

```

1
2  /*****
3  * C source with CCS
4  *   File Name: 20120405_MK316_accl_sensor.c
5  *   Description: check potentiometer and ACCL sensor out
6  *   , and turn on yellow LED and relay
7  *   PIN A0 for potentiometer
8  *   PIN A1 for y axis
9  *   PIN A2 for switch with x out and z out
10 *   PIN A3 for start
11 *   PIN A4 for Relay, active high
12 *   PIN A5 for LED, active low
13 *   COPYRIGHT 2012 MYCOMKITS.COM, owned by CNET LIMITED
14 *   当プログラムの著作権は、製作者「マイコンキットドットコム運営 有限会社クネット」に帰属します。
15 *   著作権を放棄していませんが、当プログラムを使った学習の中でプログラムを自由に変更してお使いください。
16 *****/
17 // include header file
18 #include <12f683.h>
19 #DEVICE ADC=8
20
21 // settings
22 #fuses INTRC IO, NOWDT, NOPUT, NOPROTECT, NOMCLR
23 #use delay(CLOCK = 4000000)
24
25 //
26 long value_setting=256;
27 long value_sensor=256;
28 float value_sensorx=256;
29 float value_sensory=256;
30 signed int value x=0;
31 signed int value y=0;
32 signed int Xoffset=0; //x offset, -127 to +127
33 signed int Yoffset=0; //y offset, -127 to +127
34 long value sx[17];
35 long value sy[17];
36 int hyste = 0;
37 int i;
38 //
39 //
40 // prototyping
41 #separate
42 void get_sensor();
43 #separate
44 void get_setting();
45 #separate
46 void initializing();
47 #separate
48 void check_adc();
49 //
50 //
51 // main
52 //
53 void main()
54 {
55     //
56     initializing(); //ADC port initialize
57     output_low(PIN_A4); //active high, Relay off
58     output_high(PIN_A5); //active low, LED off
59     //
60     //calibration
61     delay_ms(2000);
62     //
63     check_adc();
64     //
65     Xoffset = value_x; //
66     Yoffset = value_y; //
67     //
68     output_low(PIN_A5); // LED on
69     delay_ms(100);
70     output_high(PIN_A5); // LED off
71     delay_ms(100);
72     output_low(PIN_A5); // LED on
73     delay_ms(100);
74     output_high(PIN_A5); // LED off
75     //
76     // main loop
77     while(1)
78     {
79         if(input(PIN_A3)==1)
80         {

```

```

81         delay_ms(50);    //check it again for chattering
82         if(input(PIN_A3)==1)
83         {
84             //
85             get_setting();
86             get_sensor();
87             //
88             if(value_sensor > value_setting*0.9)
89             {
90                 output_low(PIN_A5); // LED on
91             }
92             else
93             {
94                 output_high(PIN_A5);    // LED off
95             }
96             if(value_sensor > value_setting)
97             {
98
99                 if(hyste == 0)
100                 {
101                     output_high(PIN_A4);    // Relay on
102                     hyste = 50;
103                 }
104             }
105             //else
106             else if(hyste == 0)
107             {
108                 output_low(PIN_A4); // Relay off
109                 hyste = 50;
110             }
111         }
112     }
113 }
114 }
115 //
116 //
117 #separate
118 void get_setting()
119 {
120     set_adc_channel(0); //must wait 65u
121     delay_us(65);
122     value_setting = read_adc(); // 0 to 256, 0.0V to 3.3V
123     value_setting = 20 * value_setting; // 0 to 10000
124 }
125 //
126 #separate
127 void get_sensor()
128 {
129     // moving average for X axis and Y or Z
130     // A1 shows X axis, A2 shows Z or Y axis
131     // moving average
132     int i;
133     for (i=15; i>0; i--)
134     {
135         value_sx[i] = value_sx[i-1];
136         value_sy[i] = value_sy[i-1];
137     }
138     //
139     check_adc();
140     //
141     value_x = value_x - Xoffset;
142     value_y = value_y - Yoffset;
143     value_sensorx = value_x;
144     value_sensory = value_y;
145     //
146     value_sx[0] = value_sensorx * value_sensorx;    // 0 to 8100
147     value_sy[0] = value_sensory * value_sensory;    // 0 to 8100
148     // summing
149     value_sensorx = 0;
150     value_sensory = 0;
151     for (i=0; i<16; i++)
152     {
153         value_sensorx = value_sensorx + value_sx[i];    // 0 to 129600
154         value_sensory = value_sensory + value_sy[i];    // 0 to 129600
155     }
156     //
157     //
158     value_sensorx = value_sensorx / 16; // 0 to 8100
159     value_sensory = value_sensory / 16; // 0 to 8100
160     value_sensor = value_sensorx + value_sensory; // 0 to 16200

```

```

161 }
162 //
163 // system intializing
164 //
165 #separate
166 void initializing()
167 {
168     //
169     SET_TRIS_A(0x0F);    //A0 to 3 are input, 4, 5 are output
170     //
171     // A/D converter initialize AN1 and AN2
172     setup_adc_ports(sAN0 | sAN1 | sAN2 | VSS VDD);    // gnd to 5v
173     setup_adc(ADC_CLOCK_DIV_8);    // 2usec
174     //
175     // timer 1 intialize for hysteresis count down
176     //
177     setup_timer_1(T1_INTERNAL|T1_DIV_BY_8);
178     set_timer1(0xF4F0);    // timer 1 setting almost 5msec
179     enable_interrupts(INT_TIMER1);    // timer 1 enable
180     enable_interrupts(GLOBAL);    // global enable
181     //
182     //
183 }
184 //
185 // timer 1 inturruption
186 // hysteresis count down
187 //
188 #int timer1
189 void isr1(void)
190 {
191     set_timer1(0xF4F0);    // timer 1 almost 5msec
192     //
193     if(hyste > 1) hyste = hyste - 1;
194     else hyste = 0;
195     //
196 }
197 //
198 #separate
199 void check_adc()
200 {
201     //
202     set_adc_channel(1);    //must wait 65u
203     delay_us(65);
204     value_x = read_adc();    // convert it into floating, 0 to 255
205     value_x = value_x - 127;    //-90 to 90, 0.4V to 2.8V
206     set_adc_channel(2);    //must wait 65u
207     delay_us(65);
208     value_y = read_adc();    // convert it into floating, 0 to 256
209     value_y = value_y - 127;    //-90 to 90, 0.4V to 2.8V
210 }
211 //
212 //
213

```