1 FinalProject.c

Jonathan Le, Justin Marcy, Marharyta Pliazhuk

```c
#include <stdio.h>
#include "board.h"
#include "peripherals.h"
#include "pin_mux.h"
#include "clock_config.h"
#include "MKL25Z4.h"
#include "fsl_debug_console.h"

void calcAndDisplServoAngle(int);
void setServoPositionToAngle(int);

void UART0_init(void);
void keypad_init(void);
char keypad_getkey(void);
void PWM_init(void);
void LCD_init(void);
void LCD_nibble_write(unsigned char data, unsigned char control);
void LCD_command(unsigned char command);
void LCD_data(unsigned char data);

void delayMs(int n);
void delayUs(int n);

#define RS 4
#define EN 8
#define mod 2047

short int servoMin = 1320, servoMax = 7320;
short int servoScanSpeed = 10;
char prevKeypadKeys[] = "aa";
char prevKeyboardKey = 'a';
int ldr_pot = 3;
unsigned char hexKeys[] =
                {' ','A','3','2','1','B','6','5','4','C','9','8','7','D','#','0','*'};
int counter1 = 0, counter2 = 0;
char c = '`';
unsigned char key, keypressed;
short int result;
char displayString1[] = " Potentiometer  ";
char displayString2[] = " Photoresistor  ";
char displayString3[] = "   Servo scan   ";
char displayString4[] = "  Servo manual  ";
char displayString5[] = "   DC manual    ";

int main(void) {


        BOARD_InitBootClocks();
        __disable_irq();
        PWM_init();
        ADC0_init();
        UART0_init();
        keypad_init();
        LCD_init();
        __enable_irq();
        LCD_command(1);
        delayMs(8);
```

```
58
59          while(1)
60          {
61                  TPM0_IRQHandler();
62          }
63  }
64
65  void PORTD_IRQHandler(void) {
66          unsigned char key = keypad_getkey();
67          char string[16];
68
69          keypressed = hexKeys[key];
70          PRINTF("Key pressed = %c\r\n", keypressed);
71          prevKeypadKeys[0] = prevKeypadKeys[1];
72          prevKeypadKeys[1] = keypressed;
73
74          if (strcmp(prevKeypadKeys,"#1") == 0) {
75                  strncpy(string, displayString1, 16);
76                  delayUs(8); UART0->D = '#';
77                  delayUs(8); UART0->D = '1';
78                  for (int i=0; i<sizeof(string); i++){
79                          LCD_command(0x80|i);
80                          LCD_data(string[i]);
81                          UART0->D = string[i];
82                  }
83                  delayUs(8); UART0->D = '\r';
84                  ldr_pot = 3;
85          }
86          else if (strcmp(prevKeypadKeys,"#2") == 0) {
87                  strncpy(string, displayString2, 16);
88                  delayUs(8); UART0->D = '#';
89                  delayUs(8); UART0->D = '2';
90                  for (int i=0; i<sizeof(string); i++){
91                          LCD_command(0x80|i);
92                          LCD_data(string[i]);
93                          UART0->D = string[i];
94                  }
95                  delayUs(8); UART0->D = '\r';
96                  ldr_pot = 0;
97          }
98          else if (strcmp(prevKeypadKeys,"#3") == 0) {
99                  strncpy(string, displayString3, 16);
100                 delayUs(8); UART0->D = '#';
101                 delayUs(8); UART0->D = '3';
102                 for (int i=0; i<sizeof(string); i++){
103                         LCD_command(0x80|i);
104                         LCD_data(string[i]);
105                         UART0->D = string[i];
106                 }
107                 delayUs(8); UART0->D = '\r';
108         }
109         else if (strcmp(prevKeypadKeys,"#4") == 0) {
110                 strncpy(string, displayString4, 16);
111                 delayUs(8); UART0->D = '#';
112                 delayUs(8); UART0->D = '4';
113                 for (int i=0; i<sizeof(string); i++){
114                         LCD_command(0x80|i);
115                         LCD_data(string[i]);
116                         UART0->D = string[i];
117                 }
118                 delayUs(8); UART0->D = '\r';
119         }
```

```
120         else if (strcmp(prevKeypadKeys,"#5") == 0) {
121                 strncpy(string, displayString5, 16);
122                 delayUs(8); UART0->D = '#';
123                 delayUs(8); UART0->D = '5';
124                 for (int i=0; i<sizeof(string); i++){
125                         LCD_command(0x80|i);
126                         LCD_data(string[i]);
127                         UART0->D = string[i];
128                 }
129                 delayUs(8); UART0->D = '\r';
130         }
131
132         PTD->PDDR |= 0x0F;
133         PTD->PCOR = 0x0F;
134         PORTD->ISFR |= 0xF0;
135 }
136
137 void keypad_init(void) {
138         SIM->SCGC5    |= 0x1000;
139         PORTD->PCR[0] = 0x103;
140         PORTD->PCR[1] = 0x103;
141         PORTD->PCR[2] = 0x103;
142         PORTD->PCR[3] = 0x103;
143         PORTD->PCR[4] = 0x103;
144         PORTD->PCR[5] = 0x103;
145         PORTD->PCR[6] = 0x103;
146         PORTD->PCR[7] = 0x103;
147         PTD->PDDR     = 0x0F;
148
149         PORTD->PCR[7] &= ~0xF0000;
150         PORTD->PCR[7] |= 0xA0000;
151         PORTD->PCR[6] &= ~0xF0000;
152         PORTD->PCR[6] |= 0xA0000;
153         PORTD->PCR[5] &= ~0xF0000;
154         PORTD->PCR[5] |= 0xA0000;
155         PORTD->PCR[4] &= ~0xF0000;
156         PORTD->PCR[4] |= 0xA0000;
157
158         NVIC_EnableIRQ(PORTD_IRQn);
159 }
160
161 char keypad_getkey(void) {
162         int row, col;
163         const char row_select[] = {0x01, 0x02, 0x04, 0x08};
164
165         for (row = 0; row < 4; row++)
166         {
167                 PTD->PDDR = 0;
168                 PTD->PDDR |= row_select[row];
169                 PTD->PCOR = row_select[row];
170                 delayUs(2);
171                 col = PTD->PDIR & 0xF0;
172                 if (col != 0xF0) break;
173         }
174
175         if (col == 0xE0) return row * 4 + 1;
176         if (col == 0xD0) return row * 4 + 2;
177         if (col == 0xB0) return row * 4 + 3;
178         if (col == 0x70) return row * 4 + 4;
179
180         return 0;
181 }
```

```
182
183 void LCD_init(void) {
184         SIM->SCGC5 |= 0x800;
185         PORTC->PCR[8] = 0x100;
186         PORTC->PCR[9] = 0x100;
187         PORTC->PCR[10] = 0x100;
188         PORTC->PCR[11] = 0x100;
189         PORTC->PCR[12] = 0x100;
190         PORTC->PCR[13] = 0x100;
191         PTC->PDDR |= 0x3F00;
192
193         delayMs(30);
194         LCD_nibble_write(0x30, 0);
195         delayMs(10);
196         LCD_nibble_write(0x30, 0);
197         delayMs(1);
198         LCD_nibble_write(0x30, 0);
199         delayMs(1);
200         LCD_nibble_write(0x20, 0);
201         delayMs(1);
202
203         LCD_command(0x28);
204         LCD_command(0x06);
205         LCD_command(0x01);
206         LCD_command(0x0F);
207 }
208
209 void LCD_nibble_write(unsigned char data, unsigned char control) {
210         data &= 0xF0;
211         control &= 0x0F;
212         PTC->PDOR |= (data | control) << 6;
213         PTC->PDOR |= (data | control | EN) << 6;
214         delayMs(0);
215         PTC->PDOR = data;
216         PTC->PDOR = 0;
217 }
218
219 void LCD_command(unsigned char command) {
220         LCD_nibble_write(command & 0xF0, 0);
221         LCD_nibble_write(command << 4, 0);
222
223         if (command < 4)
224                 delayMs(4);
225         else
226                 delayMs(1);
227 }
228
229 void LCD_data(unsigned char data) {
230         LCD_nibble_write(data & 0xF0, RS);
231         LCD_nibble_write(data << 4, RS);
232         delayMs(1);
233 }
234
235 void UART0_IRQHandler(void) {
236         c = UART0->D;
237         UART0->D = c;
238
239         char string[16];
240         if (prevKeyboardKey == '#'){
241                 switch(c){
242                 case '1':
243                         prevKeyboardKey = c;
```

```
244                         strncpy(string, displayString1, 16);
245                         for (int i=0; i<sizeof(string); i++){
246                                 LCD_command(0x80|i);
247                                 LCD_data(string[i]);
248                                 UART0->D = string[i];
249                         }
250                         delayUs(8); UART0->D = '\r';
251                         ldr_pot = 3;
252                         break;
253
254                 case '2':
255                         prevKeyboardKey = c;
256                         strncpy(string, displayString2, 16);
257                         for (int i=0; i<sizeof(string); i++){
258                                 LCD_command(0x80|i);
259                                 LCD_data(string[i]);
260                                 UART0->D = string[i];
261                         }
262                         delayUs(8); UART0->D = '\r';
263                         ldr_pot = 0;
264                         break;
265
266                 case '3':
267                         prevKeyboardKey = c;
268                         strncpy(string, displayString3, 16);
269                         for (int i=0; i<sizeof(string); i++){
270                                 LCD_command(0x80|i);
271                                 LCD_data(string[i]);
272                                 UART0->D = string[i];
273                         }
274                         for(int i = servoMin; i < servoMax; i += servoScanSpeed) {
275                                 TPM0->CONTROLS[1].CnV = i;
276                                 calcAndDisplServoAngle(i);
277                         }
278                         for(int i = servoMax; i > servoMin; i -= servoScanSpeed) {
279                                 TPM0->CONTROLS[1].CnV = i;
280                                 calcAndDisplServoAngle(i);
281                         }
282                         delayUs(8); UART0->D = '\r';
283                         break;
284
285                 case '4':
286                         prevKeyboardKey = c;
287                         strncpy(string, displayString4, 16);
288                         for (int i=0; i<sizeof(string); i++){
289                                 LCD_command(0x80|i);
290                                 LCD_data(string[i]);
291                                 UART0->D = string[i];
292
293                         }
294                         delayUs(8); UART0->D = '\r';
295                         break;
296
297                 case '5':
298                         prevKeyboardKey = c;
299                         strncpy(string, displayString5, 16);
300                         for (int i=0; i<sizeof(string); i++){
301                                 LCD_command(0x80|i);
302                                 LCD_data(string[i]);
303                                 UART0->D = string[i];
304                         }
305                         delayUs(8); UART0->D = '\r';
```

```
306                         break;
307
308                 default:
309                         prevKeyboardKey = c;
310                         UART0->D = c;
311                         break;
312                 }
313         }
314         else {
315                 prevKeyboardKey = c;
316         }
317         PORTA->ISFR = 0x10;
318 }
319
320 void UART0_init(void) {
321         SIM->SCGC4 |= SIM_SCGC4_UART0(1);
322         SIM->SOPT2 |= SIM_SOPT2_UART0SRC(1);
323         UART0->C2   = 0;
324         UART0->BDH  = UART0_BDH_SBR(0);
325         UART0->BDL  = UART0_BDL_SBR(26);
326         UART0->C4   = UART0_C4_OSR(15);
327         UART0->C1   = UART0_C1_M(0);
328         UART0->C2   = 0x2C;
329
330         NVIC->ISER[0] |= 0x00001000;
331
332         SIM->SCGC5     = SIM_SCGC5_PORTA(1);
333         PORTA->PCR[1]  = PORT_PCR_MUX(2);
334         PORTA->PCR[2]  = PORT_PCR_MUX(2);
335 }
336
337 void PWM_init(void) {
338
339         SIM->SCGC5 |= 0x0800;
340         SIM->SCGC6 |= 0x01000000;
341         PORTC->PCR[2] = 0x0400;
342
343         SIM->SOPT2 |= 0x01000000;
344
345         TPM0->SC = 0;
346         TPM0->CONTROLS[1].CnSC = 0x20|0x08;
347         TPM0->MOD = 60000;
348         TPM0->CONTROLS[1].CnV = 1500;
349         TPM0->SC = 0x0C;
350
351
352         SIM->SCGC5 |= 0x0400;
353         SIM->SCGC6 |= 0x04000000;
354         PORTB->PCR[3] = 0x0300;
355         TPM2->SC = 0;
356         TPM2->CONTROLS[1].CnSC = 0x20|0x08;
357         TPM2->MOD = mod;
358         TPM2->CONTROLS[1].CnV = mod/2;
359         TPM2->SC |= 0x80;
360         TPM2->SC |= 0x40;
361         TPM2->SC |= 0x0B;
362
363
364         SIM->SCGC5 |= 0x0400;
365         SIM->SCGC6 |= 0x02000000;
366         PORTB->PCR[1] = 0x0300;
367         TPM1->SC = 0;
```

```
368         TPM1->CONTROLS[1].CnSC = 0x20|0x08;
369         TPM1->MOD = mod;
370         TPM1->CONTROLS[1].CnV = mod/2;
371         TPM1->SC |= 0x80;
372         TPM1->SC |= 0x40;
373         TPM1->SC |= 0x0B;
374
375         NVIC_EnableIRQ(TPM0_IRQn);
376         NVIC_EnableIRQ(TPM1_IRQn);
377         NVIC_EnableIRQ(TPM2_IRQn);
378 }
379
380 void ADC0_init(void) {
381         uint16_t calibration;
382
383         SIM->SCGC5 |= 0x2000;
384
385         PORTE->PCR[22] = 0;
386         PORTE->PCR[20] = 0;
387
388         SIM->SCGC6 |= 0x8000000;
389         ADC0->SC2 &= ~0x40;
390
391         ADC0->CFG1 = 0x40 | 0x10 | 0x04 | 0x00;
392
393         ADC0->SC3 |= ADC_SC3_CAL_MASK;
394         while (ADC0->SC3 & ADC_SC3_CAL_MASK) { }
395
396         calibration = 0x0;
397
398         calibration += ADC0->CLP0;
399         calibration += ADC0->CLP1;
400         calibration += ADC0->CLP2;
401         calibration += ADC0->CLP3;
402         calibration += ADC0->CLP4;
403         calibration += ADC0->CLPS;
404
405         calibration /= 2;
406
407         calibration |= 0x8000;
408
409         ADC0->PG = calibration;
410
411         calibration = 0x0000;
412         calibration += ADC0->CLM0;
413         calibration += ADC0->CLM1;
414         calibration += ADC0->CLM2;
415         calibration += ADC0->CLM3;
416         calibration += ADC0->CLM4;
417         calibration += ADC0->CLMS;
418         calibration /= 2;
419         calibration |= 0x8000;
420         ADC0->MG = calibration;
421
422
423         ADC0->CFG1 = 0x40 | 0x10 | 0x04 | 0x00;
424 }
425
426 void TPM0_IRQHandler(void) {
427
428         NVIC_DisableIRQ(TPM2_IRQn);
429         NVIC_DisableIRQ(TPM1_IRQn);
```

```
430
431         ADC0->SC1[0] = ldr_pot;
432         while(!(ADC0->SC1[0] & 0x80)) { }
433         result = ADC0->R[0];
434         TPM0->MOD = 60000;
435         TPM0->CONTROLS[1].CnV = servoMin + result*(servoMax - servoMin)/4095;
436         TPM0->SC |= 0x80;
437         PORTD->ISFR = 0x10;
438         if (strcmp(prevKeypadKeys,"#3")==0) {
439                 for(int i = servoMin; i < servoMax; i += servoScanSpeed) {
440                         TPM0->CONTROLS[1].CnV = i;
441                         calcAndDisplServoAngle(i);
442                 }
443                 for(int i = servoMax; i > servoMin; i -= servoScanSpeed) {
444                         TPM0->CONTROLS[1].CnV = i;
445                         calcAndDisplServoAngle(i);
446                 }
447         } else {
448                 TPM0->CONTROLS[1].CnV = servoMin + result*(servoMax-servoMin)/4095;
449                 calcAndDisplServoAngle(TPM0->CONTROLS[1].CnV);
450         }
451
452         PORTD->ISFR = 0x10;
453         TPM0->SC |= 0x80;
454
455         NVIC_EnableIRQ(TPM2_IRQn);
456         NVIC_EnableIRQ(TPM1_IRQn);
457 }
458
459 void TPM1_IRQHandler(void) {
460
461         NVIC_DisableIRQ(TPM0_IRQn);
462         NVIC_DisableIRQ(TPM2_IRQn);
463
464         ADC0->SC1[0] = ldr_pot;
465         while(!(ADC0->SC1[0] & 0x80)) { }
466         result = ADC0->R[0];
467         TPM1->MOD = mod*2;
468         TPM1->CONTROLS[1].CnV = result;
469         TPM1->SC |= 0x80;
470
471         NVIC_EnableIRQ(TPM0_IRQn);
472         NVIC_EnableIRQ(TPM2_IRQn);
473 }
474
475 void TPM2_IRQHandler(void) {
476
477         NVIC_DisableIRQ(TPM0_IRQn);
478         NVIC_DisableIRQ(TPM1_IRQn);
479
480         ADC0->SC1[0] = ldr_pot;
481         while(!(ADC0->SC1[0] & 0x80)) { }
482         result = ADC0->R[0];
483         TPM2->MOD = 2400 + result*(12000-2400)/4095;
484         TPM2->CONTROLS[1].CnV = TPM2->MOD/2;
485         TPM2->SC |= 0x80;
486
487         NVIC_EnableIRQ(TPM0_IRQn);
488         NVIC_EnableIRQ(TPM1_IRQn);
489 }
490
491 void calcAndDisplServoAngle(int i) {
```

```
492         int servoAngle;
493         servoAngle = (i - servoMin) * 180 / (servoMax - servoMin) - 90;
494         LCD_command(0xC0);
495         LCD_data((servoAngle < 0 ? '-' : '+'));
496         LCD_command(0xC1);
497         LCD_data(abs(servoAngle) / 10 + 0x30);
498         LCD_command(0xC2);
499         LCD_data(abs(servoAngle) % 10 + 0x30);
500         LCD_command(0xC0);
501 }
502
503 void setServoPositionToAngle(int servoAngle) {
504         int i = (servoAngle + 90) * (servoMax - servoMin) / 180 + servoMin;
505         delayMs(5); TPM0->CONTROLS[1].CnV = i; delayMs(5);
506         LCD_command(0xC0);
507         LCD_data((servoAngle < 0 ? '-' : '+'));
508         LCD_command(0xC1);
509         LCD_data(abs(servoAngle) / 10 + 0x30);
510         LCD_command(0xC2);
511         LCD_data(abs(servoAngle) % 10 + 0x30);
512 }
513
514 void delayUs(int n) {
515         for(int i = 0 ; i < n; i++) {
516                 for(int j = 0; j < 5; j++);
517         }
518 }
519
520 void delayMs(int n) {
521         for(int i = 0 ; i < n; i++)
522                 for(int j = 0 ; j < 3500; j++) { }
523 }
```