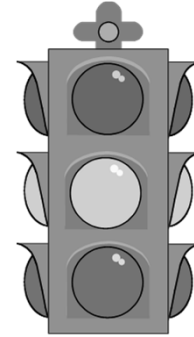




EGN1935: ECE (Ad)Ventures

Today's Menu

- Near Infrared Sensors
- CdS Cells
- Programming Simple Behaviors



University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

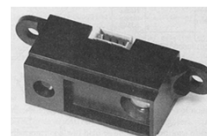
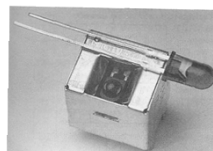
1



EGN1935: ECE (Ad)Ventures

Near-Infrared Sensors

- Infrared (IR) Sensors
 - > Near-infrared proximity sensors are called IRs for short. These devices are insensitive to the long infrared wavelengths detected by pyro-electric sensors, instead being sensitive in the range just below visible light, often around the 880 nanometer (nm) wavelength. The human eye cannot see this light, but charge coupled (CCD) cameras (e.g., camcorders) are sensitive to it. An infrared emitter detector pair can be built from a *Sharp* detector and a simple IR LED (available from Radio Shack).
 - > The Sharp detector responds to a modulated carrier signal (rapidly turning on and off the transmitter) of 40 KHz emitted by the IR LED. This means that the would be user of this detector must blink the LED in a certain pattern such that the detector will respond. This modulated carrier protocol increases the signal-to-noise ratio—the source of the light can easily be detected from varying background illumination even if the actual amount of modulated light is very small.



University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

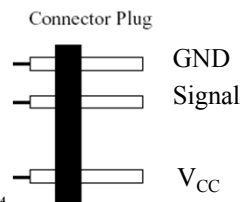
2



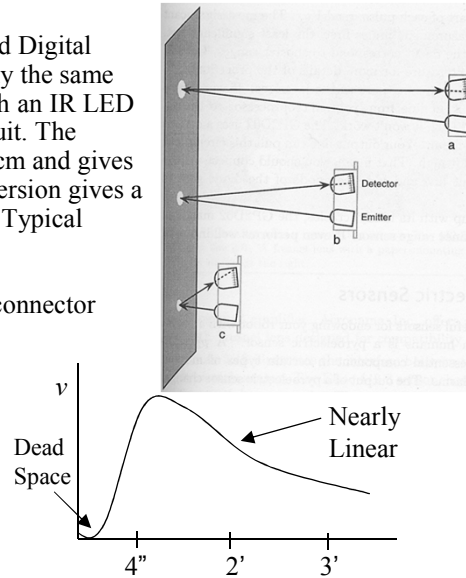
EGN1935: ECE (Ad)Ventures

Infrared Sensors

- Infrared (IR) Sensors
 - > IR sensors are available in Analog and Digital versions from Sharp for approximately the same price, \$8-\$16. They are packaged with an IR LED and a photo-receiver in the same circuit. The digital version detects up to about 16cm and gives a logic 1 or 0 as output. The analog version gives a value that is proportional to distance. Typical models include GP2D* where
 *={02,120,12,05,15}
 - > Interface consists of a simple 3 wire connector with V_{CC} , GND and Signal



University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz



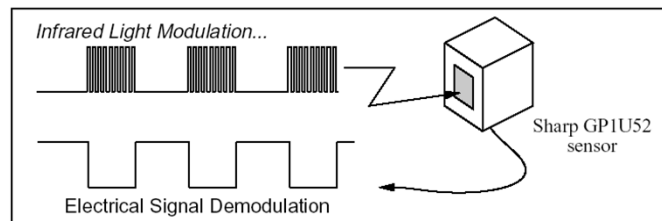
3



EGN1935: ECE (Ad)Ventures

Infrared Sensors

- Infrared (IR) Sensors
 - > Sharp cans are not as readily available as they once were, but one can still find them in eBay and Tower Hobbies. They cost about \$3 each.



University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

4

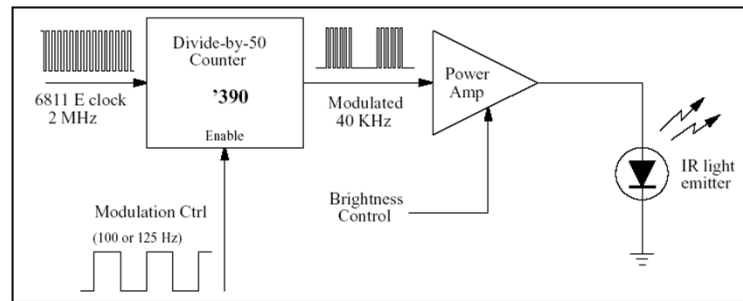


EGN1935: ECE (Ad)Ventures

Infrared Sensors

- Infrared (IR) Sensors

- > MIL has published a document with instructions to hack a Sharp can to obtain an analog detector yielding values in an A/D pin between 80 and 140. Thus, together with a \$0.50 IR LED, a ranging sensor can be constructed for under \$5.



University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

5



EGN1935: ECE (Ad)Ventures

Sensors

- Sensor Types: Two basic types Analog & Digital.
 - Digital sensors are intended to be plugged into the digital ports (pins) of a μC and always return either “0” or “1.” For digital sensors “0” or LOW or 0v or 0 and “1” or HIGH or 5v or 255
 - Analog sensors are connected to the ADC input pins in a μC
 - The analog to digital converter (ADC) in the robot board (μC) convert the input voltage to a n -bit value, where n is usually 8, 10, or 12 bits. In the 1935 robots, the Atmega128 chip uses 10 bits.
 - The input voltage must be between 0v and 5v (newer μCs use 3.3v)
 - For 2 bits: the binary values are $00_2, 01_2, 10_2, 11_2$ (0,1,2,3 in decimal). There are 4 total values $\therefore$ a 5v value is represented as $5v \div 4 = 1.25v$ per number (0v for 00_2 , 1.25v for 01_2 , 2.5v for 10_2 & 3.75v for 11_2)
 - For 8-bits: $0 \leq \text{value} \leq 255$ (2^8-1); for 10-bits: $0 \leq \text{value} \leq 1023$ ($2^{10}-1$)
 - For an 8-bit ADC (2^8 is 256 symbols) $\therefore 0v \rightarrow 0$ & $5v \rightarrow 255$ (2^8-1)
 - For a 10-bit ADC (2^{10} is 1024 symbols), $0v \rightarrow 0$ & $5v \rightarrow 1023$ ($2^{10}-1$)

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

6



EGN1935: ECE (Ad)Ventures

Simple Behaviors

- Simple Behavior Software

Surprisingly simple behaviors can be accomplished with the sensors and hardware we have seen in class.

- Motors

- > We'll assume the existence of a *motor(m,s)* function where argument $m = \{0 \text{ or } 1\}$ for the left or the right motor, respectively, and argument $s = \text{speed}$, which is a value between -100 to +100, as a percent of full speed.

motor(0,100) means move the left motor 100% forward

motor(1,-100) means move the right motor 100% backward

motor(0,0) means stop the left motor

- > Alternatively, we could create functions: forward, reverse, left, right and speed which yield equivalent functionality.

- > *init_motor()* turns on the PWM system

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

7



EGN1935: ECE (Ad)Ventures

Simple Behaviors

- ADC

- > We'll assume the existence of an *analog(c)* routine where c is an integer representing the available channel in the A/D system.

– *analog(0)* returns the 10-bit integer value in ADC channel 0

– *init_analog()* turns on the ADC system

- Timing

- > We'll assume a timing routine to gives us a desired delay, *delay(tms)* where tms is an integer value representing time in ms (1/1000 of a second)
- *delay(500)* wait 500 ms (1/2 a second) before continuing

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

8



EGN1935: ECE (Ad)Ventures

Simple Behaviors

• Printing

- > We'll assume we'll have a way to print values to the LCD
 - Function `lcd_int(integer)`; prints an integer to the LCD
 - Function `lcd_string("string")`; prints a constant string to the LCD
 - Function `lcd_clear()`; clears the LCD
 - Function `lcd_row(0 or 1)`; determines which line to write on the LCD
 - Function `init_lcd()`; initializes the LCD system

• Conditionals

- > We will also assume we have a way to test a condition (usually a sensor value) and issue an actuator command which might change depending on the condition. This is analogous to an *if* statement in a high-level language.

```
if (left_ir > 125) /* If the Left IR indicates we are close */
    speedr = -100; /* to an obstacle, turn right */
```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

9



EGN1935: ECE (Ad)Ventures

Simple Behaviors

• Testing the sensors

- > We should always test the values of the sensors and actuators (motors) under software control to verify the correct operation of our sensory systems under run-time conditions
- > For IRs and CdS cells, we should connect them into the AD system and display the values sensed on the LCD. We need to find out what is the value in the AD corresponding to too close.
- > For motors and servos, we should verify correct operation & measure timing for various maneuvers.

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

10



EGN1935: ECE (Ad)Ventures Testing Motors 1

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Motor_Test_1\MotTest1.c - Notepad2
File Edit View Settings ?
1 /*****
2 *
3 * Title:      EGN-1935 Motor Test 1
4 * Programmer: A. Antonio Arroyo
5 * Date:      February 16, 2011
6 * Version:   1.0
7 *
8 * Description:
9 * The robot will wait 5 seconds and then turns on both motors,
10 * the left motor forward, the right motor in reverse
11 *****/
12
13 /***** Includes *****/
14 #include "aaa.h"
15
16 /***** Prototypes *****/
17
18 /***** Globals *****/
19
20 int main(void) {
21     int j=0, value=MAX_SPEED;
22
23     init_system(); // initialize Ports A, C, E
24     init_motor(); // initialize PPFC PWM on Timer 1
25     init_lcd(); // set lcd in 4 bit mode, 2-line mode, with blinking cursor
26
27     // 1234567890ABCDEF
28     lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
29     lcd_row(1);
30     // 1234567890ABCDEF
31     lcd_string("Motor Test 1"); // Heading line 2 for LCD
32
33     // Delay a few seconds ~5s
34
35     // ***** Main Loop *****
36
37     // ***** End of Main Loop *****
38
39     return 0;
40 }

```

University of Florida, EGN 1935—
© Drs. A. A. Arroyo & E. M. Schwartz

Ln 51: 66 Col 20 Sel 0 1.88 KB ANSI CR+LF INS C/C++ Source Code

11



EGN1935: ECE (Ad)Ventures Testing Motors 1

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Motor_Test_1\MotTest1.c - Notepad2
File Edit View Settings ?
34     delay(5000);
35
36     lcd_clear();
37     // 1234567890ABCDEF
38     lcd_string("Left/Right Motor");
39
40     for (j = 0; j < 100; j++){ // finite loop
41         motor(LEFT_MOTOR, value);
42         motor(RIGHT_MOTOR, -value);
43
44         // Update LCD Panel
45         lcd_row(1);
46         lcd_string("L:");
47         lcd_int(value);
48         lcd_string(" R:");
49         lcd_int(-value);
50         lcd_string(" ");
51         lcd_int(j);
52
53         // complement pin PA0-PA1 and write back
54         if (PINA & 0x01) (PORTA=PINA&0xFE; PORTA=PINA|0x02;);
55         else (PORTA=PINA&0xFD; PORTA=PINA|0x01;);
56
57         // Delay about ~100ms
58         delay(100);
59     } // end for loop
60
61     motor(LEFT_MOTOR, 0); // Stop the left motor
62     motor(RIGHT_MOTOR, 0); // Stop the right motor
63     return 0;
64 } // end main
65
66

```

University of Florida, EGN 1935—
© Drs. A. A. Arroyo & E. M. Schwartz

Ln 26: 66 Col 1 Sel 0 1.88 KB ANSI CR+LF INS C/C++ Source Code

12



EGN1935: ECE (Ad)Ventures Testing Motors 2

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Motor_Test_2\MotTest2.c - Notepad2
File Edit View Settings ?
1|*****
2|*
3|* Title:      EGN-1935 Motor Test 2
4|* Programmer: A. Antonio Arroyo
5|* Date:       February 16, 2011
6|* Version:    1.0
7|*
8|* Description:
9|* The robot will wait 5 seconds and then turns on both motors,
10|* the right motor forward, the left motor in reverse.
11|* The speed is obtained from a pot input connected to ADC channel A3
12|*****
13|
14|/***** Includes *****/
15|#include "AAA.h"
16|
17|/***** Prototypes *****/
18|void sensor_module();
19|
20|/***** Globals *****/
21|int pot = 0; // value for the potentiometer on ADC A3
22|
23|int main(void) {
24|    long j=0;
25|    int speedr, speedl, value;
26|
27|    init_system(); // initialize Ports A, C, E
28|    init_motor(); // initialize FPFC PWM on Timer 1
29|    init_lcd(); // set lcd in 4 bit mode, 2-line mode, with blinking cursor
30|    init_analog(); // initialize ADC System Analog Channels A0-A3
31|
32|    //
33|    1234567890ABCDEF
34|    lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
35|}
Ln1:88 Col1 Sel0 2.43 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 4
© Drs. A. A. Arroyo & E. M. Schwartz

13



EGN1935: ECE (Ad)Ventures Testing Motors 2

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Motor_Test_2\MotTest2.c - Notepad2
File Edit View Settings ?
34|    lcd_row(1);
35|    // 1234567890ABCDEF
36|    lcd_string("Motor Test 2"); // Heading line 2 for LCD
37|
38|    // Delay a few seconds ~5s
39|    delay(5000);
40|
41|
42|    lcd_clear();
43|    // 1234567890ABCDEF
44|    lcd_string("Varying Speed");
45|
46|    for(j=0; j<100; j++) { // finite loop
47|        sensor_module();
48|        value = pot;
49|        value = MIN(value,1000) / 10;
50|        speedl = value;
51|        speedr = -value;
52|        motor(LEFT_MOTOR, speedl);
53|        motor(RIGHT_MOTOR, speedr);
54|
55|        // Display values on the LCD
56|        lcd_clear();
57|        lcd_string("ADC3:");
58|        lcd_int(pot);
59|        lcd_row(1);
60|        lcd_string(" L:");
61|        lcd_int(speedl);
62|        lcd_string(" R:");
63|        lcd_int(speedr);
64|        lcd_string(" ");
65|        lcd_int(j);
66|        lcd_string(" ");
67|    }
Ln13:88 Col1 Sel0 2.43 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 4
© Drs. A. A. Arroyo & E. M. Schwartz

14



EGN1935: ECE (Ad)Ventures Testing Motors 2

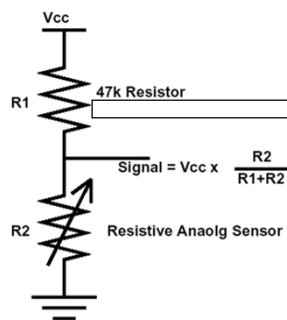
```

67
68 // complement pin PA0-PA1 and write back
69 if (PINA & 0x01) {PORTA=PINA&0xFE; PORTA=PINA|0x02;}
70 else {PORTA=PINA&0xFD; PORTA=PINA|0x01;}
71
72 // Delay about ~500ms
73 delay(500);
74
75 } // end for loop
76
77 motor(LEFT_MOTOR, 0); // Stop the Left Motor
78 motor(RIGHT_MOTOR,0); // Stop the Right Motor
79 return 0;
80 } // main
81
82 void sensor_module() { // Collect input sensor data
83 // l1r = analog(0);
84 // c1r = analog(1);
85 // r1r = analog(2);
86 // pot = analog(3);
87 }
88
Ln 55 : 88 Col 34 Sel 0      2.43 KB      ANSI      CR+LF  INS  C/C++ Source Code

```



EGN1935: ECE (Ad)Ventures CdS Cells



When light shines on a CdS cell, R_2 below decreases to a minimum, say 1K, causing the AD value to decrease to $1/48 V_{cc}$ or ≈ 21 . When a small amount of light shines on the CdS cell, the value of R_2 increases to a maximum value, say, 10K so the AD will read $10/57 V_{cc}$ or ≈ 180 . We will pick R_1 so that these values are from 500 - 900



EGN1935: ECE (Ad)Ventures Testing IRs

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\IR_CDS_Test\Irtst.c - Notepad2
File Edit View Settings ?
1 | *****
2 | *
3 | * Title:      EGN-1935 IRs & Cds Test
4 | * Programmer: A. Antonio Arroyo
5 | * Date:       February 16, 2011
6 | * Version:    1.0
7 | *
8 | * Description:
9 | * The robot will wait 5 seconds and then displays the IRs & Cds
10 | * input sensor values collected from ADC A0-A3
11 | *****
12 |
13 | ***** Includes *****
14 | #include "AAA.h"
15 |
16 | ***** Prototypes *****
17 | void sensor_module();
18 |
19 | ***** Globals *****
20 | int lir, cir, rir, cds;
21 |
22 | int main(void) {
23 |     long j=0;
24 |
25 |     init_system(); // initialize Ports A, C, E
26 |     init_motor();  // initialize FPFC PWM on Timer 1
27 |     init_lcd();    // set lcd in 4 bit mode, 2-line mode, with blinking cursor
28 |     init_analog(); // initialize ADC System Analog Channels A0-A3
29 |
30 |     // 1234567890ABCDEF
31 |     lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
32 |     lcd_row(1);
33 |     // 1234567890ABCDEF
34 |     lcd_string("IR & Cds Test"); // Heading line 2 for LCD
35 |
36 |     // Delay a few seconds ~5s
37 |     delay(5000);
38 | }
39 |
40 |
41 |
42 |
43 |
44 |
45 |
46 |
47 |
48 |
49 |
50 |
51 |
52 |
53 |
54 |
55 |
56 |
57 |
58 |
59 |
60 |
61 |
62 |
63 |
64 |
65 |
66 |
67 |
68 |
69 |
70 |
71 |
72 |
73 |
74 |
Ln1:74 Col1 Sel0 2.02 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 0
© Drs. A. A. Arroyo & E. M. Schwartz

17



EGN1935: ECE (Ad)Ventures Testing IRs

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\IR_CDS_Test\Irtst.c - Notepad2
File Edit View Settings ?
39 |
40 | for (j = 0; j < 100; j++) { // finite loop
41 |     sensor_module();
42 |
43 |     // Display sensor values from ADC A0-A3
44 |     lcd_clear();
45 |     // 1234567890ABCDEF
46 |     lcd_string("L C R Cd ");
47 |     lcd_int(j);
48 |     lcd_row(1);
49 |     lcd_string(" ");
50 |     lcd_int(lir);
51 |     lcd_string(" ");
52 |     lcd_int(cir);
53 |     lcd_string(" ");
54 |     lcd_int(rir);
55 |     lcd_string(" ");
56 |     lcd_int(cds);
57 |     lcd_string(" ");
58 |
59 |     // complement pin PA0-PA1 and write back
60 |     if (PINA & 0x01) (PORTA=PINA&0xFE; PORTA=PINA|0x02;)
61 |     else (PORTA=PINA&0xFD; PORTA=PINA|0x01;)
62 |
63 |     // Delay about ~100ms
64 |     delay(100);
65 |
66 | } // end for loop
67 | return 0;
68 | // main
69 |
70 | void sensor_module() { // Collect input sensor data
71 |     lir = analog(0);
72 |     cir = analog(1);
73 |     rir = analog(2);
74 |     cds = analog(3);
75 | }
76 |
77 |
78 |
79 |
80 |
81 |
82 |
83 |
84 |
85 |
86 |
87 |
88 |
89 |
90 |
91 |
92 |
93 |
94 |
95 |
96 |
97 |
98 |
99 |
Ln23:74 Col14 Sel0 2.02 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 0
© Drs. A. A. Arroyo & E. M. Schwartz

18



EGN1935: ECE (Ad)Ventures

Simple Behaviors

- Obstacle Avoidance
 - > Read the IRs
 - > If the left IR is “too close” reverse the right motor
 - > If the right IR is “too close” reverse the left motor
 - > If the center IR is “too close” back up 500ms & random turn
 - > To get a uniform random number take the low order bit of the 16-bit Timer1 counting register (TCNT1). If it is 0 turn one way, else turn the other way. Take the remainder of $TCNT1 \% 1023$ and use it as the amount of ms to turn (a minimum of 500ms)

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

19

EGN1935: ECE (Ad)Ventures
Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoid\Obsavoid.c - Notepad2
File Edit View Settings ?
1 /*****
2 *
3 * Title:      EGN-1935 IR-based Obstacle Avoidance
4 * Programmer: A. Antonio Arroyo
5 * Date:      February 16, 2011
6 * Version:   1.0
7 *
8 * Description:
9 * The robot will wait 5 seconds and then move forward & read IRs
10 * It will then check the IRs and decide whether to turn or not
11 *****/
12
13 /***** Includes *****/
14 #include "AaA.h"
15
16 /***** Prototypes *****/
17 void turn();
18 void sensor_module();
19
20 /***** Globals *****/
21 int lir, cir, rir, cds;
22
23 int main(void) {
24     long j=0;
25     int speedr, speedl;
26
27     init_system(); // initialize Ports A, C, E
28     init_led();   // set led in 4 bit mode, 2-line mode, with cursor on and set to blink
29     init_motor(); // Initialize Timer 1 PFC PWM System
30     init_analog(); // Initialize ADC : Ports A0-A3
31     lcd_clear();  // Clear LCD Panel
32
33     // 1234567890ABCDEF
34     lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
35     lcd_row(1);
36     // 1234567890ABCDEF

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

20



EGN1935: ECE (Ad)Ventures Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoid\Obsavoid.c - Notepad2
File Edit View Settings ?
37 lcd_string("Obstacle Avoid"); // Heading line 2 for LCD
38
39 // Delay a few seconds ~5s
40 delay(5000);
41
42 for (j = 0; j < 200; j++) { // finite loop
43     sensor_module();
44
45     // Display Sensor Values on Line 0
46     lcd_clear();
47     lcd_int(lir);
48     lcd_string(" ");
49     lcd_int(cir);
50     lcd_string(" ");
51     lcd_int(rir);
52     lcd_string(" ");
53     lcd_int(cds);
54     lcd_string(" ");
55
56     if (lir > THRESHOLD) // Veer Right
57         speedr = -MAX_SPEED;
58     else
59         speedr = MAX_SPEED; // Go straight
60
61     if (rir > THRESHOLD) // Veer Left
62         speedl = -MAX_SPEED;
63     else
64         speedl = MAX_SPEED; // Go straight
65
66     motor(LEFT_MOTOR, speedl);
67     motor(RIGHT_MOTOR, speedr);
68
69 // Display the (Left,Right) Motor Speed on Line 1
70 lcd_row(1);
71 lcd_int(speedl);
72 lcd_string(" ");

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

21



EGN1935: ECE (Ad)Ventures Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoid\Obsavoid.c - Notepad2
File Edit View Settings ?
73 lcd_int(speedr);
74 lcd_string(" ");
75
76 if (cir > THRESHOLD) { // Frontal Collision Avoidance
77     // Backup Robot for 0.5s
78     motor(LEFT_MOTOR, -MAX_SPEED);
79     motor(RIGHT_MOTOR, -MAX_SPEED);
80     delay(500);
81     turn(); // Make a random turn based on TCNT1
82 } // end Frontal Collision Avoidance
83
84 // complement pin PA0-PA1 and write back
85 if (PINA & 0x01) (PORTA=PINA&0xFE; PORTA=PINA|0x02;)
86 else (PORTA=PINA&0xFD; PORTA=PINA|0x01;)
87
88 // Delay about ~35ms
89 delay(35);
90
91 // end for loop
92 motor(RIGHT_MOTOR, 0); // Stop the Right Motor
93 motor(LEFT_MOTOR, 0); // Stop the Left Motor
94 return 0;
95 // main
96
97 void turn(void) {
98     // *****
99     * Function: Will turn in a random direction for a "random" amount of *
100     * time, dictated by the fast changing lower bits in ms *
101     * Returns: None *
102     * *
103     * Inputs *
104     * Parameters: None *
105     * Globals: None *
106     * Registers: TCNT *
107     * Outputs *
108     * Parameters: None *

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

22



EGN1935: ECE (Ad)Ventures Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoid\Obsavoid.c - Notepad2
File Edit View Settings ?
109 * Globals: None
110 * Registers: None
111 * Functions called: motor()
112 * Notes:
113 *****
114 int j;
115 unsigned rand;
116
117 rand = TCNT1;
118
119 if (rand & 0x0001) {
120 /*turn left*/
121 motor(RIGHT_MOTOR, MAX_SPEED);
122 motor(LEFT_MOTOR, -MAX_SPEED);
123 }
124 else {
125 /*turn right*/
126 motor(RIGHT_MOTOR, -MAX_SPEED);
127 motor(LEFT_MOTOR, MAX_SPEED);
128 }
129
130 j=(rand % 1024);
131 if (j>500)
132 // Delay for j>500ms
133 delay(j);
134
135 else
136 // Delay ~500 ms
137 delay(500);
138
139) /****End Function turn ****/
140
141 void sensor_module() { // Collect input sensor data
142 ltr = analog(0);
143 ctr = analog(1);
144 rtr = analog(2);
145 cds = analog(3);
146)

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

23



EGN1935: ECE (Ad)Ventures Behaviors

- Multiple Behaviors
 - > When multiple behaviors co-exist and the potential is there for contradictory values being sent to an actuator (motor), some sort of arbitration must take place to insure that the software does not generate “spikes.”
 - > A simple **arbitrator** uses a sensor, such as a switch, or resistive sensor to determine which behavior has control
 - > No behavior should ever control the motors directly
 - > Let us add CdS control to Obstacle Avoidance

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

24



EGN1935: ECE (Ad)Ventures CdS & Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoidance_CDS\Obscds.c - Notepad2
File Edit View Settings ?
1 //***** Includes *****/
2 *
3 * Title:      EGN-1935 IR-based Obstacle Avoidance w/ Cds Sensor
4 * Programmer: A. Antonio Arroyo
5 * Date:       February 16, 2011
6 * Version:    1.0
7 *
8 * Description:
9 * The robot will wait 5 seconds and then move forward & read IRs
10 * It will then check the IRs and decide whether to turn or not
11 * If the Cds sensor does not detect enough light, the robot will stop
12 *****/
13
14 /***** Includes *****/
15 #include "AAA.h"
16
17 /***** Prototypes *****/
18 void turn();
19 void sensor_module();
20
21 /***** Globals *****/
22 int lir, cir, rir, cds;
23
24 int main(void) {
25     long j=0;
26     int speedr, speedl;
27
28     init_system(); // initialize Ports A, C, E
29     init_lcd();    // set lcd in 4 bit mode, 2-line mode, with cursor on and set to blink
30     init_motor();  // Initialize Timer 1 PFC PWM System
31     init_analog(); // Initialize ADC : Ports AD-A3
32
33     //
34     lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
35     lcd_row(1);
36     //
37     lcd_string("Cds & Obs. Avoid"); // Heading line 2 for LCD
38 }

```

University of Florida, EGN 1935 – File 4
© Drs. A. A. Arroyo & E. M. Schwartz

25



EGN1935: ECE (Ad)Ventures CdS & Obstacle Avoidance

```

C:\Arroyo\EGN-1935\Spring 2011\EGN Robot\Obstacle_Avoidance_CDS\Obscds.c - Notepad2
File Edit View Settings ?
39 // Delay a few seconds ~5s
40 delay(5000);
41
42 for (j = 0; j < 200; j++) { // Finite Loop
43     sensor_module();
44
45     // Display Sensor Values on Line 0
46     lcd_clear();
47     lcd_int(lir);
48     lcd_string(" ");
49     lcd_int(cir);
50     lcd_string(" ");
51     lcd_int(rir);
52     lcd_string(" ");
53     lcd_int(cds);
54     lcd_string(" ");
55
56     if (cds > CDS_THRESHOLD) { // Stop the motors if too dark
57         speedl=0;
58         speedr=0;
59     } // end Stop the motors if too dark
60
61     else { // Normal Collision Avoidance
62         if (lir > THRESHOLD) // Veer Right
63             speedr = -MAX_SPEED;
64         else
65             speedr = MAX_SPEED; // Go straight
66
67         if (rir > THRESHOLD) // Veer Left
68             speedl = -MAX_SPEED;
69         else
70             speedl = MAX_SPEED; // Go straight
71     } // end Normal Collision Avoidance
72
73     motor(LEFT_MOTOR, speedl);
74     motor(RIGHT_MOTOR, speedr);
75
76 // Display the (Left,Right) Motor Speed on Line 1

```

University of Florida, EGN 1935 – File 4
© Drs. A. A. Arroyo & E. M. Schwartz

26



EGN1935: ECE (Ad)Ventures CdS & Obstacle Avoidance

```

C:\Arroyo\EGN 1935\Spring 2011\EGN Robot\Obstacle_Avoidance_CdS\Obscds.c - Notepad2
File Edit View Settings ?
77 lcd_row(1);
78 lcd_int(speed1);
79 lcd_string(" ");
80 lcd_int(speed2);
81 lcd_string(" ");
82
83 if (c1r > THRESHOLD) { // Frontal Collision Avoidance
84 // Backup Robot
85 motor(LEFT_MOTOR, -MAX_SPEED);
86 motor(RIGHT_MOTOR, -MAX_SPEED);
87 delay(500);
88 turn(); // Make a random turn based on TCNT1
89 } // end Frontal Collision Avoidance
90
91 // complement pin PA0-PA1 and write back
92 if (PINA & 0x01) (PORTA=PINA&0xFE; PORTA=PINA|0x02;)
93 else (PORTA=PINA&0xFD; PORTA=PINA|0x01;)
94
95 // Delay about ~35ms
96 delay(35);
97
98 } // end for loop
99 motor(RIGHT_MOTOR, 0); // Stop the Right Motor
100 motor(LEFT_MOTOR, 0); // Stop the Left Motor
101 return 0;
102 // main
103 void turn(void) {
104 //*****
105 * Function: Will turn in a random direction for a "random" amount of
106 * time, dictated by the fast changing lower bits in ms
107 * Returns: None
108 *
109 * Inputs
110 * Parameters: None
111 * Globals: None
112 * Registers: TCNT1
113 *
114 *
115
Ln 52:153 Col 25 Sel 0 4.75 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

27



EGN1935: ECE (Ad)Ventures CdS & Obstacle Avoidance

```

C:\Arroyo\EGN 1935\Spring 2011\EGN Robot\Obstacle_Avoidance_CdS\Obscds.c - Notepad2
File Edit View Settings ?
115 * Outputs
116 * Parameters: None
117 * Globals: None
118 * Registers: None
119 * Functions called: motor()
120 * Notes:
121 //*****
122 int j;
123 unsigned rand;
124 rand = TCNT1;
125
126 if (rand < 0x0001) {
127 /*turn left*/
128 motor(RIGHT_MOTOR, MAX_SPEED);
129 motor(LEFT_MOTOR, -MAX_SPEED);
130 }
131 else {
132 /*turn right*/
133 motor(RIGHT_MOTOR, -MAX_SPEED);
134 motor(LEFT_MOTOR, MAX_SPEED);
135 }
136
137 j=(rand % 512); // Take the remainder of TCNT1 / 512 (0 < rem < 511)
138 if (j>500)
139 // Delay j>500ms
140 delay(j);
141 else
142 // Delay ~500ms
143 delay(500);
144
145 //*****End Function turn *****
146
147 void sensor_module() { // Collect input sensor data
148 l1r = analog(0);
149 c1r = analog(1);
150 r1r = analog(2);
151 cds = analog(3);
152 }
Ln 91:153 Col 46 Sel 0 4.75 KB ANSI CR+LF INS C/C++ Source Code

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

28



EGN1935: ECE (Ad)Ventures Hacked Servo Test

```

1 /*
2  * Title:      EGN-1935 Robots Hacked Servo Test
3  * Programmer: A. Antonio Arroyo
4  * Date:      October 13, 2014
5  * Version:   1.0
6  * Description:
7  * Verifies the PWM system on the MIL Atmel Board on EGN-1935 Robot
8  * is working by generating PWM signal on Timer1 using PFC PWM Mode
9  * On the EGN-1935 robots the left servo is connected to OC1A/PB5/PWMA
10 * and the right servo is connected to OC1B/PB6/PWMB
11 * Also toggles PA0 & PA1 where LEDs w/ serial resistors are connected */
12
13 //***** Includes *****/
14 #include "AAA.h"
15 //***** Prototypes *****/
16 //***** Globals *****/
17
18 int main(void) {
19     long i = 0;
20     int value=1500;
21     init_system(); // initialize Ports A, C, E
22     init_lcd();    // set lcd in 4 bit mode, 2-line mode, cursor on and set to blink
23     init_motor();  // Initialize Timer 1 PFC PWM System
24     // 1234567890ABCDEF
25     lcd_string("EGN-1935 Robots "); // Heading line 1 for LCD
26     lcd_row(1);
27     // 1234567890ABCDEF
28     lcd_string("Hacked Servo Tst"); // Heading line 2 for LCD
29 }

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

29



EGN1935: ECE (Ad)Ventures Hacked Servo Test

```

28 // Delay a few seconds ~5s
29 delay(5000);
30 lcd_clear();
31 lcd_string("OCR1A & OCR1B");
32 for (value = 500; value < 2501; value=value+50){ // Finite Loop
33     OCR1A = value; // One motor is wired in forward
34     OCR1B = 3000 - value; // 2nd motor is wired in reverse
35     OCR1C = value; // 3rd motor is same as first
36 // Display the Hacked Servo Values on the LCD
37     lcd_row(1);
38     lcd_int(value);
39     lcd_string(" ");
40     lcd_int(3000 - value);
41     lcd_string(" ");
42 // Delay a few seconds ~2s to read LCD
43     delay(2000);
44     // complement pin PA0-PA1 and write back
45     if (PINA & 0x01) {PORTA=PINA&0xFE; PORTA=PINA|0x02;}
46     else {PORTA=PINA&0xFD; PORTA=PINA|0x01;}
47 } // end for loop
48 OCR1A=1500; // Stop Servo on PB5/OC1A
49 OCR1B=1500; // Stop Servo on PB6/OC1B
50 OCR1C=1500; // Stop Servo on PB7/OC1C
51 // Turn both LEDs on
52 PORTA=PINA|0x03;
53 return 0;
54 } // main

```

University of Florida, EGN 1935 – File 04
© Drs. A. A. Arroyo & E. M. Schwartz

30



EGN1935: ECE (Ad)Ventures

“Learning is an important part of autonomy. A system is autonomous to the extent that its behaviour is determined by its immediate inputs and past experience, rather by its designer’s. Agents are usually designed for a class of environments, where each member of the class is consistent with what the designer knows about what the real environment might hold in store for the agent. Truly autonomous systems should be able to operate successfully in any environment, given sufficient time to adapt. The system’s internal knowledge structures should therefore be constructible, in principle, from its experience of the world...”

The End!