



Prifysgol Cymru
Y Drindod Dewi Sant
University of Wales
Trinity Saint David



ACDA4013 Embedded C

Assignment

Matthew Prentice (2116359)

Abstract

LEARNING OUTCOMES

Understand the syntax and semantics of an embedded C program and develop programs using good programming practices. Select and deploy basic software constructs using simple algorithms in the development of solutions to embedded programming problems

Contents

0 Section 1	1
A Appendix - Assignment Code	2

List of Figures

1	count.asm circuit	1
---	-----------------------------	---

0 Section 1

The circuit shown below has been made available to you on moodle, named 'Embedded C assignment circuit'. You

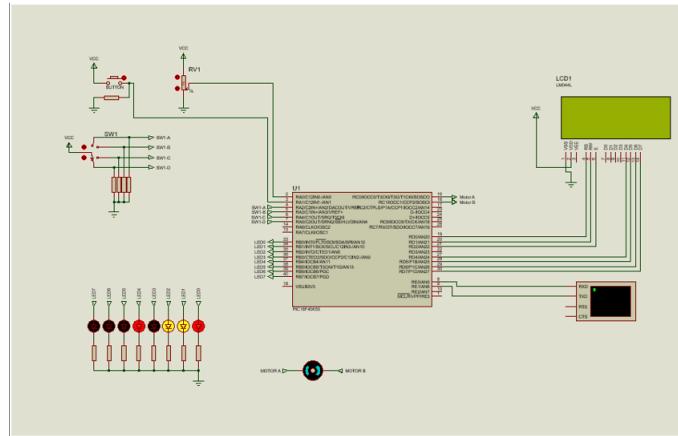


Figure 1: count.asm circuit

must write a program that fulfils the following brief. Brief.

- 1 When SW1-A is selected the digital output of the ADC reading RV1 must be shown on the LEDs, and the value must be displayed centred on line 1 of the LCD and echoed onto the virtual terminal.
2. When SW1-B is selected the value of RC1 will drive Motor A, if the value is below 75 the motor will drive clockwise, if the value is above 175 the motor will drive anti clockwise. If the value is between 76 and 174 the motor will stop. The value of the reading from the ADC must be displayed along with the direction of rotation on line 2 of the display.
3. When SW1-C is selected pressing the button will make Motor A rotate clockwise as long as the button is depressed, and the motor will stop when released. The status of the button is to be displayed on the LCD on line 3 and echoed once on the virtual terminal.
4. When SW1-D is selected the LEDs will flash from left to right and back again in knight rider fashion changing @ 150mS intervals. The message "Kitt mode" will be displayed centred on line 4 of the LCD display and echoed once on the virtual terminal.

The program should be well structured and be remarked well enough that anyone with knowledge of C will be able to understand the flow of the program.

You should write your own functions, as required, to keep the program compact and readable.

A Appendix - Assignment Code

```

1  /*
2  /  Module:      L4 – Embedded C (EES) (ACDA4013)      /  Functions:
3  /  Task Title:  Assignment                          /
4  /  Author:     M Prentice (2116359)                /  print2LCD – string ,
5  /  line , position (L,C,R)                        /
6  /  manage printing to the LCD                      /  Function to
7  /  Objectives: On setting a switch (SW1) to position A, B, C /  read in string to
8  /  print , starting line number                    /
9  /  or D the circuit should execute the following actions: /  and position (L –
10 /  left , C – centred , R – right)                  /
11 /  A– Read in analog value from RV1 and display on LEDs (0xFF) /  TODO Return –
12 /  overflow .                                         /
13 /  and displayed on the LCD (centred Line 1) and on /
14 /  the virtual terminal                             /
15 /  B– Read in analog value from RV1 and rotate Motor A: /  readRV1
16 /  value at RV1, pin AN0                             /  read in analog
17 /  clockwise if RV1 < 75                             /
18 /  no rotation if 75 < RV1 < 175                     /  motorSpin – start /
19 /  stop , 1 / 0                                       /
20 /  anti-clockwise if 175 < RV1                     /  start or stop the
21 /  spin for the motor                               /
22 /  direction                                         /  in a set
23 /  C– If the push button (BUTTON) is depressed , rotate the /  1– Clockwise
24 /  0– Anti-Clockwise                               /
25 /  motor clockwise. if not, don't rotate.           /
26 /  Display button status on the LCD (line 3) and on /  knightRider
27 /  virtual terminal                                 /  light the led's
28 /  in epic-ness                                     /
29 /  D– Flash LED's knight rider style at 150mS intervals /  readSw
30 /  Display "Kitt mode" on the LCD (centred , Line 4) /  read in the status of
31 /  the rotary switch and button                     /
32 /  and also echoed on virtual terminal               /  use mask 0x3C
33 /  */
34 /  */
35 /  */
36 /  */
37 /  */
38 /  */
39 /  */
40 /  */
41 /  */
42 /  */
43 /  */
44 /  */
45 /  */
46 /  */
47 /  */
48 /  */
49 /  */
50 /  */
51 /  */
52 /  */
53 /  */
54 /  */
55 /  */
56 /  */
57 /  */
58 /  */
59 /  */
60 /  */
61 /  */
62 /  */
63 /  */
64 /  */
65 /  */
66 /  */
67 /  */
68 /  */
69 /  */
70 /  */
71 /  */
72 /  */
73 /  */
74 /  */
75 /  */
76 /  */
77 /  */
78 /  */
79 /  */
80 /  */
81 /  */
82 /  */
83 /  */
84 /  */
85 /  */
86 /  */
87 /  */
88 /  */
89 /  */
90 /  */
91 /  */
92 /  */
93 /  */
94 /  */
95 /  */
96 /  */
97 /  */
98 /  */
99 /  */
100 /  */
101 /  */
102 /  */
103 /  */
104 /  */
105 /  */
106 /  */
107 /  */
108 /  */
109 /  */
110 /  */
111 /  */
112 /  */
113 /  */
114 /  */
115 /  */
116 /  */
117 /  */
118 /  */
119 /  */
120 /  */
121 /  */
122 /  */
123 /  */
124 /  */
125 /  */
126 /  */
127 /  */
128 /  */
129 /  */
130 /  */
131 /  */
132 /  */
133 /  */
134 /  */
135 /  */
136 /  */
137 /  */
138 /  */
139 /  */
140 /  */
141 /  */
142 /  */
143 /  */
144 /  */
145 /  */
146 /  */
147 /  */
148 /  */
149 /  */
150 /  */
151 /  */
152 /  */
153 /  */
154 /  */
155 /  */
156 /  */
157 /  */
158 /  */
159 /  */
160 /  */
161 /  */
162 /  */
163 /  */
164 /  */
165 /  */
166 /  */
167 /  */
168 /  */
169 /  */
170 /  */
171 /  */
172 /  */
173 /  */
174 /  */
175 /  */
176 /  */
177 /  */
178 /  */
179 /  */
180 /  */
181 /  */
182 /  */
183 /  */
184 /  */
185 /  */
186 /  */
187 /  */
188 /  */
189 /  */
190 /  */
191 /  */
192 /  */
193 /  */
194 /  */
195 /  */
196 /  */
197 /  */
198 /  */
199 /  */
200 /  */
201 /  */
202 /  */
203 /  */
204 /  */
205 /  */
206 /  */
207 /  */
208 /  */
209 /  */
210 /  */
211 /  */
212 /  */
213 /  */
214 /  */
215 /  */
216 /  */
217 /  */
218 /  */
219 /  */
220 /  */
221 /  */
222 /  */
223 /  */
224 /  */
225 /  */
226 /  */
227 /  */
228 /  */
229 /  */
230 /  */
231 /  */
232 /  */
233 /  */
234 /  */
235 /  */
236 /  */
237 /  */
238 /  */
239 /  */
240 /  */
241 /  */
242 /  */
243 /  */
244 /  */
245 /  */
246 /  */
247 /  */
248 /  */
249 /  */
250 /  */
251 /  */
252 /  */
253 /  */
254 /  */
255 /  */
256 /  */
257 /  */
258 /  */
259 /  */
260 /  */
261 /  */
262 /  */
263 /  */
264 /  */
265 /  */
266 /  */
267 /  */
268 /  */
269 /  */
270 /  */
271 /  */
272 /  */
273 /  */
274 /  */
275 /  */
276 /  */
277 /  */
278 /  */
279 /  */
280 /  */
281 /  */
282 /  */
283 /  */
284 /  */
285 /  */
286 /  */
287 /  */
288 /  */
289 /  */
290 /  */
291 /  */
292 /  */
293 /  */
294 /  */
295 /  */
296 /  */
297 /  */
298 /  */
299 /  */
300 /  */
301 /  */
302 /  */
303 /  */
304 /  */
305 /  */
306 /  */
307 /  */
308 /  */
309 /  */
310 /  */
311 /  */
312 /  */
313 /  */
314 /  */
315 /  */
316 /  */
317 /  */
318 /  */
319 /  */
320 /  */
321 /  */
322 /  */
323 /  */
324 /  */
325 /  */
326 /  */
327 /  */
328 /  */
329 /  */
330 /  */
331 /  */
332 /  */
333 /  */
334 /  */
335 /  */
336 /  */
337 /  */
338 /  */
339 /  */
340 /  */
341 /  */
342 /  */
343 /  */
344 /  */
345 /  */
346 /  */
347 /  */
348 /  */
349 /  */
350 /  */
351 /  */
352 /  */
353 /  */
354 /  */
355 /  */
356 /  */
357 /  */
358 /  */
359 /  */
360 /  */
361 /  */
362 /  */
363 /  */
364 /  */
365 /  */
366 /  */
367 /  */
368 /  */
369 /  */
370 /  */
371 /  */
372 /  */
373 /  */
374 /  */
375 /  */
376 /  */
377 /  */
378 /  */
379 /  */
380 /  */
381 /  */
382 /  */
383 /  */
384 /  */
385 /  */
386 /  */
387 /  */
388 /  */
389 /  */
390 /  */
391 /  */
392 /  */
393 /  */
394 /  */
395 /  */
396 /  */
397 /  */
398 /  */
399 /  */
400 /  */
401 /  */
402 /  */
403 /  */
404 /  */
405 /  */
406 /  */
407 /  */
408 /  */
409 /  */
410 /  */
411 /  */
412 /  */
413 /  */
414 /  */
415 /  */
416 /  */
417 /  */
418 /  */
419 /  */
420 /  */
421 /  */
422 /  */
423 /  */
424 /  */
425 /  */
426 /  */
427 /  */
428 /  */
429 /  */
430 /  */
431 /  */
432 /  */
433 /  */
434 /  */
435 /  */
436 /  */
437 /  */
438 /  */
439 /  */
440 /  */
441 /  */
442 /  */
443 /  */
444 /  */
445 /  */
446 /  */
447 /  */
448 /  */
449 /  */
450 /  */
451 /  */
452 /  */
453 /  */
454 /  */
455 /  */
456 /  */
457 /  */
458 /  */
459 /  */
460 /  */
461 /  */
462 /  */
463 /  */
464 /  */
465 /  */
466 /  */
467 /  */
468 /  */
469 /  */
470 /  */
471 /  */
472 /  */
473 /  */
474 /  */
475 /  */
476 /  */
477 /  */
478 /  */
479 /  */
480 /  */
481 /  */
482 /  */
483 /  */
484 /  */
485 /  */
486 /  */
487 /  */
488 /  */
489 /  */
490 /  */
491 /  */
492 /  */
493 /  */
494 /  */
495 /  */
496 /  */
497 /  */
498 /  */
499 /  */
500 /  */
501 /  */
502 /  */
503 /  */
504 /  */
505 /  */
506 /  */
507 /  */
508 /  */
509 /  */
510 /  */
511 /  */
512 /  */
513 /  */
514 /  */
515 /  */
516 /  */
517 /  */
518 /  */
519 /  */
520 /  */
521 /  */
522 /  */
523 /  */
524 /  */
525 /  */
526 /  */
527 /  */
528 /  */
529 /  */
530 /  */
531 /  */
532 /  */
533 /  */
534 /  */
535 /  */
536 /  */
537 /  */
538 /  */
539 /  */
540 /  */
541 /  */
```

```

LCD and Virtual Terminal "Kitt mode"
51 unsigned int readSw(unsigned int mask); // read the status of rotary
    switch and button into variable of your choice. Rotary Switch Mask 0x3C, Button mask 0x40
52
53
54 void main()
55 {
56     printf("System booting up\r");
57     set_tris_b(0b11111111);
58     lcd_init();
59     setup_adc_ports(sAN0); // initialise adc channel
60     setup_adc(ADC_CLOCK_INTERNAL|ADC_TAD_MUL_0); // setup adc operation
61     unsigned int selection;
62     unsigned int btnPress;
63     printf("\fSetup Done\r");
64     printf(lcd_putc("\fSetup Done"));
65     delay_ms(500);
66     while(TRUE)
67     {
68         selection = readSw(0x3C); // apply mask 0x3C aka
00111100
69         //printf(" \r Got a switch position! - %u ",selection ); // Useful
        debug, display the registered Switch position on virtual terminal
        //printf(lcd_putc("\f"));
        switch(selection)
        {
        72         {
        73             case 4: // SW1-A - Read in analog value from RV1 and display on LEDs (0xFF) and
                displayed on the LCD (centred Line 1) and on the virtual terminal
                //printf("\fSW A - Read in RV1 then display");
        74                 readRV1(); // Read the value of RV1
        75                 via analog pin AN0
                output_b(adc); // Display adc value on
        76                 LED's as Binary number
                print2LCD("adc",1,'C'); // Display adc value on
        77                 LCD
                printf("\f\rRV1 value: %u", adc); // Display adc value on
        78                 Virtual Terminal, %u means unsigned decimal integer
        79
                break;
        80
        81             case 8: // SW1-B - Read in analog value from RV1 and rotate Motor A: clockwise if
                RV1 < 75, no rotation if 75 < RV1 < 175, anti-clockwise if 175 < RV1
                readRV1();
                printf("\r\rRV1 value: %u", adc); // Display adc value on
        82                 Virtual Terminal, %u means unsigned decimal integer
                printf(lcd_putc("\f"));
        83                 if(adc<75){ //clockwise if RV1 < 75
        84                     motorSpin('t', 1);
        85                 }else if(adc>75 && adc<175){ //no rotation if 75 < RV1 <
        86                     175
        87                     motorSpin('p', 0);
        88                 }else if(adc>175){ //anti-clockwise if 175 < RV1
        89                     motorSpin('t', 0);
        90                 }
        91                 break;
        92
        93             case 16: // SW1-C
                btnPress = readSw(0x02); //Button (RA1) mask: 0x02
                printf("\fSW C - Motor spin based on Button - %u ",btnPress);
        94
                if(btnPress==2){
        95                     motorSpin('t', 1);
        96                     print2LCD("BTN Pressed",3,'C');
        97                     printf("\rBTN Pressed ---");
        98                 }else{
        99                     motorSpin('p', 1);
        100                     print2LCD("BTN off",3,'C');
        101                     printf("\rBTN Off —");
        102                 }
        103                 break;
        104
        105             case 32: // SW1-D
                printf("\fSW B - Flashy Lights");
        106                 print2LCD("Kitt mode",4,'C');
        107
        108
        109

```

```

110         printf("\rKitt mode");
111         knightRider();
112         break;
113     default:
114
115         break;
116     }
117 }
118 }
119
120 //#####
121 //     NAME: print2LCD
122 //     DESCRIPTION: Function to manage printing to the LCD read in string to print ,
123 //                  starting line number and position
124 //     ARGUMENTS:  char* string  - text to be printed to LCD
125 //                  int line     - line to print
126 //                  char position - string alignment (L - left , C - centred , R - right)
127 //     RETURN:
128 //#####
129 //TODO work out overflow return , do I limit string to 20 char or overflow to next line.
130 void print2LCD(char* string ,int line ,char position)
131 {
132     int pos = 0;
133     int Len;
134     if(strcmp(string , "adc")==0){
135         Len = 3;
136     }else{
137         Len = strlen(string); //sizeof(string);
138     }
139     switch (position)
140     {
141         case 'L': pos = 1;
142                 printf("\rLeft - %d" , pos);
143                 break;
144         case 'C': pos = 10 - (Len/2);
145                 printf("\rCentral - %d" , pos);
146                 break;
147         case 'R': pos = 21 - Len;
148                 printf("\rRight - %d" , pos);
149                 break;
150         default : printf("\rDefault - %d" , pos);
151                 break;
152     }
153     printf(lcd_putc("\f"));
154     lcd_gotoxy(pos ,line);
155     if(strcmp(string , "adc")==0){
156         printf(lcd_putc,"%u" , adc);
157     }else{
158         printf(lcd_putc,"%s" , string);
159     }
160 }
161
162 //#####
163 //     NAME: readRV1
164 //     DESCRIPTION: Function to manage reading the analog value from RV1 via analog pin AN0
165 //                  reads into global variable adc
166 //     ARGUMENTS:  nil
167 //     RETURN:     value into global variable adc
168 //#####
169 void readRV1(void)
170 {
171     set_adc_channel(0);           //read adc channel 0
172     delay_us(30);                 // wait 30uS
173     adc = read_adc();             // store value in variable ADC
174 }
175
176 //#####
177 //     NAME: motorSpin
178 //     DESCRIPTION: Function to manage the spin of the motor
179 //     ARGUMENTS:  char* status  - starT / stoP
180 //                  int dir      - 1-Clockwise , 0-Anti-Clockwise

```

```

181 // RETURN:      nil
182 //#####
183 void motorSpin(char* status, int dir) // Status (start/stop) dir (1–Clockwise, 0–Anti–
    Clockwise)
184 {
185     switch (status)
186     {
187         case 't': if(dir == 1)                // starT
188             {
189                 // Spin motor in one direction, set B low and A high
190                 output_low(MotorB);
191                 output_high(MotorA);
192                 printf(" – Motor A");
193             }
194             else if (dir == 0)
195             {
196                 // Spin motor in the other direction, set A low and B high
197                 output_low(MotorA);
198                 output_high(MotorB);
199                 printf(" – Motor B");
200             }
201             else
202             {
203                 // error case, don't spin motor, set A low and B low
204                 output_low(MotorA);
205                 output_low(MotorB);
206                 printf(" – start Fail");
207             }
208             break;
209         case 'p': output_low(MotorA);          // stoP, don't spin the motor, set A low and B low
210                 output_low(MotorB);
211                 printf(" – Stop");
212                 break;
213         default: output_low(MotorA);           // Default situation, don't spin the motor, set A low
214                 and B low
215                 output_low(MotorB);
216                 printf(" – Default (Stop) ");
217                 break;
218     }
219 }
220 //#####
221 // NAME: knightRider
222 // DESCRIPTION: Function to make the LED's flash
223 // ARGUMENTS: char* status – starT / stoP
224 // int dir – 1–Clockwise, 0–Anti–Clockwise
225 // RETURN:      nil
226 //#####
227 void knightRider(void)
228 {
229     unsigned int kitt = 0x80;
230     while(kitt){
231         output_b(kitt);
232         delay_ms(150);
233         kitt = kitt >> 1;
234     }
235     delay_ms(150);
236     kitt = 0x01;
237     while(kitt){
238         output_b(kitt);
239         delay_ms(150);
240         kitt = kitt << 1;
241     }
242 }
243 //#####
244 // NAME: readSW
245 // DESCRIPTION: Function to read the status of the rotart switch and button
246 // ARGUMENTS: unsigned int mask – mask on the Port A pins
247 // RETURN:      nil
248 //#####
249 unsigned int readSw(unsigned int mask){
    unsigned int buttons=0;

```

```
250     buttons=input_a() & mask;
251     return(buttons);
252 }
```