

```

1 /*****
2 This program was produced by the
3 CodeWizardAVR V2.03.4 Standard
4 Automatic Program Generator
5 © Copyright 1998-2008 Pavel Haiduc, HP InfoTech s.r.l.
6 http://www.hpinfotech.com
7
8 Project :
9 Version :
10 Date : 2009.10.25.
11 Author :
12 Company :
13 Comments:
14
15
16 Chip type : ATmega8
17 Program type : Application
18 Clock frequency : 8.000000 MHz
19 Memory model : Small
20 External RAM size : 0
21 Data Stack size : 256
22 *****/
23
24 #include <mega8.h>
25 #include <delay.h>
26 #include <stdio.h>
27 #include <delay.h>
28
29
30
31 #define FIRST_ADC_INPUT 0
32 #define LAST_ADC_INPUT 2
33 unsigned char adc_data[LAST_ADC_INPUT-FIRST_ADC_INPUT+1];
34 #define ADC_VREF_TYPE 0x20
35
36
37
38 #define LED1_on S_bit(PORTD,0)
39 #define LED1_off C_bit(PORTD,0)
40 #define LED2_on S_bit(PORTD,1)
41 #define LED2_off C_bit(PORTD,1)
42 #define LED3_on S_bit(PORTD,2)
43 #define LED3_off C_bit(PORTD,2)
44 #define Brake_on S_bit(PORTC,5)
45 #define Brake_off C_bit(PORTC,5)
46 #define EN1_on S_bit(PORTB,0)
47 #define EN1_off C_bit(PORTB,0)
48 #define EN2_on S_bit(PORTB,1)
49 #define EN2_off C_bit(PORTB,1)
50 #define A1_on S_bit(PORTB,2)
51 #define A1_off C_bit(PORTB,2)
52 #define B1_on S_bit(PORTB,3)
53 #define B1_off C_bit(PORTB,3)
54 #define A2_on S_bit(PORTB,4)
55 #define A2_off C_bit(PORTB,4)
56 #define B2_on S_bit(PORTB,5)
57 #define B2_off C_bit(PORTB,5)
58 #define Me1ns 170
59
60 #define F_bit(ADDRESS,BIT) (ADDRESS ^= (1<<BIT)) //Flip bit
61 #define T_bit(ADDRESS,BIT) (ADDRESS & (1<<BIT)) //Test bit
62 #define S_bit(ADDRESS,BIT) (ADDRESS |= (1<<BIT)) //set bit
63 #define C_bit(ADDRESS,BIT) (ADDRESS &= ~(1<<BIT)) //Clear bit
64 // Timer 0 overflow interrupt service routine
65 unsigned int Count,Speed_a,Speed_at,Speed_b,Speed_bt;
66 unsigned char Linija_L,Linija_K,Linija_C;
67
68
69 interrupt [TIM0_OVF] void timer0_ovf_isr(void)
70 {
71     if (Count++ == 100){
72         Count=0;
73         A1_on;
74         A2_on;
75     }
76     if (Count>Speed_at){
77         A1_off;
78     }
79     if (Count>Speed_bt){
80         A2_off;
81     }
82     TCNT0=0x0F;
83 }
84
85

```

```

86
87 // ADC interrupt service routine
88 // with auto input scanning
89 interrupt [ADC_INT] void adc_isr(void)
90 {
91     static unsigned char input_index=0;
92     // Read the 8 most significant bits
93     // of the AD conversion result
94     adc_data[input_index]=ADCH;
95     Linija_K=adc_data[0];
96     Linija_C=adc_data[1];
97     Linija_L=adc_data[2];
98     // Select next ADC input
99     if (++input_index > (LAST_ADC_INPUT-FIRST_ADC_INPUT))
100         input_index=0;
101     ADMUX=(FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff))+input_index;
102     // Delay needed for the stabilization of the ADC input voltage
103     delay_us(10);
104     // Start the AD conversion
105     ADCSRA|=0x40;
106 }
107
108 // Declare your global variables here
109 void Init(void){
110
111     // Input/Output Ports initialization
112     // Port B initialization
113     // Func7=In Func6=In Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
114     // State7=T State6=T State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
115     PORTB=0x00;
116     DDRB=0x3F;
117
118     // Port D initialization
119     // Func7=In Func6=In Func5=In Func4=In Func3=In Func2=Out Func1=Out Func0=Out
120     // State7=T State6=T State5=T State4=T State3=T State2=0 State1=0 State0=0
121     PORTD=0x00;
122     DDRD=0x07;
123
124     // Timer/Counter 0 initialization
125     // Clock source: System Clock
126     // Clock value: 1000.000 kHz
127     TCCR0=0x01;
128     TCNT0=0xFE;
129     // Timer(s)/Counter(s) Interrupt(s) initialization
130     TIMSK=0x01;
131
132     // Analog Comparator initialization
133     // Analog Comparator: Off
134     // Analog Comparator Input Capture by Timer/Counter 1: Off
135     ACSR=0x80;
136     SFIOR=0x00;
137
138
139     // ADC initialization
140     // ADC Clock frequency: 250.000 kHz
141     // ADC Voltage Reference: AREF pin
142     // Only the 8 most significant bits of
143     // the AD conversion result are used
144     ADMUX=FIRST_ADC_INPUT | (ADC_VREF_TYPE & 0xff);
145     ADCSRA=0xCD;
146
147     // Global enable interrupts
148     #asm("sei")
149 }
150
151 void main(void)
152 {
153     //Uzstādām parametrus dzelziem
154     Init();
155     //Sakuma pakāpeniska diozu iespīdināšana
156     LED1_on;
157     delay_ms(1000);
158     LED2_on;
159     delay_ms(1000);
160     LED3_on;
161     delay_ms(1000);
162     B1_off;
163     B2_off;
164     EN1_on;
165     EN2_on;
166     Speed_a=0;
167     Speed_b=0;
168     while (1)
169     {
170 //-----

```

```

171 //Parbaudam vai Kreisaja senora redzama linija
172 //ja redzama iesledzam diodi
173 if (Linija_K<Melns){
174     LED1_on;
175 }
176 else LED1_off;
177 //Parbaudam vai Videja senora redzama linija
178 //ja redzama iesledzam diodi
179 if (Linija_C<Melns){
180     LED2_on;
181 }
182 else LED2_off;
183 //Parbaudam vai Labaja senora redzama linija
184 //ja redzama iesledzam diodi
185 if (Linija_L<Melns){
186     LED3_on;
187 }
188 else LED3_off;
189 //-----
190 //Palelinata motora atruma uznemsana un bremzesana
191 //lai izvairitos no stravas pikiem
192     if (Speed_a>Speed_at){
193         Speed_at++;
194     }
195     if (Speed_a<Speed_at){
196         Speed_at--;
197     }
198     if (Speed_b>Speed_bt){
199         Speed_bt++;
200     }
201     if (Speed_b<Speed_bt){
202         Speed_bt--;
203     }
204     if ((Speed_at<10) && (Speed_bt<10)){
205         delay_ms(20);
206         F_bit(PORTC,5);
207     }
208 //-----
209 //-----PARBAUDAM LINIJAS IESPEJAMOS STAVOKLUS UN VEICAM KOREKCIJU
210
211 //Taisni uz prieksu
212 if ((Linija_K<Melns)&&(Linija_C>Melns)&&(Linija_L<Melns)){
213     Speed_a=20;
214     Speed_b=20;
215 };
216
217 //Mazliet pa kreisi
218 if ((Linija_K>Melns)&&(Linija_C>Melns)&&(Linija_L<Melns)){
219     Speed_a=10;
220     Speed_b=20;
221 };
222
223 //Kreisa puse
224 if ((Linija_K>Melns)&&(Linija_C<Melns)&&(Linija_L<Melns)){
225     Speed_a=0;
226     Speed_b=20;
227 };
228
229 //Mazliet pa labi
230 if ((Linija_K<Melns)&&(Linija_C>Melns)&&(Linija_L>Melns)){
231     Speed_a=20;
232     Speed_b=10;
233 };
234
235 //Labaja puse
236 if ((Linija_K<Melns)&&(Linija_C<Melns)&&(Linija_L>Melns)){
237     Speed_a=20;
238     Speed_b=0;
239 };
240
241 //Krustojums
242 if ((Linija_K>Melns)&&(Linija_C>Melns)&&(Linija_L>Melns)){
243     Speed_a=10;
244     Speed_b=10;
245 };
246 };
247 }
248

```