

```

1 #include "Functions.h"
2
3 int main(void)
4 {
5     // define variables
6     int N_step, N_delay;
7     unsigned long y_max, y_min, sp_x;
8
9     // ADC&DAC setting
10    ADCpoweron(20000); // power on ADC
11    REFCON = 0x01; // internal 2.5V reference
12    DAC0CON = 0x12; // AGND-ARef range 0x12 2.5V
13    ADCCP = 0x00; // conversion on ADC0
14    ADCCON = 0x6E4; // continuous conversion
15
16    // IO setting
17    GP1CON = 0x00000000; // IO initialization
18    GP1DAT = 0xFF000000; // set group 1 as output
19    GP0CON = 0x00000000; // IO initialization
20    GP0DAT = 0x00000000; // set group 0 as input
21
22    N_step = 1; // step size
23    N_delay = 500; // wait for certain time
24
25    Write_Digital(0,0); //Write a low level digital value on pin P1.0 for indicating out of locking
26
27    while(1){
28        Max_Min(&y_max, &y_min); // sweeping for obtaining the maximum and minimum values
29
30        while(Read_Digital(3) == 0){ // switch between sweep mode and locking mode;
31            Max_Min(&y_max, &y_min);
32            Write_Digital(0,0); //Write a low level digital value on pin P1.0 for indicating in the sweeping
mode
33        }
34
35        Start_Point( y_max, y_min, &sp_x); //find start point for the locking
36
37        Lock_Maximum_Searching(N_step, N_delay, y_max, y_min, sp_x); // Locking based on maximum_searching
algorithm
38        //Lock_PI_Like(N_step, N_delay, y_max, y_min, sp_x); // Locking based on PI-like algorithm
39    }
40 }
41
42

```

```

1 #include<aduc7020.h>
2
3 void ADCpoweron(int);
4 void Delay(int);
5 int Read_Digital(int n);
6 void Write_Digital(int n, int state);
7 void Delay(int);
8 unsigned long Bound(unsigned long N);
9 int ADCtoDAT(unsigned long ADC);
10 unsigned long DATtoADC(int DAT);
11 void Max_Min(unsigned long *y_max, unsigned long *y_min);
12 void Start_Point(unsigned long y_max, unsigned long y_min, unsigned long *sp_x);
13 void Lock_Maximum_Searching(int N_step, int N_delay, unsigned long y_max, unsigned long y_min, unsigned
long sp_x);
14 void Lock_PI_Like(int N_gain, int N_delay, unsigned long y_max, unsigned long y_min, unsigned long sp_x);
15
16 // flag for sweep direction
17 struct bs
18 {
19     unsigned c0:1;
20     unsigned c1:1;
21     unsigned c2:1;
22     unsigned c3:1;
23     unsigned c4:1;
24     unsigned c5:1;
25     unsigned c6:1;
26     unsigned c7:1;
27 };
28 struct bs flag = {0,0,0,0,0,0,0,0};
29
30 // read digital value from the pin P0.n
31 int Read_Digital(int n)
32 {
33     return ((GP0DAT&0x000000FF) >> n) & 0x1;
34 }
35
36 // write digital value to the pin P1.n
37 void Write_Digital(int n, int state)
38 {
39     if(state == 1)
40         GP1DAT = (0x00000001<<(n + 16)) | GP1DAT;
41     else
42         GP1DAT = ~((0x00000001<<(n + 16)) | (~GP1DAT));
43 }
44
45 // wait for ADC to be fully powered on
46 void ADCpoweron(int time)
47 {
48     ADCCON = 0x620; // power-on the ADC
49     while (time >= 0)
50         time--;
51 }
52
53 // convert ADC/DAC format to integer format
54 int ADCtoDAT(unsigned long ADC)
55 {
56     return (ADC&0xFFFF0000)>>16;
57 }
58
59 // convert integer format to ADC/DAC format
60 unsigned long DATtoADC(int DAT)
61 {
62     unsigned long ADC;
63     ADC = DAT;
64     return ADC<<16;
65 }

```

```

66
67 // wait for a certain time
68 void Delay(int cnt)
69 {
70     while(cnt>0) cnt--;
71 }
72
73 // full scanning for finding the max and min value
74 void Max_Min(unsigned long *y_max, unsigned long *y_min)
75 {
76     int i;
77     unsigned long V, Ymax, Ymin;
78
79     Ymax = 0;
80     Ymin = 0xFFFF0000;
81     V = 0;
82
83     if(flag.c0 == 0){ // make a ramp sweeping
84         for(i = 10; i <= 4080; i += 10){
85             DAC0DAT = DATtoADC(i);
86             Delay(100); // wait for PZT
87             V = ADCDAT;
88             if(V > Ymax) Ymax = V;
89             if(V < Ymin) Ymin = V;
90         }
91     }
92     else {
93         for(i = 4080; i >= 10; i -= 10){
94             DAC0DAT = DATtoADC(i);
95             Delay(100); // wait for PZT
96             V = ADCDAT;
97             if(V > Ymax) Ymax = V;
98             if(V < Ymin) Ymin = V;
99         }
100    }
101    flag.c0++;
102    *y_max = Ymax;
103    *y_min = Ymin;
104 }
105
106 // find start point for the locking
107 void Start_Point(unsigned long y_max, unsigned long y_min, unsigned long *sp_x)
108 {
109     int i;
110     unsigned long V_th, V;
111     V_th = (y_max - y_min) * 3/4 + y_min; // set the threshold
112
113     if(flag.c0 == 0){
114         for(i = 10; i <= 4080; i++){
115             DAC0DAT = DATtoADC(i);
116             Delay(100); // wait for certain time for the responce of PZT
117             V = ADCDAT;
118             if(V > V_th){
119                 *sp_x = DATtoADC(i);
120                 break;
121             }
122         }
123     }
124     else{
125         for(i = 4080; i >= 10; i--){
126             DAC0DAT = DATtoADC(i);
127             Delay(100); // wait for certain time for the responce of PZT
128             V = ADCDAT;
129             if(V > V_th){
130                 *sp_x = DATtoADC(i);
131                 break;
132             }
133         }
134     }
135 }
```

```

132         }
133     }
134 }
135 flag.c0++;
136 }
137
138
139 // Bound the output value of DAC, when V reaches the limit of DAC range
140 unsigned long Bound(unsigned long V)
141 {
142     if (V > 0xFF60000)
143         return 0xFF60000;
144     else if(V < 0xA0000)
145         return 0xA0000;
146     else
147         return V;
148 }
149
150
151
152 // Locking based on maximum_searching algorithm
153 void Lock_Maximum_Searching(int N_step, int N_delay, unsigned long y_max, unsigned long y_min, unsigned long sp_x)
154 {
155     int flag, k, sum, N;
156     unsigned long V1, V2, V_th, V;
157
158     flag = 1; // indicator for searching direction
159     V_th = (y_max-y_min)/4+y_min; // set the threshold
160     V = sp_x; // initialized with start ponit
161     V1 = ADCDAT; // read ADC value
162     N = 200; // accumulation number for each locking step
163
164
165     while(1){
166         if(Read_Digital(3) == 0 || V1 < V_th) break; // lower than the threshold or enabling of sweep mode will break out of the while
167         Write_Digital(0,1); // Output a high level digital output on pin P1.0 for indicating in the locking mode
168
169         V = V + flag * DATtoADC(N_step); // calculate the voltage for next step
170         V = Bound(V); // limit the range of V
171         DAC0DAT = V; // output voltage on DAC0
172         Delay(N_delay); //wait for a certain time for the response of PZT
173
174         sum = 0; // initialization
175         for(k = 0; k < N; k++){
176             while (!ADCSTA){} // wait for end of ADC conversion
177             sum += ADCtoDAT(ADCDAT); // read voltage from ADC0
178         }
179         V2 = DATtoADC(sum/N); // calculate average value for the voltage of second step
180
181         if(V2 < V1) flag = -1 * flag; // change maximum searching direction if V2 < V1
182
183         V1 = V2; // update the voltage of first step
184
185     }
186     Write_Digital(0,0); //Write a low level digital value on pin P1.0 for indicating out of locking
187 }
188
189 // Locking based on PI-like algorithm
190 void Lock_PI_Like(int N_gain, int N_delay, unsigned long y_max, unsigned long y_min, unsigned long sp_x)
191 {
192     int i, N, diff, gp, gi, I, error;
193     unsigned long V, V1, V2, sum, V_th;
194

```

```

195     N = 100;
196     V = sp_x;
197     V_th = (y_max-y_min)/4+y_min; // set the threshold
198     gp = 1;
199     gi = 1;
200     I = 0;
201     error = 0;
202
203     while(1){
204         Write_Digital(0,1); // Output a high level digital output on pin P1.0 for indicating in the locking
mode
205
206         sum = 0; // initialization
207         for(i = 1; i <= N; i++){
208             while (!ADCSTA){} // wait for end of ADC conversion
209             sum += ADCtoDAT(ADCDAT);
210         }
211         V1=DATtoADC(sum/N); // calculte average value for the voltage of second step
212
213         if(Read_Digital(3) == 0 || V1 < V_th) break; // lower than the threshold or enabling of sweep mode
(P0.3) will break out of the while
214
215         V=V+DATtoADC(1);
216         V=Bound(V);
217         DAC0DAT=V;
218         Delay(N_delay);
219
220         sum = 0;
221         for(i = 1; i <= N; i++){
222             while (!ADCSTA){}
223             sum += ADCtoDAT(ADCDAT);
224         }
225         V2 = DATtoADC(sum/N);
226
227         diff = ADCtoDAT(V2) - ADCtoDAT(V1);
228
229         I = I + diff;
230         error = I * gi / 2 + diff * gp / 5;
231         V = sp_x + DATtoADC(error * N_gain);
232
233         V = Bound(V); // limit the range of V
234         DAC0DAT = V; // output voltage on DAC0
235         Delay(N_delay); //wait for a certain time for the response of PZT
236     }
237     Write_Digital(0,0); //Write a low level digital value on pin P1.0 for indicating out of locking
238 }
```