EVB Pin			Port Bit	Bit Addresses & Labels
1	2	1.		
		2.		
3	4	3.		
		4.		
5	6	5.	P1.7	potentiometer
		6.		
7	8	7.	3.3V	3.3 Volts
		8.		
9	10	9.		
		10.	P1.2	Motor
11	12	11.	P1.3	LED
		12.	P1.0	Steering
13	14	13.		
		14.	P0.6	SDA
15	16	15.	P0.7	SCL
		16.		
17	18	17.		
		18.		
19	20	19.		
		20.	P0.0	TX0
21	22	21.	P0.1	RX0
		22.		
23	24	23.		
		24.		
25	26	25.		
		26.		
27	28	27.		
		28.		
29	30	29.		
		30.		
31	32	31.	P3.6	SS1
		32.	P3.7	SS2
33	34	33.		
		34.	P3.5	BILED red
35	36	35.		
		36.	P3.3	BILED green
37	38	37.		
		38.		
39	40	39.		
		40.		

41 ⟷ 60

Software Initializations

A) Port I/0

```
P1MDOUT |= 0x0C
P1MDOUT &= ~0x80
P3MDOUT &= 0x80
P3MDOUT |= 0x28
P3 |= 0x80
P1MDIN &= ~0x80
P1 |= 0x80
```

B) Timers

C) Interrupts

```
EIE1 |= 0x08
EA = 1
```

D) A/D

```
REF0CN = 0x03
ACD1CF |= 0x01
ACD1CN =0x80
```

E) PCA

```
PCA0CN = 0x40
PCA0CR = 0x81
PCA0CPM0 = 0xC2
PCA0CPM2 = 0xC2
```

F) XBAR

```
XBR0 = 0x27
```

G) I2C

```
SMB0CR = 0x93
ENSMB = 1
```

File needed

```
#include <c8051_SDCC.h>
#include <stdlib.h>// needed for abs function
#include <stdio.h>
#include <i2c.h>
```

8051 initialize functions

```
void Port_Init(void);
void PCA_Init (void);
void SMB_Init (void);
void Interrupt_Init(void); void PCA_ISR ( void ) __interrupt 9;
void read_accel (void); //Sets global variables gx & gy
void set_servo_PWM (void);
void set_drive_PWM(void);
void updateLCD(void);
void set_gains(void); // function which allow operator to set feedback gains
```

```
//define global variables
unsigned int PW_CENTER = ____;
unsigned int PW_RIGHT = ____;
unsigned int PW_LEFT = ____;
unsigned int SERVO_PW = ____;
unsigned int SERVO_MAX= _____;
unsigned int SERVO_MIN= _____;
unsigned int  heading;
unsigned int  range;
unsigned int  light;
int compass_adj = 0;          // correction value from compass
```

```
int range_adj = 0;            // correction value from ranger
unsigned char r_count;        // overflow count for range
unsigned char h_count;        // overflow count for heading
unsigned char print_count;    // overflow count for printing

__sbit __at ____ RUN // a slide switch
__sbit__at ____ BILED0
__sbit__at ____ BILED1
```

Main function

Declare local variables

None

Funcion initialization

Do infinite while loop

Print battery voltage for check

if run out of battery

charge the battery

else

if (run switch is off)

Set the motor stop

Set the steer parallel to the car

BILED is red

Else if (run switch is on)

Set gain first (only once)

If (enough overflows to update accel)

```
read_accels();
set_servo_PWM(); // set the servo PWM
set_drive_PWM(); // set drive PWM
new_accels = 0; //set the flag off
a_count = 0; //clear the accel counts
```

```
                    if (enough overflows to update LCD)
                        updateLCD(); // display values
                        new_lcd = 0;
                        lcd_count = 0;
        finish the loop
end main function



void PCA_ISR ( void ) __interrupt 9 {
        if (CF) {
            CF = 0; // clear overflow indicator
            a_count++;
            if(a_count>=____) {
                new_accel=1;
                a_count = 0;
            }
            lcd_count++;
            if (lcd_count>=____) {
                new_lcd = 1;
                lcd_count = 0;
            }
            PCA0 = PCA_start;
        } // handle other PCA interrupt sources
        PCA0CN &= 0xC0;
}
```