Hardware Driver for FunnyRobot V1

Source Code

Gong, Zhangxiaowen (Andy)

November 2007

(Last modified on October 17th 2009)

Table of Contents

Driver.h

```
1   /*-----------------------------------------//
2   File Name: Driver.h
3   Date Established: 2007/10/06
4   Last Modified: 2008/09/19
5   Programmer: Gong Zhangxiaowen(Andygongyb)
6   E-Mail: gongzhangxiaowen@hotmail.com
7   //-----------------------------------------*/
8
9   #ifndef _Driver_Demo_
10  #define _Driver_Demo_
11
12  #warning FunnyRobot V1.0 Drive , Version Demo , Written by Andygongyb
13  #define LCD1602_BIT 4
14  #define LCD1602_RS (PA0)
15  #define LCD1602_E (PA1)
16  #define LCD1602_D7 (PA5)
17  #define LCD1602_D6 (PA4)
18  #define LCD1602_D5 (PA3)
19  #define LCD1602_D4 (PA2)
20  #define AVRM_PORT_LCD1602_RS (PORTA)
21  #define AVRM_PORT_LCD1602_E (PORTA)
22  #define AVRM_PORT_LCD1602_D7 (PORTA)
23  #define AVRM_PORT_LCD1602_D6 (PORTA)
24  #define AVRM_PORT_LCD1602_D5 (PORTA)
25  #define AVRM_PORT_LCD1602_D4 (PORTA)
26  #define AVRM_DDR_LCD1602_RS (DDRA)
27  #define AVRM_DDR_LCD1602_E (DDRA)
28  #define AVRM_DDR_LCD1602_D7 (DDRA)
29  #define AVRM_DDR_LCD1602_D6 (DDRA)
30  #define AVRM_DDR_LCD1602_D5 (DDRA)
31  #define AVRM_DDR_LCD1602_D4 (DDRA)
32  #ifndef FREQ
33  #define FREQ 8
34  #warning Using default FREQ value ( 8 MHz )
35  #endif
36  #define IN1 PB0
37  #define IN2 PB1
38  #define IN3 PB2
39  #define IN4 PB3
40  #define IN5 PG2
41  #define IN6 PG1
42  #define IN7 PG0
43  #define IN8 PG4
44  #if FREQ==16
45  #define ADCCLK ADCSRA=_BV(ADPS2)|_BV(ADPS1)
46  #define BAUD UBRR1L=103
```

```
47  #define TWCLK TWBR=72
48  #endif
49  #if FREQ==8
50  #define ADCCLK ADCSRA=_BV(ADPS2)|_BV(ADPS0)
51  #define BAUD UBRR1L=51
52  #define TWCLK TWBR=32
53  #endif
54  #if ((FREQ!=8)&&(FREQ!=16))
55  #error "FREQ must be 8 or 16!"
56  #endif
57  #ifndef _I2C_PORT_
58  #define _I2C_PORT_ (PORTA)
59  #warning Using default I2C command port ( PORTA )
60  #endif
61  #ifndef _I2C_CHK_
62  #define _I2C_CHK_ (PA6)
63  #warning Using default I2C check I/O ( PA6 )
64  #endif
65  #ifndef _I2C_DIR_
66  #define _I2C_DIR_ (PA7)
67  #warning Using default I2C direct I/O ( PA7 )
68  #endif
69  #warning USART BAUDRATE Setting: 9600 8 n 1
70
71  void IO_Init(void)__attribute__((naked))__attribute__((section(".init7")));
72  /* This function is actually naked code run before the main() function.
73   * It initializes hardware modules.
74   */
75  void I2C_Start(void);//This function starts a data transmission on the I2C bus.
76  void I2C_Stop(void);//This function terminates the I2C transmission.
77  void I2C_Write_Byte(unsigned char data);//This function sends one byte to the I2C bus.
78  void Compass_Init(void);//This function initializes the digital compass.
79  void Compass_Chk(void);//This function tends to reduce the noise disturbing the compass.
80  void Compass_Dir(void);//This function sets a direction to be the zero degree.
81  unsigned int Compass(void);//This function reads data from the compass.
82  void Delay_Ms(unsigned int Ms);//This function provides delays in millisecond level.
83  unsigned int Get_ADC(unsigned char CH);//This function reads data from ADC.
84  void Get_All_ADC(unsigned int ADCValue[8]);
85  //This function receives data from all ADC channels and stores it in a buffer.
86  void Go_1(signed int L,signed int R);//This function controls motor 1 and motor 2.
87  void Go_2(signed int L, signed int R);//This function controls motor 3 and motor 4.
88  void Stop_1(unsigned char Mode);//This function stops motor 1 and motor 2.
89  void Stop_2(unsigned char Mode);//This function stops motor 3 and motor 4.
90  void Stop_Motor(unsigned char CH,unsigned char Mode);//This function stops a single motor.
91  static int USART_Get_Char(FILE *stream);//This function sends one byte to the USART.
92  static int USART_Put_Char(char c, FILE *stream);
93  //This function gets one byte from the USART.
```

```
94
95  #endif
96
```

Driver.c

```
1   /*-----------------------------------------//
2   File Name: Driver.c
3   Date Established: 2007/10/06
4   Last Modified: 2008/09/19
5   Programmer: Gong Zhangxiaowen(Andygongyb)
6   E-Mail: gongzhangxiaowen@hotmail.com
7   //-----------------------------------------*/
8
9   #include <avr/io.h>
10  #include <avr/interrupt.h>
11  #include <util/delay.h>
12  #include <avr/pgmspace.h>
13  #include <avr/eeprom.h>
14  #include <avr/sfr_defs.h>
15  #include <stdio.h>
16  #include <stdlib.h>
17  #include <inttypes.h>
18
19  #include "Drive.h"
20  #include "Lib/LCD1602/LCD1602.c"
21
22  static unsigned char X=0,Y=0;
23  static FILE
24  MyStdStream=FDEV_SETUP_STREAM(USART_Put_Char,USART_Get_Char,_FDEV_SETUP_RW);
25
26  void IO_Init(void){
27      cli();
28      DDRF=0x00;
29      PORTF=0x00;
30      ADCCLK;
31      ADCSRA=_BV(ADEN);
32      TWCLK;
33      TWSR&=~_BV(TWPS1);
34      TWSR&=~_BV(TWPS0);
35      TCCR1A=_BV(COM1A1)|_BV(COM1B1)|_BV(WGM10);
36      TCCR3A=_BV(COM3A1)|_BV(COM3B1)|_BV(WGM30);
37      TCCR1B=_BV(CS12)|_BV(WGM12);
38      TCCR3B=_BV(CS32)|_BV(WGM32);
39      DDRB|=_BV(PB0)|_BV(PB1)|_BV(PB2)|_BV(PB3)|_BV(PB5)|_BV(PB6);
40      DDRG|=_BV(PG0)|_BV(PG1)|_BV(PG2)|(PG4);
41      DDRE|=_BV(PE3)|_BV(PE4);
42      BAUD;
```

```
43      UCSR1C=_BV(UCSZ11)|_BV(UCSZ10);
44      UCSR1B|=_BV(TXEN)|_BV(RXEN);
45      UCSR1B&=~_BV(UCSZ12);
46      Compass_Init();
47      LCD_Init();
48      stdout=&MyStdStream;
49      stdin=&MyStdStream;
50      DDRE|=_BV(PE7);
51      PORTE|=_BV(PE7);
52      Delay_Ms(200);
53      DDRE&=~_BV(PE7);
54      PORTE|=_BV(PE7);
55      Delay_Ms(200);
56      while(!(PINE&_BV(PE7))){
57              LCD_Locate(0,0);
58              LCD_Printf("%d ",Get_ADC(0));
59              LCD_Locate(0,4);
60              LCD_Printf("%d ",Get_ADC(1));
61              LCD_Locate(0,8);
62              LCD_Printf("%d ",Get_ADC(2));
63              LCD_Locate(0,12);
64              LCD_Printf("%d ",Get_ADC(3));
65              LCD_Locate(1,0);
66              LCD_Printf("%d ",Get_ADC(4));
67              LCD_Locate(1,4);
68              LCD_Printf("%d ",Get_ADC(5));
69              LCD_Locate(1,8);
70              LCD_Printf("%d ",Get_ADC(6));
71              LCD_Locate(1,12);
72              LCD_Printf("%d ",Get_ADC(7));
73              Delay_Ms(30);
74      };
75      LCD_Cls();
76      LCD_Printf("Funny Robot Demo\n By  Andygongyb",NULL);
77      sei();
78  }
79
80  unsigned int Get_ADC(unsigned char CH){
81      ADMUX=CH;
82      ADCSRA|=_BV(ADSC);
83      loop_until_bit_is_set(ADCSRA,ADIF);
84      ADCSRA|=_BV(ADIF);
85      return (ADC);
86  }
87
88  void I2C_Start(void){
89      TWCR=_BV(TWINT)|_BV(TWSTA)|_BV(TWEN)|_BV(TWEA);
```

```
90      while (!(TWCR&_BV(TWINT)));
91      TWCR=_BV(TWEN)|_BV(TWEA);
92  }
93
94  void I2C_Stop(void){
95      TWCR=_BV(TWINT)|_BV(TWSTO)|_BV(TWEN);
96      while (TWCR&_BV(TWSTO));
97  }
98
99  void I2C_Write_Byte(unsigned char data){
100     TWDR=data;
101     TWCR=_BV(TWINT)|_BV(TWEN)|_BV(TWEA);
102     while (!(TWCR&_BV(TWINT)));
103 }
104
105 void Compass_Init(void){
106     TWAR=_BV(TWGCE);
107     TWCR=_BV(TWEN)|_BV(TWINT);
108     I2C_Start();
109     I2C_Write_Byte(0x42);
110     I2C_Write_Byte(0x64);
111     I2C_Write_Byte(0x00);
112     I2C_Write_Byte(0x00);
113     I2C_Start();
114     I2C_Write_Byte(0x42);
115     I2C_Write_Byte(0x60);
116     I2C_Write_Byte(0x00);
117     I2C_Write_Byte(0x00);
118 }
119
120 void Compass_Chk(void){
121     while(!(_I2C_PORT_&_BV(_I2C_CHK_)));
122     Delay_Ms(100);
123     I2C_Start();
124     I2C_Write_Byte(0x42);
125     I2C_Write_Byte(0x70);
126     while((_I2C_PORT_&_BV(_I2C_CHK_)));
127     Delay_Ms(100);
128     I2C_Start();
129     I2C_Write_Byte(0x42);
130     I2C_Write_Byte(0x72);
131 }
132
133 void Compass_Dir(void){
134     while(!(_I2C_PORT_&_BV(_I2C_DIR_)));
135     Delay_Ms(100);
136     unsigned char data1=0,data2=0;
```

```
137     I2C_Start();
138     I2C_Write_Byte(0x42);
139     I2C_Write_Byte(0x74);
140     Delay_Ms(250);
141     I2C_Start();
142     I2C_Write_Byte(0x42);
143     I2C_Write_Byte(0x77);
144     Delay_Ms(250);
145     I2C_Start();
146     I2C_Write_Byte(0x43);
147     TWCR =_BV(TWINT)|_BV(TWEN)|_BV(TWEA);
148     while (!(TWCR&_BV(TWINT)));
149     data1=TWDR;
150     TWCR =_BV(TWINT)|_BV(TWEN);
151     while (!(TWCR&_BV(TWINT)));
152     data2=TWDR;
153     Delay_Ms(250);
154     eeprom_write_byte((unsigned char*)0x3fe,data1&1);
155     Delay_Ms(250);
156     eeprom_write_byte((unsigned char*)0x3ff,data2);
157 }
158
159 unsigned int Compass(void){
160     unsigned int data1=0,data2=0;
161     I2C_Start();
162     I2C_Write_Byte(0x42);
163     I2C_Write_Byte(0x74);
164     I2C_Start();
165     I2C_Write_Byte(0x42);
166     I2C_Write_Byte(0x77);
167     I2C_Start();
168     I2C_Write_Byte(0x43);
169     TWCR =_BV(TWINT)|_BV(TWEN)|_BV(TWEA);
170     while (!(TWCR&_BV(TWINT)));
171     data1=TWDR;
172     TWCR =_BV(TWINT)|_BV(TWEN);
173     while (!(TWCR&_BV(TWINT)));
174     data2=TWDR;
175     data2|=(data1&1)<<8;
176     data1=eeprom_read_byte((unsigned char*)0x3ff);
177     data1|=(eeprom_read_byte((unsigned char*)0x3fe)&1)<<8;
178     if(data1>360)data1=0;
179     if(data2>=data1)
180         data2-=data1;
181     else
182         data2=(360-(data1-data2));
183     return(data2);
```

```
184 }
185
186 void Go_1(signed int L, signed int R){
187     if (L>=0){
188         PORTB|=_BV(IN1);
189         PORTB&=~_BV(IN2);
190         if (L>1023)
191             L=1023;
192         OCR1A=L;
193     }
194     else{
195         PORTB|=_BV(IN2);
196         PORTB&=~_BV(IN1);
197         if (L<-1023)
198             L=-1023;
199         OCR1A=L*(-1);
200     }
201     if (R>=0){
202         PORTB|=_BV(IN3);
203         PORTB&=~_BV(IN4);
204         if (R>1023)
205             R=1023;
206         OCR1B=R;
207     }
208     else{
209         PORTB|=_BV(IN4);
210         PORTB&=~_BV(IN3);
211         if (R<-1023)
212             R=-1023;
213         OCR1B=R*(-1);
214     }
215 }
216
217 void Go_2(signed int L, signed int R){
218     if (L>=0){
219         PORTG|=_BV(IN5);
220         PORTG&=~_BV(IN6);
221         if (L>1023)
222             L=1023;
223         OCR3A=L;
224     }
225     else{
226         PORTG|=_BV(IN6);
227         PORTG&=~_BV(IN5);
228         if (L<-1023)
229             L=-1023;
230         OCR3A=L*(-1);
```

```
231     }
232     if (R>=0){
233         PORTG|=_BV(IN7);
234         PORTG&=~_BV(IN8);
235         if (R>1023)
236             R=1023;
237         OCR3B=R;
238     }
239     else{
240         PORTG|=_BV(IN8);
241        PORTG&=~_BV(IN7);
242         if (R<-1023)
243             R=-1023;
244         OCR3B=R*(-1);
245     }
246 }
247
248 void Stop_1(unsigned char Mode){
249     if (Mode){
250         PORTB|=_BV(IN1)|_BV(IN2)|_BV(IN3)|_BV(IN4);
251         OCR1A=1023;
252         OCR1B=1023;
253     }
254     else{
255         OCR1A=0;
256         OCR1B=0;
257     }
258 }
259
260 void Stop_2(unsigned char Mode){
261     if (Mode){
262         PORTG|=_BV(IN5)|_BV(IN6)|_BV(IN7)|_BV(IN8);
263         OCR3A=1023;
264         OCR3B=1023;
265     }
266     else{
267         OCR3A=0;
268         OCR3B=0;
269     }
270 }
271
272 void Stop_Motor(unsigned char CH,unsigned char Mode){
273     if (Mode)
274         switch(CH){
275             case 1:
276             PORTB|=_BV(IN1)|_BV(IN2);
277             OCR1A=1023;
```

```
278             break;
279             case 2:
280             PORTB|=_BV(IN3)|_BV(IN4);
281             OCR1B=1023;
282             break;
283             case 3:
284             PORTG|=_BV(IN5)|_BV(IN6);
285             OCR3A=1023;
286             break;
287             case 4:
288             PORTG|=_BV(IN7)|_BV(IN8);
289             OCR3B=1023;
290             break;
291             default: break;
292         }
293     else
294         switch(CH){
295             case 1:
296             OCR1A=0;
297             break;
298             case 2:
299             OCR1B=0;
300             break;
301             case 3:
302             OCR3A=0;
303             break;
304             case 4:
305             OCR3B=0;
306             break;
307             default: break;
308         }
309
310 }
311
312 void Delay_Ms(unsigned int Ms){
313     unsigned int i;
314     for(i=0;i<Ms;i++)
315         _delay_loop_2 (FREQ*250);
316 }
317
318 void Get_All_ADC(unsigned int ADCValue[8]){
319     for(unsigned char i=0;i<=7;i++)
320         ADCValue[i]=Get_ADC(i);
321 }
322
323 static int USART_Put_Char(char c, FILE *stream){
324     if (c =='\n')
```

```
325         USART_Put_Char('\r', stream);
326     loop_until_bit_is_set(UCSR1A, UDRE1);
327     UDR1 = c;
328     return 0;
329 }
330
331 static int USART_Get_Char(FILE *stream){
332     loop_until_bit_is_set(UCSR1A,RXC1);
333     return UDR1;
334 }
335
336 ISR(__vector_default){
337     cli();
338     LCD_Cls();
339     LCD_Printf("An interrupt has happened!");
340 }
341
```

Lib/LCD1602/LCD1602.h

```
1   /*-----------------------------------------//
2   File Name: LCD1602.h
3   Date Established: 2007/09/06
4   Last Modified: 2007/09/7
5   Programmer: Gong Zhangxiaowen(Andygongyb)
6   E-Mail: gongzhangxiaowen@hotmail.com
7   //-----------------------------------------*/
8
9   #ifndef _LCD1602_1602_
10  #define _LCD1602_1602_
11  #include <inttypes.h>
12  #include <avr/io.h>
13  #ifndef LCD1602_BIT
14  #warning "LCD1602_BIT not defined for LCD1602.h"
15  #define LCD1602_BIT 4
16  #endif
17  #if LCD1602_BIT != 4 && LCD1602_BIT != 8
18  #error "LCD1602_BIT must be 4 or 8"
19  #endif
20  #ifndef LCD1602_RS
21  #warning "LCD1602_RS not defined for LCD1602.h"
22  #define LCD1602_RS (PA0)
23  #endif
24  #ifndef LCD1602_E
25  #warning "LCD1602_E not defined for LCD1602.h"
26  #define LCD1602_E (PA1)
27  #endif
28  #ifndef LCD1602_D7
```

```
29  #warning "LCD1602_D7 not defined for LCD1602.h"
30  #define LCD1602_D7 (PA5)
31  #endif
32  #ifndef LCD1602_D6
33  #warning "LCD1602_D6 not defined for LCD1602.h"
34  #define LCD1602_D6 (PA4)
35  #endif
36  #ifndef LCD1602_D5
37  #warning "LCD1602_D5 not defined for LCD1602.h"
38  #define LCD1602_D5 (PA3)
39  #endif
40  #ifndef LCD1602_D4
41  #warning "LCD1602_D4 not defined for LCD1602.h"
42  #define LCD1602_D4 (PA2)
43  #endif
44  #if LCD1602_BIT == 8
45  #ifndef LCD1602_D3
46  #warning "LCD1602_D3 not defined for LCD1602.h"
47  #define LCD1602_D3 (PB0)
48  #endif
49  #ifndef LCD1602_D2
50  #warning "LCD1602_D2 not defined for LCD1602.h"
51  #define LCD1602_D2 (PB1)
52  #endif
53  #ifndef LCD1602_D1
54  #warning "LCD1602_D1 not defined for LCD1602.h"
55  #define LCD1602_D1 (PB2)
56  #endif
57  #ifndef LCD1602_D0
58  #warning "LCD1602_D0 not defined for LCD1602.h"
59  #define LCD1602_D0 (PB3)
60  #endif
61  #endif
62  #ifndef AVRM_PORT_LCD1602_RS
63  #warning "AVRM_PORT_LCD1602_RS not defined for LCD1602.h"
64  #define AVRM_PORT_LCD1602_RS (PORTA)
65  #endif
66  #ifndef AVRM_PORT_LCD1602_E
67  #warning "AVRM_PORT_LCD1602_E not defined for LCD1602.h"
68  #define AVRM_PORT_LCD1602_E (PORTA)
69  #endif
70  #ifndef AVRM_PORT_LCD1602_D7
71  #warning "AVRM_PORT_LCD1602_D7 not defined for LCD1602.h"
72  #define AVRM_PORT_LCD1602_D7 (PORTA)
73  #endif
74  #ifndef AVRM_PORT_LCD1602_D6
75  #warning "AVRM_PORT_LCD1602_D6 not defined for LCD1602.h"
```

```
76   #define AVRM_PORT_LCD1602_D6 (PORTA)
77   #endif
78   #ifndef AVRM_PORT_LCD1602_D5
79   #warning "AVRM_PORT_LCD1602_D5 not defined for LCD1602.h"
80   #define AVRM_PORT_LCD1602_D5 (PORTA)
81   #endif
82   #ifndef AVRM_PORT_LCD1602_D4
83   #warning "AVRM_PORT_LCD1602_D4 not defined for LCD1602.h"
84   #define AVRM_PORT_LCD1602_D4 (PORTA)
85   #endif
86   #if LCD1602_BIT == 8
87   #ifndef AVRM_PORT_LCD1602_D3
88   #warning "AVRM_PORT_LCD1602_D3 not defined for LCD1602.h"
89   #define AVRM_PORT_LCD1602_D3 (PORTB)
90   #endif
91   #ifndef AVRM_PORT_LCD1602_D2
92   #warning "AVRM_PORT_LCD1602_D2 not defined for LCD1602.h"
93   #define AVRM_PORT_LCD1602_D2 (PORTB)
94   #endif
95   #ifndef AVRM_PORT_LCD1602_D1
96   #warning "AVRM_PORT_LCD1602_D1 not defined for LCD1602.h"
97   #define AVRM_PORT_LCD1602_D1 (PORTB)
98   #endif
99   #ifndef AVRM_PORT_LCD1602_D0
100  #warning "AVRM_PORT_LCD1602_D0 not defined for LCD1602.h"
101  #define AVRM_PORT_LCD1602_D0 (PORTB)
102  #endif
103  #endif
104  #ifndef AVRM_DDR_LCD1602_RS
105  #warning "AVRM_DDR_LCD1602_RS not defined for LCD1602.h"
106  #define AVRM_DDR_LCD1602_RS (DDRA)
107  #endif
108  #ifndef AVRM_DDR_LCD1602_E
109  #warning "AVRM_DDR_LCD1602_E not defined for LCD1602.h"
110  #define AVRM_DDR_LCD1602_E (DDRA)
111  #endif
112  #ifndef AVRM_DDR_LCD1602_D7
113  #warning "AVRM_DDR_LCD1602_D7 not defined for LCD1602.h"
114  #define AVRM_DDR_LCD1602_D7 (DDRA)
115  #endif
116  #ifndef AVRM_DDR_LCD1602_D6
117  #warning "AVRM_DDR_LCD1602_D6 not defined for LCD1602.h"
118  #define AVRM_DDR_LCD1602_D6 (DDRA)
119  #endif
120  #ifndef AVRM_DDR_LCD1602_D5
121  #warning "AVRM_DDR_LCD1602_D5 not defined for LCD1602.h"
122  #define AVRM_DDR_LCD1602_D5 (DDRA)
```

```
123 #endif
124 #ifndef AVRM_DDR_LCD1602_D4
125 #warning "AVRM_DDR_LCD1602_D4 not defined for LCD1602.h"
126 #define AVRM_DDR_LCD1602_D4 (DDRA)
127 #endif
128 #if LCD1602_BIT == 8
129 #ifndef AVRM_DDR_LCD1602_D3
130 #warning "AVRM_DDR_LCD1602_D3 not defined for LCD1602.h"
131 #define AVRM_DDR_LCD1602_D3 (DDRB)
132 #endif
133 #ifndef AVRM_DDR_LCD1602_D2
134 #warning "AVRM_DDR_LCD1602_D2 not defined for LCD1602.h"
135 #define AVRM_DDR_LCD1602_D2 (DDRB)
136 #endif
137 #ifndef AVRM_DDR_LCD1602_D1
138 #warning "AVRM_DDR_LCD1602_D1 not defined for LCD1602.h"
139 #define AVRM_DDR_LCD1602_D1 (DDRB)
140 #endif
141 #ifndef AVRM_DDR_LCD1602_D0
142 #warning "AVRM_DDR_LCD1602_D0 not defined for LCD1602.h"
143 #define AVRM_DDR_LCD1602_D0 (DDRB)
144 #endif
145 #endif
146 #define LCD1602_INST 0
147 #define LCD1602_DATA 1
148 #ifndef FREQ
149 #define FREQ 8
150 #warning "Using default FREQ value ( 8 MHz )"
151 #endif
152
153 void LCD_Init(void);
154 void LCD_Write4(uint8_t cmd, uint8_t type);
155 void LCD_Write8(uint8_t cmd, uint8_t type);
156 void DelayCLK(void);
157 void LCD_Clear(void);
158 void LCD_Home(void);
159 void LCD_Close(void);
160 /* The functions above are fundamental LCD controls.*/
161
162 void LCD_Write_Place( uint8_t l, uint8_t c, uint8_t cmd);
163 //This function displays a character in a specific coordinate.
164 void LCD_Printf(char *fmt, ...);//This function offers formatted display of strings.
165 void LCD_Cls(void);
166 //This function clears the display and sets the cursor back to the origin.
167 void LCD_Locate( unsigned char X, unsigned char Y);
168 //This function sets the cursor to a specific position.
169
```

```
170 #endif
171
```

Lib/LCD1602/LCD1602.c

```
1   /*-----------------------------------------//
2   File Name: LCD1602.c
3   Date Established: 2007/09/06
4   Last Modified: 2007/09/10
5   Programmer: Gong Zhangxiaowen(Andygongyb)
6   E-Mail: gongzhangxiaowen@hotmail.com
7   //-----------------------------------------*/
8
9   #include "LCD1602.h"
10
11  void DelayCLK(void)
12  {
13      _delay_loop_2 (FREQ*190);
14  }
15
16  void avr_output(void){
17      AVRM_DDR_LCD1602_E |= _BV(LCD1602_E);
18      AVRM_DDR_LCD1602_RS |= _BV(LCD1602_RS);
19      AVRM_DDR_LCD1602_D7 |= _BV(LCD1602_D7);
20      AVRM_DDR_LCD1602_D6 |= _BV(LCD1602_D6);
21      AVRM_DDR_LCD1602_D5 |= _BV(LCD1602_D5);
22      AVRM_DDR_LCD1602_D4 |= _BV(LCD1602_D4);
23  #if LCD1602_BIT == 8
24      AVRM_DDR_LCD1602_D3 |= _BV(LCD1602_D3);
25      AVRM_DDR_LCD1602_D2 |= _BV(LCD1602_D2);
26      AVRM_DDR_LCD1602_D1 |= _BV(LCD1602_D1);
27      AVRM_DDR_LCD1602_D0 |= _BV(LCD1602_D0);
28  #endif
29  }
30
31  void avr_input(void){
32      AVRM_DDR_LCD1602_E &= ~_BV(LCD1602_E);
33      AVRM_DDR_LCD1602_RS &= ~_BV(LCD1602_RS);
34      AVRM_DDR_LCD1602_D7 &= ~_BV(LCD1602_D7);
35      AVRM_DDR_LCD1602_D6 &= ~_BV(LCD1602_D6);
36      AVRM_DDR_LCD1602_D5 &= ~_BV(LCD1602_D5);
37      AVRM_DDR_LCD1602_D4 &= ~_BV(LCD1602_D4);
38  #if LCD1602_BIT == 8
39      AVRM_DDR_LCD1602_D3 &= ~_BV(LCD1602_D3);
40      AVRM_DDR_LCD1602_D2 &= ~_BV(LCD1602_D2);
41      AVRM_DDR_LCD1602_D1 &= ~_BV(LCD1602_D1);
42      AVRM_DDR_LCD1602_D0 &= ~_BV(LCD1602_D0);
43  #endif
```

```
44  }
45
46  void lcd_data(uint8_t cmd, uint8_t offset){
47  #if LCD1602_BIT == 4
48      if( (cmd >> (0+offset) ) & 0x01 ) AVRM_PORT_LCD1602_D4 |=  _BV(LCD1602_D4);
49      else               AVRM_PORT_LCD1602_D4 &= ~_BV(LCD1602_D4);
50      if( (cmd >> (1+offset) ) & 0x01 ) AVRM_PORT_LCD1602_D5 |=  _BV(LCD1602_D5);
51      else               AVRM_PORT_LCD1602_D5 &= ~_BV(LCD1602_D5);
52      if( (cmd >> (2+offset) ) & 0x01 ) AVRM_PORT_LCD1602_D6 |=  _BV(LCD1602_D6);
53      else               AVRM_PORT_LCD1602_D6 &= ~_BV(LCD1602_D6);
54      if( (cmd >> (3+offset) ) & 0x01 ) AVRM_PORT_LCD1602_D7 |=  _BV(LCD1602_D7);
55      else               AVRM_PORT_LCD1602_D7 &= ~_BV(LCD1602_D7);
56  #else
57      if( (cmd >> 0 ) & 0x01 ) AVRM_PORT_LCD1602_D0 |=  _BV(LCD1602_D0);
58      else           AVRM_PORT_LCD1602_D0 &= ~_BV(LCD1602_D0);
59      if( (cmd >> 1 ) & 0x01 ) AVRM_PORT_LCD1602_D1 |=  _BV(LCD1602_D1);
60      else           AVRM_PORT_LCD1602_D1 &= ~_BV(LCD1602_D1);
61      if( (cmd >> 2 ) & 0x01 ) AVRM_PORT_LCD1602_D2 |=  _BV(LCD1602_D2);
62      else           AVRM_PORT_LCD1602_D2 &= ~_BV(LCD1602_D2);
63      if( (cmd >> 3 ) & 0x01 ) AVRM_PORT_LCD1602_D3 |=  _BV(LCD1602_D3);
64      else           AVRM_PORT_LCD1602_D3 &= ~_BV(LCD1602_D3);
65      if( (cmd >> 4 ) & 0x01 ) AVRM_PORT_LCD1602_D4 |=  _BV(LCD1602_D4);
66      else           AVRM_PORT_LCD1602_D4 &= ~_BV(LCD1602_D4);
67      if( (cmd >> 5 ) & 0x01 ) AVRM_PORT_LCD1602_D5 |=  _BV(LCD1602_D5);
68      else           AVRM_PORT_LCD1602_D5 &= ~_BV(LCD1602_D5);
69      if( (cmd >> 6 ) & 0x01 ) AVRM_PORT_LCD1602_D6 |=  _BV(LCD1602_D6);
70      else           AVRM_PORT_LCD1602_D6 &= ~_BV(LCD1602_D6);
71      if( (cmd >> 7 ) & 0x01 ) AVRM_PORT_LCD1602_D7 |=  _BV(LCD1602_D7);
72      else           AVRM_PORT_LCD1602_D7 &= ~_BV(LCD1602_D7);
73  #endif
74  }
75
76  void lcd_clk(void){
77      AVRM_PORT_LCD1602_E |= _BV(LCD1602_E);
78      DelayCLK();
79      AVRM_PORT_LCD1602_E &= ~_BV(LCD1602_E);
80      DelayCLK();
81  }
82
83  void lcd_rs_ct(uint8_t ct){
84      if(ct)
85          AVRM_PORT_LCD1602_RS |= _BV(LCD1602_RS);
86      else
87          AVRM_PORT_LCD1602_RS &= ~_BV(LCD1602_RS);
88  }
89
90  void LCD_Init(void){
```

```
91      avr_output();
92      #if LCD1602_BIT == 4
93      for (unsigned char i=0;i<25;i++)
94          DelayCLK();
95      LCD_Write4( 0x3, LCD1602_INST );
96      for (unsigned char i=0;i<8;i++)
97          DelayCLK();
98      LCD_Write4( 0x3, LCD1602_INST );
99          DelayCLK();
100     LCD_Write4( 0x3, LCD1602_INST );
101         DelayCLK();
102     LCD_Write4( 0x2, LCD1602_INST );
103     LCD_Write8( 0x2C, LCD1602_INST );
104     LCD_Write8( 0x08, LCD1602_INST );
105     LCD_Write8( 0x0C, LCD1602_INST );
106     LCD_Write8( 0x06, LCD1602_INST );
107     LCD_Write8( 0x02, LCD1602_INST );
108     LCD_Write8( 0x01, LCD1602_INST );
109     # else
110     for (unsigned char i=0;i<25;i++)
111         DelayCLK();
112     LCD_Write8( 0x38, LCD1602_INST );
113     for (unsigned char i=0;i<8;i++)
114         DelayCLK();
115     LCD_Write8( 0x38, LCD1602_INST );
116         DelayCLK();
117     LCD_Write8( 0x38, LCD1602_INST );
118         DelayCLK();
119     LCD_Write8( 0x3C, LCD1602_INST );
120     LCD_Write8( 0x08, LCD1602_INST );
121     LCD_Write8( 0x0C, LCD1602_INST );
122     LCD_Write8( 0x06, LCD1602_INST );
123     LCD_Write8( 0x02, LCD1602_INST );
124     LCD_Write8( 0x01, LCD1602_INST );
125     #endif
126 }
127
128 void LCD_Write4(uint8_t cmd, uint8_t type){
129     lcd_rs_ct(type);
130     lcd_data( cmd, 0 );
131     lcd_clk();
132 }
133
134 void LCD_Write8(uint8_t cmd, uint8_t type){
135     #if LCD1602_BIT == 4
136     lcd_rs_ct(type);
137     lcd_data( cmd, 4 );
```

```
138     lcd_clk();
139     lcd_data( cmd, 0 );
140     lcd_clk();
141     #else
142     lcd_rs_ct(type);
143     lcd_data( cmd, 0 );
144     lcd_clk();
145     #endif
146 }
147 
148 void LCD_Write_Place( uint8_t l, uint8_t c, uint8_t cmd){
149     uint8_t ram = (l*0x40)+c;
150     ram |= 0x80;
151     LCD_Write8( ram, LCD1602_INST);
152     LCD_Write8( cmd, LCD1602_DATA);
153 }
154 
155 void LCD_Clear(void){
156     LCD_Write8( 0x01, LCD1602_INST);
157 }
158 
159 void LCD_Home(void){
160     LCD_Write8( 0x02, LCD1602_INST);
161 }
162 
163 void LCD_Close(void){
164     LCD_Write8( 0x08, LCD1602_INST);
165 }
166 
167 void LCD_Printf(char *Str, ...){
168     va_list Ap;
169     char Strval[6];
170     char *p;
171     int Nval;
172     unsigned char i=0;
173     va_start(Ap,Str);
174     for(p=Str;*p;p++) {
175         if (*p!='%'){
176             if ((Y==16)&&(*p!='\n')){
177                 Y=0;
178                 X=(X==0)?1:0;
179                 LCD_Locate(X,Y);
180             }
181             if (*p=='\n'){
182                 Y=0;
183                 X=(X==0)?1:0;
184                 p++;
```

```
185                 LCD_Locate(X,Y);
186             }
187             LCD_Write8(*p,LCD1602_DATA);
188             Y++;
189             continue;
190         }
191         p++;
192         switch (*p){
193         case 'd':
194             Nval = va_arg(Ap, int);
195             itoa(Nval,Strval,10);
196             break;
197         case 'x':
198             Nval = va_arg(Ap, int);
199             itoa(Nval,Strval,16);
200             break;
201         default:continue;
202         }
203         for(i=0;Strval[i];i++){
204             if (Y==16){
205                 Y=0;
206                 X=(X==0)?1:0;
207                 LCD_Locate(X,Y);
208             }
209             LCD_Write8(Strval[i], LCD1602_DATA);
210             Y++;
211         }
212     }
213     va_end(Ap);
214 }
215
216 void LCD_Locate(unsigned char X, unsigned char Y){
217     unsigned char Ram=(X*0x40)+Y;
218     Ram|=0x80;
219     LCD_Write8(Ram,LCD1602_INST);
220 }
221
222 void LCD_Cls(void){
223     X=0;
224     Y=0;
225     LCD_Clear();
226 }
227
```