
	drawParts(60,0,1,320,red);		//right boundary
	drawParts(180,0,1,320,red);		//left boundary
	drawParts(61,0,119,40,gray);	//starting area
	writeLetterBig(135,319,0x57);	//end area

	writeLetterYellow(120,275, 0x53);		//S
	writeLetterBlue(140,235,0x74);		//t
	writeLetterMagenta(120,195,0x61);		//a
	writeLetterBlue(140,155,0x72);		//r
	writeLetterYellow(120,115,0x74);		//t
	writeLetterMagenta(140,75,0x33);		//3

	createFrogUp(FROGX,FROGY,FROGCOLOR);
	while(1)
	{
		updateAllCars();
		checkAll();
	}
}

//this function updates all the cars by calling each row
void updateAllCars(void)
{
	firstRow();
	secondRow();
	thirdRow();
	fourthRow();
	fifthRow();
	sixthRow();
}

//this function updates the first row of cars
void firstRow(void)
{
```

```
	//this row goes from left to right
	car1A = updateCarLtoR(car1A,car1Y,blue,250);
	car1B = updateCarLtoR(car1B,car1Y,red,250);
}

//this function updates the second row of cars
void secondRow(void)
{
	//this row goes from right to left
	car2A = updateCarRtoL(car2A,car2Y,yellow,-60);
	car2B = updateCarRtoL(car2B,car2Y,magenta,-60);
}

//this function updates the third row of cars
void thirdRow(void)
{
	//this row goes from right to left
	car3A = updateCarRtoL(car3A,car3Y,red,-40);
	car3B = updateCarRtoL(car3B,car3Y,magenta,-40);
}

//this function updates the fourth row of cars
void fourthRow(void)
{
	//this row goes from left to right
	car4A = updateCarLtoR(car4A,car4Y,green,270);
	car4B = updateCarLtoR(car4B,car4Y,blue,270);
}

//this function updates the fifth row of cars
void fifthRow(void)
{
	//this row goes from left to right
	car5A = updateCarLtoR(car5A,car5Y,magenta,250);
	car5B = updateCarLtoR(car5B,car5Y,green,250);
}

//this function updates the sixth row of cars
void sixthRow(void)
{
	//this row goes from right to left
	car6A = updateCarRtoL(car6A,car6Y,yellow,-50);
	car6B = updateCarRtoL(car6B,car6Y,blue,-50);
	car6C = updateCarRtoL(car6C,car6Y,green,-50);
}



//this function updates the before the frog can move and counts down
//once we count down then GO appears and the move flag gets set
```

```
void checkStart(void)
{
	if(countForStart == 6)
	{	writeLetterMagenta(140,75,0x32);		}//2
	if(countForStart == 11)
	{	writeLetterMagenta(140,75,0x31);		}//1
	if(countForStart == 16)
	{		writeLetterMagenta(140,75,0x30);	}//0
	if(countForStart == 22)
	{
		writeLetterBlue(160,35,0x47);				//G
		writeLetterBlue(100,35,0x4F);				//O
		startFlag = 1;
	//can begin moving
	}
	countForStart++;
}

//this function checks buttons, if you win or if you lose
void checkAll()
{
	if(startFlag>0x0)
	{	checkBtnsGame();
		scoreCounter++;
		if(scoreCounter>5)
		{
			scoreCounter = 0x0;
			timeScore++;
		}
	}
	if(countForStart <=50)
	{	checkStart();	}
	checkWin();
	checkLoss();

}
//this function checks to see if the frog is on the winning position
void checkWin()
{
		//calcScore(void);
		if(FROGX == 100 && FROGY ==280)
		{
			calcHighScore();
			clearLCD(black);
			writeLetter(180,300,0x53);				//S
			writeLetter(170,300,0x63);				//c
			writeLetter(160,300,0x6F);				//o
			writeLetter(150,300,0x72);				//r
			writeLetter(140,300,0x65);				//e
			writeLetter(110,300,scoreTen+0x30);	//score value
```

```
                        writeLetter(100,300,scoreOne+0x30);    //score value

                        writeLetterBig(180,260,0x59);                          //Y
                        writeLetterBig(140,245,0x6F);                          //o
                        writeLetterBig(100,230,0x75);                          //u
                        writeLetterBig(170,180,0x57);                          //W
                        writeLetterBig(130,165,0x49);                          //I
                        writeLetterBig(90,150,0x4E);                           //N
                        createFrogLeft(160,60,red);
                        createFrogRight(40,60,blue);
                        createFrogUp(100,40,green);
                        numOfWins++;
                        while(1)
                        {
                                countEnd++;
                                delayMS(200);
                                if(countEnd >= 4000)//wait for a while then go back to menu screen
                                {
                                        clearLCD(black);
                                        menuScreen();
                                }
                        }
                }
}

//this function initializes everything when starting the menu screen
//this includes flags, positions, timers, etc
void initAll()
{
        menuCar = 250;

        FROGX = 100;
        FROGY = 0;
        FROGCOLOR = green;
        countForStart = 0;              //counter for start number
        startFlag = 0;                          //flag for start
        countEnd = 0;                           //counter for end wait
        loseFlag = 0;
        movesScore = 0x0;
        timeScore = 0x0;
        scoreCountDown = 115;
        finalScore = 0x0;


        //global variables for the cars
        car1Y =40;
        car2Y = 80;
        car3Y = 120;
        car4Y = 160;
        car5Y = 200;
```

```
	car6Y = 240;
	car1A =260;
	car1B =360;
	car2A =-80;
	car2B =-220;
	car3A =-40;
	car3B = -200;
	car4A = 280;
	car4B = 370;
	car5A = 240;
	car5B = 420;
	car6A = -60;
	car6B = -160;
	car6C = -230;

}
//this function checks to see if the car's position is the same as the frog's position
void checkCar(int carX, unsigned int carY)
{
	if(FROGY == carY-20)
	{
		if(FROGX == carX)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX)
		{ loseFlag = 1;	}
		if(FROGX == carX+2)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+2)
		{	loseFlag = 1;	}
		if(FROGX == carX+4)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+4)
		{	loseFlag = 1;	}
		if(FROGX == carX+6)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+6)
		{	loseFlag = 1;	}
		if(FROGX == carX+8)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+8)
		{	loseFlag = 1;	}
		if(FROGX == carX+10)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+10)
		{	loseFlag = 1;	}
		if(FROGX == carX+12)
		{	loseFlag = 1;	}
		if(FROGX+40 == carX+12)
		{	loseFlag = 1;	}
		if(FROGX == carX+14)
```

```
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+14)
        {       loseFlag = 1;   }
        if(FROGX == carX+18)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+18)
        {       loseFlag = 1;   }
        if(FROGX == carX+24)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+24)
        {       loseFlag = 1;   }
        if(FROGX == carX+28)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+28)
        {       loseFlag = 1;   }
        if(FROGX == carX+30)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+30)
        {       loseFlag = 1;   }
        if(FROGX == carX+36)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+36)
        {       loseFlag = 1;   }
}
if(FROGY == carY)
{
        if(FROGX == carX)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX)
        { loseFlag = 1;   }
        if(FROGX == carX+2)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+2)
        {       loseFlag = 1;   }
        if(FROGX == carX+4)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+4)
        {       loseFlag = 1;   }
        if(FROGX == carX+6)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+6)
        {       loseFlag = 1;   }
        if(FROGX == carX+8)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+8)
        {       loseFlag = 1;   }
        if(FROGX == carX+10)
        {       loseFlag = 1;   }
        if(FROGX+40 == carX+10)
        {       loseFlag = 1;   }
```

```
            if(FROGX == carX+12)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+12)
            {           loseFlag = 1;      }
            if(FROGX == carX+14)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+14)
            {           loseFlag = 1;      }
            if(FROGX == carX+18)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+18)
            {           loseFlag = 1;      }
            if(FROGX == carX+24)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+24)
            {           loseFlag = 1;      }
            if(FROGX == carX+28)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+28)
            {           loseFlag = 1;      }
            if(FROGX == carX+30)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+30)
            {           loseFlag = 1;      }
            if(FROGX == carX+36)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+36)
            {           loseFlag = 1;      }
}
if(FROGY == carY+20)
{

            if(FROGX == carX)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX)
            { loseFlag = 1;    }
            if(FROGX == carX+2)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+2)
            {           loseFlag = 1;      }
            if(FROGX == carX+4)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+4)
            {           loseFlag = 1;      }
            if(FROGX == carX+6)
            {           loseFlag = 1;      }
            if(FROGX+40 == carX+6)
            {           loseFlag = 1;      }
            if(FROGX == carX+8)
            {           loseFlag = 1;      }
```

```
                if(FROGX+40 == carX+8)
                {       loseFlag = 1;   }
                if(FROGX == carX+10)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+10)
                {       loseFlag = 1;   }
                if(FROGX == carX+12)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+12)
                {       loseFlag = 1;   }
                if(FROGX == carX+14)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+14)
                {       loseFlag = 1;   }
                if(FROGX == carX+18)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+18)
                {       loseFlag = 1;   }
                if(FROGX == carX+24)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+24)
                {       loseFlag = 1;   }
                if(FROGX == carX+28)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+28)
                {       loseFlag = 1;   }
                if(FROGX == carX+30)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+30)
                {       loseFlag = 1;   }
                if(FROGX == carX+36)
                {       loseFlag = 1;   }
                if(FROGX+40 == carX+36)
                {       loseFlag = 1;   }
        }
}

//this function checks each car's position with the frog's position using the checkCar function
void checkLoss(void)
{
        checkCar(car1A,car1Y);
        checkCar(car1B,car1Y);
        checkCar(car2A,car2Y);
        checkCar(car2B,car2Y);
        checkCar(car3A,car3Y);
        checkCar(car3B,car3Y);
        checkCar(car4A,car4Y);
        checkCar(car4B,car4Y);
        checkCar(car5A,car5Y);
        checkCar(car5B,car5Y);
```

```
	checkCar(car6A,car6Y);
	checkCar(car6B,car6Y);
	checkCar(car6C,car6Y);

	if(loseFlag >0x0)
	{
		numOfLosses++;
		lossState();
	}
}

//this function triggers when the frog losses and displays the lose screen
void lossState(void)
{
	clearLCD(black);

	writeLetterBlue(180,240,0x47);		//G
	writeLetterBlue(150,240,0x41);		//A
	writeLetterBlue(120,240,0x4D);		//M
	writeLetterBlue(90,240,0x45);				//E
	writeLetterMagenta(180,180,0x4F);	//O
	writeLetterMagenta(150,180,0x56);	//V
	writeLetterMagenta(120,180,0x45);	//E
	writeLetterMagenta(90,180,0x52);	//R

	createCarLtoR(185,70,blue);
	createCarLtoR(155,25,magenta);
	createCarRtoL(25,50,red);
	createCarRtoL(115,60,yellow);
	createCarRtoL(60,105,blue);

	while(1)
	{
		countEnd++;
		delayMS(200);
		if(countEnd >= 4000)//wait for a while then go back to menu screen
		{
			clearLCD(black);
			menuScreen();
		}
	}
}

//this function checks the buttons of the frog when in the game screen
void checkBtnsGame(void)
{
		unsigned int tester;
		tester = PE[0x3FC];
		tester &= 0x1;
		if(tester == 0x0)
```

```
		{
			movesScore++;
			drawParts(FROGX,FROGY,42,40,black);
			if(FROGY<280)
			{	FROGY += 20; }
			createFrogUp(FROGX,FROGY,FROGCOLOR);

		}

		tester = PE[0x3FC];
		tester &= 0x2;
		if(tester == 0x0)
		{
			movesScore++;
			drawParts(FROGX,FROGY,42,40,black);
			if(FROGY>0x0)
			{	FROGY -= 20; }
			createFrogDown(FROGX,FROGY,FROGCOLOR);
		}

		tester = PE[0x3FC];
		tester &= 0x4;
		if(tester == 0x0)
		{
			movesScore++;
			drawParts(FROGX,FROGY,42,40,black);
			if(FROGX<140)
			{	FROGX += 20; }
			createFrogLeft(FROGX,FROGY,FROGCOLOR);
		}

		tester = PE[0x3FC];
		tester &= 0x10;
		if(tester == 0x0)
		{
			movesScore++;
			drawParts(FROGX,FROGY,42,40,black);
			if(FROGX>60)
			{	FROGX -= 20; }
			createFrogRight(FROGX,FROGY,FROGCOLOR);
		}
}

//this functino delays for a long time
void longDelay(unsigned int number)
{
	unsigned int i,j;
	for(i=0;i<1000;i++)
	{	for(j=0;j<number;j++){}}
```

```
}

void calcHighScore(void)
{
	finalScore = scoreCountDown - movesScore - timeScore;
	oldScore = scoreOne + scoreTen*10;

	scoreTen = finalScore/10;
	scoreOne = finalScore - scoreTen*10;
	if(finalScore>oldScore)
	{
		finalScoreOne = scoreOne;
		finalScoreTen = scoreTen;
	}
}
```