Judah Ben-Eliezer

112352727
11/17//2020

Prelab 11:

Sensors, Basic Analog-to-Digital Conversion, and the ATmega4809's 10-bit ADC

```
 1
 2  AVRASM ver. 2.2.7  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11
      \post_display\post_display\main.asm Tue Nov 17 11:36:27 2020
 3
 4  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\post_display\post_display
      \main.asm(12): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 5  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\post_display\post_display
      \main.asm(12): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 6
 7
 8                                    ; post_display.asm
 9                                    ;
10                                    ; Created: 11/17/2020 11:13:12 AM
11                                    ; Author : hp
12                                    ;
13
14
15                                    ; Replace with your application code
16
17                                    .list
18
19                                    .equ PERIOD_EXAMPLE_VALUE = 25
20
21                                    reset:
22  000000 940c 0010                     jmp start
23
24                                    .org TCA0_OVF_vect
25  00000e 940c 002e                     jmp toggle_pins_ISR
26
27                                    start:
28                                       ;configure PORTC and PORTD and output FF to
                          both
29  000010 ef0f                          ldi r16, $FF
30  000011 b90c                          out VPORTD_DIR, r16
31  000012 b908                          out VPORTC_DIR, r16
32  000013 b90d                          out VPORTD_OUT, r16
33  000014 b909                          out VPORTC_OUT, r16
34
35                                       ;configure TCA0
36  000015 e000                          ldi r16, TCA_SINGLE_WGMODE_NORMAL_gc
      ;WGMODE normal
37  000016 9300 0a01                     sts TCA0_SINGLE_CTRLB, r16
38
39                                       ;enable overflow interrupt
40  000018 e001                          ldi r16, TCA_SINGLE_OVF_bm
41  000019 9300 0a0a                     sts TCA0_SINGLE_INTCTRL, r16
42
```

```
43                                      ;load period low byte then high byte
44  00001b e109                         ldi r16, LOW(PERIOD_EXAMPLE_VALUE)
45  00001c 9300 0a26                    sts TCA0_SINGLE_PER, r16
46  00001e e000                         ldi r16, HIGH(PERIOD_EXAMPLE_VALUE)
47  00001f 9300 0a27                    sts TCA0_SINGLE_PER + 1, r16
48
49                                      ;set clock and start timer
50  000021 e00d                         ldi r16, TCA_SINGLE_CLKSEL_DIV256_gc |
      TCA_SINGLE_ENABLE_bm
51  000022 9300 0a00                    sts TCA0_SINGLE_CTRLA, r16
52
53  000024 e000                         ldi r16, $00
54  000025 b90d                         out VPORTD_OUT, r16
55
56  000026 9478                         sei     ;enable global interrupts
57
58
59                                  ;*****************************************
                                ******************************
60                                  ;*
61                                  ;* "post_display"
62                                  ;*
63                                  ;* Description: toggles value for all PORTC
                                pins.  Since PORTC is used to multiplex the led display,
                                this will
64                                  ;* turn the LED display on and off
65                                  ;* Author:  Judah Ben-Eliezer
66                                  ;* Version:    1.0
67                                  ;* Last updated:   11/17
68                                  ;* Target:  ATmega4809
69                                  ;* Number of words:    13
70                                  ;* Number of cycles:   6
71                                  ;* Low registers modified:
72                                  ;* High registers modified:
73                                  ;* Parameters:  none
74                                  ;* Returns:        none
75                                  ;*
76                                  ;* Notes:
77                                  ;*
78                                  ;*****************************************
                                ******************************
79                                  post_display:
80  000027 ef1f                         ldi r17, $FF
81  000028 b109                         in r16, VPORTC_OUT
82  000029 2701                         eor r16, r17
83  00002a b909                         out VPORTC_OUT, r16
84  00002b 9508                         ret
85
86                                  main_loop:
```

```
 87  00002c 0000                        nop
 88  00002d cffe                        rjmp main_loop
 89
 90                                     ;*****************************************
                     ****************************
 91                                     ;*
 92                                     ;* "toggle_pins_ISR" - title
 93                                     ;*
 94                                     ;* Description:        ISR to toggle PORTC
                     pins, called whenever timing buffer TCA0 overflows
 95                                     ;*
 96                                     ;* Author: Judah Ben-Eliezer
 97                                     ;* Version:    1.0
 98                                     ;* Last updated:   11/17/2020
 99                                     ;* Target: ATmega4809
100                                     ;* Number of words:    27
101                                     ;* Number of cycles:   12
102                                     ;* Low registers modified:
103                                     ;* High registers modified:
104                                     ;*
105                                     ;* Parameters: none
106                                     ;* Returns:    none
107                                     ;*
108                                     ;* Notes:
109                                     ;*
110                                     ;*****************************************
                     ****************************
111                                    toggle_pins_ISR:
112 00002e 930f                           push r16
113 00002f b70f                           in r16, CPU_SREG
114 000030 930f                           push r16
115 000031 931f                           push r17
116
117 000032 dff4                           rcall post_display  ;call subroutine to
      toggle display
118
119 000033 e001                           ldi r16, TCA_SINGLE_OVF_bm  ;clear OVF flag
120 000034 9300 0a0b                      sts TCA0_SINGLE_INTFLAGS, r16
121
122 000036 911f                           pop r17
123 000037 910f                           pop r16
124 000038 bf0f                           out CPU_SREG, r16
125 000039 910f                           pop r16
126
127 00003a 9518                           reti
128
129
130 RESOURCE USE INFORMATION
131 ------------------------
```

```
132
133 Notice:
134 The register and instruction counts are symbol table hit counts,
135 and hence implicitly used resources are not counted, eg, the
136 'lpm' instruction without operands implicitly uses r0 and z,
137 none of which are counted.
138
139 x,y,z are separate entities in the symbol table and are
140 counted separately from r26..r31 here.
141
142 .dseg memory usage only counts static data declared with .byte
143
144 "ATmega4809" register use summary:
145 x  :   0 y  :   0 z  :   0 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
146 r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   0
147 r13:   0 r14:   0 r15:   0 r16:  28 r17:   4 r18:   0 r19:   0 r20:   0
148 r21:   0 r22:   0 r23:   0 r24:   0 r25:   0 r26:   0 r27:   0 r28:   0
149 r29:   0 r30:   0 r31:   0
150 Registers used: 2 out of 35 (5.7%)
151
152 "ATmega4809" instruction use summary:
153 .lds  :   0 .sts  :   0 adc   :   0 add   :   0 adiw  :   0 and   :   0
154 andi  :   0 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
155 brcc  :   0 brcs  :   0 break :   0 breq  :   0 brge  :   0 brhc  :   0
156 brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
157 brne  :   0 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
158 brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :   0 cbr   :   0
159 clc   :   0 clh   :   0 cli   :   0 cln   :   0 clr   :   0 cls   :   0
160 clt   :   0 clv   :   0 clz   :   0 com   :   0 cp    :   0 cpc   :   0
161 cpi   :   0 cpse  :   0 dec   :   0 des   :   0 eor   :   1 fmul  :   0
162 fmuls :   0 fmulsu:   0 icall :   0 ijmp  :   0 in    :   2 inc   :   0
163 jmp   :   2 ld    :   0 ldd   :   0 ldi   :   9 lds   :   0 lpm   :   0
164 lsl   :   0 lsr   :   0 mov   :   0 movw  :   0 mul   :   0 muls  :   0
165 mulsu :   0 neg   :   0 nop   :   1 or    :   0 ori   :   0 out   :   7
166 pop   :   3 push  :   3 rcall :   1 ret   :   1 reti  :   1 rjmp  :   1
167 rol   :   0 ror   :   0 sbc   :   0 sbci  :   0 sbi   :   0 sbic  :   0
168 sbis  :   0 sbiw  :   0 sbr   :   0 sbrc  :   0 sbrs  :   0 sec   :   0
169 seh   :   0 sei   :   1 sen   :   0 ser   :   0 ses   :   0 set   :   0
170 sev   :   0 sez   :   0 sleep :   0 spm   :   0 st    :   0 std   :   0
171 sts   :   6 sub   :   0 subi  :   0 swap  :   0 tst   :   0 wdr   :   0
172
173 Instructions used: 14 out of 114 (12.3%)
174
175 "ATmega4809" memory use summary [bytes]:
176 Segment   Begin    End      Code   Data   Used    Size   Use%
177 ----------------------------------------------------------------
178 [.cseg] 0x000000 0x000076     94      0     94   49152   0.2%
179 [.dseg] 0x002800 0x002800      0      0      0    6144   0.0%
180 [.eseg] 0x000000 0x000000      0      0      0     256   0.0%
```

```
181
182  Assembly complete, 0 errors, 0 warnings
183
```

```
 1
 2  AVRASM ver. 2.2.7  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11
      \ADC_sgnl_conv\ADC_sgnl_conv\main.asm Tue Nov 17 18:18:10 2020
 3
 4  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\ADC_sgnl_conv\ADC_sgnl_conv
      \main.asm(9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 5  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\ADC_sgnl_conv\ADC_sgnl_conv
      \main.asm(9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 6
 7
 8                                  ; ADC_sgnl_conv.asm
 9                                  ;
10                                  ; Created: 11/17/2020 2:02:20 PM
11                                  ; Author : Judah Ben-Eliezer
12                                  ;
13
14                                  .list
15
16                                  .equ PERIOD_EXAMPLE_VALUE = 25
17
18                                  .dseg
19  002800                          led_display: .byte 4
20  002804                          digit_num: .byte 1
21
22
23                                  .cseg
24
25                                  reset:
26  000000 940c 0010                   jmp start
27
28                                  .org TCA0_OVF_vect
29  00000e 940c 0038                   jmp post_display_ISR
30
31                                  start:
32                                     ;configure inputs and outputs
33  000010 9881                        cbi VPORTE_DIR, 1
34  000011 ef0f                        ldi r16, $FF
35  000012 b908                        out VPORTC_DIR, r16
36  000013 b90c                        out VPORTD_DIR, r16
37  000014 9500                        com r16
38  000015 b909                        out VPORTC_OUT, r16
39  000016 b90d                        out VPORTD_OUT, r16
40
41                                     ;configure TCA0
42  000017 e000                        ldi r16, TCA_SINGLE_WGMODE_NORMAL_gc
      ;WGMODE normal
43  000018 9300 0a01                   sts TCA0_SINGLE_CTRLB, r16
```

```
44
45                                      ;enable overflow interrupt
46  00001a e001                         ldi r16, TCA_SINGLE_OVF_bm
47  00001b 9300 0a0a                    sts TCA0_SINGLE_INTCTRL, r16
48
49                                      ;load period low byte then high byte
50  00001d e109                         ldi r16, LOW(PERIOD_EXAMPLE_VALUE)
51  00001e 9300 0a26                    sts TCA0_SINGLE_PER, r16
52  000020 e000                         ldi r16, HIGH(PERIOD_EXAMPLE_VALUE)
53  000021 9300 0a27                    sts TCA0_SINGLE_PER + 1, r16
54
55                                      ;set clock and start timer
56  000023 e00d                         ldi r16, TCA_SINGLE_CLKSEL_DIV256_gc |      ➲
      TCA_SINGLE_ENABLE_bm
57  000024 9300 0a00                    sts TCA0_SINGLE_CTRLA, r16
58
59                                      ;set voltage reference
60  000026 e200                         ldi r16, VREF_ADC0REFSEL_2V5_gc
61  000027 9300 00a0                    sts VREF_CTRLA, r16
62
63                                      ;select PE1/ AIN9
64  000029 e009                         ldi r16, ADC_MUXPOS_AIN9_gc
65  00002a 9300 0606                    sts ADC0_MUXPOS, r16
66
67                                      ;enable internal reference and set          ➲
                      prescaler to div 64
68  00002c e005                         ldi r16, ADC_PRESC_DIV64_gc |               ➲
      ADC_REFSEL_INTREF_gc
69  00002d 9300 0602                    sts ADC0_CTRLC, r16
70
71                                      ;set resolution to 10 bit and enable adc
72  00002f e001                         ldi r16, ADC_RESSEL_10BIT_gc |              ➲
      ADC_ENABLE_bm;
73  000030 9300 0600                    sts ADC0_CTRLA, r16
74
75                                      ;start conversion
76  000032 e001                         ldi r16, ADC_STCONV_bm;
77  000033 9300 0608                    sts ADC0_COMMAND, r16
78
79                                      ;enable interrupts
80  000035 9478                         sei
81  000036 940c 0047                    jmp wait_for_post
82
83                                    ;*****************************************  ➲
                      ****************************
84                                    ;*
85                                    ;* "post_display" - title
86                                    ;*
87                                    ;* Description: toggles value for all PORTC    ➲
```

```
                        pins.  Since PORTC is used to multiplex the led display,
                        this will
 88                                 ;* turn the LED display on and off
 89                                 ;* Author:  Judah Ben-Eliezer
 90                                 ;* Version:    1.0
 91                                 ;* Last updated:   11/17
 92                                 ;* Target:  ATmega4809
 93                                 ;* Number of words:    13
 94                                 ;* Number of cycles:   6
 95                                 ;* Low registers modified:
 96                                 ;* High registers modified:
 97                                 ;* Parameters:  none
 98                                 ;* Returns:         none
 99                                 ;*
100                                 ;* Notes:
101                                 ;*
102                                 ;*****************************************
                        ******************************
103                                 post_display_ISR:
104 000038 930f                        push r16
105 000039 b70f                        in r16, CPU_SREG
106 00003a 930f                        push r16
107 00003b 931f                        push r17
108
109 00003c ef1f                        ldi r17, $FF
110 00003d b109                        in r16, VPORTC_OUT
111 00003e 2701                        eor r16, r17
112 00003f b909                        out VPORTC_OUT, r16
113
114                                    ;ldi r16, TCA_SINGLE_OVF_bm ;clear OVF flag
115                                    ;sts TCA0_SINGLE_INTFLAGS, r16
116
117 000040 911f                        pop r17
118 000041 910f                        pop r16
119 000042 bf0f                        out CPU_SREG, r16
120 000043 910f                        pop r16
121
122 000044 9478                        sei
123 000045 940c 0049                   jmp main_loop
124
125                                 wait_for_post:
126 000047 0000                        nop
127 000048 cffe                        rjmp wait_for_post
128
129                                 main_loop:
130 000049 d012                        rcall multiplex_display
131 00004a d026                        rcall mux_digit_delay
132 00004b 9130 060b                   lds r19, ADC0_INTFLAGS
133 00004d fd30                        sbrc r19, 0
```

```
134  00004e d001                        rcall read
135  00004f cff9                        rjmp main_loop
136
137                                     ;*********************************************** ↵
                        ******************************
138                                     ;*
139                                     ;* "read" - title
140                                     ;*
141                                     ;* Description: loads ADC0_RES into r17:r16    ↵
                        and calls bin16_to_led
142                                     ;*
143                                     ;* Author:  Judah Ben-Eliezer
144                                     ;* Version:     1.0
145                                     ;* Last updated:    11/17/2020
146                                     ;* Target:  ATmega4809
147                                     ;* Number of words:
148                                     ;* Number of cycles:
149                                     ;* Low registers modified:  none
150                                     ;* High registers modified:         r17:r16
151                                     ;*
152                                     ;* Parameters:  ADC0_RES
153                                     ;* Returns:         r17:r16
154                                     ;*
155                                     ;* Notes:
156                                     ;*
157                                     ;*********************************************** ↵
                        ******************************
158                                    read:
159  000050 9110 0611                      lds r17, ADC0_RESH
160  000052 9100 0610                      lds r16, ADC0_RESL
161  000054 d023                           rcall bin16_to_led
162
163                                        ;reset interrupt flag
164  000055 e001                           ldi r16, ADC_RESRDY_bm;
165  000056 9300 060a                      sts ADC0_INTCTRL, r16
166
167                                        ;restart conversion
168  000058 e001                           ldi r16, ADC_STCONV_bm;
169  000059 9300 0608                      sts ADC0_COMMAND, r16
170
171  00005b 9508                           ret
172
173                                     ;*********************************************** ↵
                        ******************************
174                                     ;*
175                                     ;* "multiplex_display" - title
176                                     ;*
177                                     ;* Description:    outputs values from         ↵
                        led_display array to 7 segment display on PORTD driven by   ↵
```

```
                        highest two bits of PORTC
178                                 ;*
179                                 ;* Author:  Judah Ben-Eliezer
180                                 ;* Version:     1.0
181                                 ;* Last updated:    11/10/2020
182                                 ;* Target:  ATmega4809
183                                 ;* Number of words:
184                                 ;* Number of cycles:
185                                 ;* Low registers modified:
186                                 ;* High registers modified:
187                                 ;*
188                                 ;* Parameters:
189                                 ;* Returns:
190                                 ;*
191                                 ;* Notes:
192                                 ;*
193                                 ;*****************************************
                        ******************************
194                                 multiplex_display:
195 00005c e2d8                         ldi YH, HIGH(led_display)
196 00005d e0c0                         ldi YL, LOW(led_display)
197 00005e 9110 2804                    lds r17, digit_num
198 000060 7013                         andi r17, $03
199 000061 2f41                         mov r20, r17
200 000062 0fc1                         add YL, r17
201 000063 8128                         ld r18, Y
202 000064 e850                         ldi r21, $80
203 000065 9543                         inc r20
204                                 loop:
205 000066 9556                         lsr r21
206 000067 954a                         dec r20
207 000068 f7e9                         brne loop
208 000069 0f55                         lsl r21
209 00006a 9550                         com r21
210 00006b b959                         out VPORTC_OUT, r21
211 00006c b92d                         out VPORTD_OUT, r18
212 00006d 9513                         inc r17
213 00006e 9310 2804                    sts digit_num, r17
214 000070 9508                         ret
215
216                                 ;*****************************************
                        ******************************
217                                 ;*
218                                 ;* "mux_digit_delay" - title
219                                 ;*
220                                 ;* Description: delays 0.1 * r23
221                                 ;*
222                                 ;* Author:  Judah Ben-Eliezer
223                                 ;* Version:     1.0
```

```
                                ;* Last updated:
                                ;* Target:
                                ;* Number of words:
                                ;* Number of cycles:
                                ;* Low registers modified:
                                ;* High registers modified:
                                ;*
                                ;* Parameters:
                                ;* Returns:
                                ;*
                                ;* Notes:
                                ;*
                                ;*********************************************
                        *****************************
                                mux_digit_delay:
000071 e078                        ldi r23, $08 ; 0.1 * r23 = delay
                                outer_loop:
000072 e086                        ldi r24, $06
                                inner_loop:
000073 958a                        dec r24
000074 f7f1                        brne inner_loop
000075 957a                        dec r23
000076 f7d9                        brne outer_loop
000077 9508                        ret

                                ;*********************************************
                        *****************************
                                ;*
                                ;* "bin16_to_led" - title
                                ;*
                                ;* Description:    Converts bin16 input to
                        7seg output, from bcd_entries array to led_display array
                                ;*
                                ;* Author: Judah Ben-Eliezer
                                ;* Version:    1.0
                                ;* Last updated:   11/17/2020
                                ;* Target: ATmega4809
                                ;* Number of words:
                                ;* Number of cycles:
                                ;* Low registers modified:
                                ;* High registers modified:
                                ;*
                                ;* Parameters: r17:r16 16 bit binary number.
                                ;* Returns:    none
                                ;*
                                ;* Notes:
                                ;*
                                ;*********************************************
                        *****************************
```

```
269
270                                 bin16_to_led:
271  000078 e2b8                       ldi XH, HIGH(led_display)
272  000079 e0a0                       ldi XL, LOW(led_display)
273  00007a 2f21                       mov r18, r17
274  00007b 7f20                       andi r18, $F0
275  00007c 9522                       swap r18
276  00007d d00f                       rcall hex_to_7seg
277  00007e 932d                       st X+, r18
278  00007f 2f21                       mov r18, r17
279  000080 702f                       andi r18, $0F
280  000081 d00b                       rcall hex_to_7seg
281  000082 932d                       st X+, r18
282  000083 2f20                       mov r18, r16
283  000084 7f20                       andi r18, $F0
284  000085 9522                       swap r18
285  000086 d006                       rcall hex_to_7seg
286  000087 932d                       st X+, r18
287  000088 2f20                       mov r18, r16
288  000089 702f                       andi r18, $0F
289  00008a d002                       rcall hex_to_7seg
290  00008b 932c                       st X, r18
291  00008c 9508                       ret
292
293                                 ;***************************************** ➲
                    ****************************
294                                 ;*
295                                 ;* "hex_to_7seg" - Hexadecimal to Seven    ➲
                    Segment Conversion
296                                 ;*
297                                 ;* Description: Converts a right justified ➲
                    hexadecimal digit to the seven
298                                 ;* segment pattern required to display it. ➲
                    Pattern is right justified a
299                                 ;* through g. Pattern uses 0s to turn segments ➲
                     on ON.
300                                 ;*
301                                 ;* Author:                      Ken Short
302                                 ;* Version:                         1.0    ➲

303                                 ;* Last updated:                101620
304                                 ;* Target:                      ATmega4809
305                                 ;* Number of words:                 8
306                                 ;* Number of cycles:            13
307                                 ;* Low registers modified:      none
308                                 ;* High registers modified:         r19, r18,  ➲
                    ZL, ZH
309                                 ;*
310                                 ;* Parameters: r18: right justified hex digit, ➲
```

```
                         high nibble 0
311                                 ;* Returns: r18: segment values a through g
                        right justified
312                                 ;*
313                                 ;* Notes:
314                                 ;*
315                                 ;*****************************************
                        ****************************
316
317                                 hex_to_7seg:
318 00008d 702f                        andi r18, 0x0F              ;clear ms
      nibble
319 00008e e0f1                         ldi ZH, HIGH(hextable * 2)  ;set Z to
      point to start of table
320 00008f e2ea                         ldi ZL, LOW(hextable * 2)
321 000090 e030                         ldi r19, $00                ;add offset to
       Z pointer
322 000091 0fe2                         add ZL, r18
323 000092 1ff3                         adc ZH, r19
324 000093 9124                         lpm r18, Z                  ;load byte
      from table pointed to by Z
325 000094 9508                        ret
326
327                                     ;Table of segment values to display digits
                         0 - F
328                                     ;!!! seven values must be added - verify
                        all values
329 000095 4f01
330 000096 0612
331 000097 244c
332 000098 0f20
333 000099 0400
334 00009a 6008
335 00009b 4231
336 00009c 3830                     hextable: .db $01, $4F, $12, $06, $4C, $24,
      $20, $0F, $00, $04, $08, $60, $31, $42, $30, $38
337
338
339 RESOURCE USE INFORMATION
340 ------------------------
341
342 Notice:
343 The register and instruction counts are symbol table hit counts,
344 and hence implicitly used resources are not counted, eg, the
345 'lpm' instruction without operands implicitly uses r0 and z,
346 none of which are counted.
347
348 x,y,z are separate entities in the symbol table and are
349 counted separately from r26..r31 here.
```

```
350
351 .dseg memory usage only counts static data declared with .byte
352
353 "ATmega4809" register use summary:
354 x  :   4 y  :   1 z  :   1 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
355 r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   0
356 r13:   0 r14:   0 r15:   0 r16:  42 r17:  13 r18:  19 r19:   4 r20:   3
357 r21:   5 r22:   0 r23:   2 r24:   2 r25:   0 r26:   1 r27:   1 r28:   2
358 r29:   1 r30:   2 r31:   2
359 Registers used: 17 out of 35 (48.6%)
360
361 "ATmega4809" instruction use summary:
362 .lds  :   0 .sts  :   0 adc   :   1 add   :   2 adiw  :   0 and   :   0
363 andi  :   6 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
364 brcc  :   0 brcs  :   0 break :   0 breq  :   0 brge  :   0 brhc  :   0
365 brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
366 brne  :   3 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
367 brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :   1 cbr   :   0
368 clc   :   0 clh   :   0 cli   :   0 cln   :   0 clr   :   0 cls   :   0
369 clt   :   0 clv   :   0 clz   :   0 com   :   2 cp    :   0 cpc   :   0
370 cpi   :   0 cpse  :   0 dec   :   3 des   :   0 eor   :   1 fmul  :   0
371 fmuls :   0 fmulsu:   0 icall :   0 ijmp  :   0 in    :   2 inc   :   2
372 jmp   :   4 ld    :   1 ldd   :   0 ldi   :  24 lds   :   4 lpm   :   2
373 lsl   :   1 lsr   :   1 mov   :   5 movw  :   0 mul   :   0 muls  :   0
374 mulsu :   0 neg   :   0 nop   :   1 or    :   0 ori   :   0 out   :   8
375 pop   :   3 push  :   3 rcall :   8 ret   :   5 reti  :   0 rjmp  :   2
376 rol   :   0 ror   :   0 sbc   :   0 sbci  :   0 sbi   :   0 sbic  :   0
377 sbis  :   0 sbiw  :   0 sbr   :   0 sbrc  :   1 sbrs  :   0 sec   :   0
378 seh   :   0 sei   :   2 sen   :   0 ser   :   0 ses   :   0 set   :   0
379 sev   :   0 sez   :   0 sleep :   0 spm   :   0 st    :   4 std   :   0
380 sts   :  13 sub   :   0 subi  :   0 swap  :   2 tst   :   0 wdr   :   0
381
382 Instructions used: 30 out of 114 (26.3%)
383
384 "ATmega4809" memory use summary [bytes]:
385 Segment   Begin    End      Code   Data   Used    Size   Use%
386 ----------------------------------------------------------------
387 [.cseg] 0x000000 0x00013a    274     16    290   49152   0.6%
388 [.dseg] 0x002800 0x002805      0      5      5    6144   0.1%
389 [.eseg] 0x000000 0x000000      0      0      0     256   0.0%
390
391 Assembly complete, 0 errors, 0 warnings
392
```

```
 1
 2  AVRASM ver. 2.2.7  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11
      \ADC_MCP9700A\ADC_MCP9700A\main.asm Tue Nov 17 18:37:40 2020
 3
 4  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\ADC_MCP9700A\ADC_MCP9700A
      \main.asm(9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 5  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\ADC_MCP9700A\ADC_MCP9700A
      \main.asm(9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs
      \atmel\ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 6
 7
 8                                  ; ADC_MCP9700A.asm
 9                                  ;
10                                  ; Created: 11/17/2020 2:02:20 PM
11                                  ; Author : Judah Ben-Eliezer
12                                  ;
13
14                                  .list
15
16                                  .equ PERIOD_EXAMPLE_VALUE = 25
17
18                                  .dseg
19  002800                          led_display: .byte 4
20  002804                          digit_num: .byte 1
21
22
23                                  .cseg
24
25                                  reset:
26  000000 940c 002e                   jmp start
27
28                                  .org TCA0_OVF_vect
29  00000e 940c 0056                   jmp post_display_ISR
30
31                                  .org ADC0_RESRDY_vect
32  00002c 940c 006a                   jmp read_ISR
33
34                                  start:
35                                     ;configure inputs and outputs
36  00002e 9883                        cbi VPORTE_DIR, 3
37  00002f ef0f                        ldi r16, $FF
38  000030 b908                        out VPORTC_DIR, r16
39  000031 b90c                        out VPORTD_DIR, r16
40  000032 9500                        com r16
41  000033 b909                        out VPORTC_OUT, r16
42  000034 b90d                        out VPORTD_OUT, r16
43
44                                     ;configure TCA0
```

```
 45  000035 e000                        ldi r16, TCA_SINGLE_WGMODE_NORMAL_gc
       ;WGMODE normal
 46  000036 9300 0a01                   sts TCA0_SINGLE_CTRLB, r16
 47
 48                                     ;enable overflow interrupt
 49  000038 e001                        ldi r16, TCA_SINGLE_OVF_bm
 50  000039 9300 0a0a                   sts TCA0_SINGLE_INTCTRL, r16
 51
 52                                     ;load period low byte then high byte
 53  00003b e109                        ldi r16, LOW(PERIOD_EXAMPLE_VALUE)
 54  00003c 9300 0a26                   sts TCA0_SINGLE_PER, r16
 55  00003e e000                        ldi r16, HIGH(PERIOD_EXAMPLE_VALUE)
 56  00003f 9300 0a27                   sts TCA0_SINGLE_PER + 1, r16
 57
 58                                     ;set clock and start timer
 59  000041 e00d                        ldi r16, TCA_SINGLE_CLKSEL_DIV256_gc |
       TCA_SINGLE_ENABLE_bm
 60  000042 9300 0a00                   sts TCA0_SINGLE_CTRLA, r16
 61
 62                                     ;set voltage reference
 63  000044 e200                        ldi r16, VREF_ADC0REFSEL_2V5_gc
 64  000045 9300 00a0                   sts VREF_CTRLA, r16
 65
 66                                     ;select PE1/ AIN9
 67  000047 e00b                        ldi r16, ADC_MUXPOS_AIN11_gc
 68  000048 9300 0606                   sts ADC0_MUXPOS, r16
 69
 70                                     ;enable internal reference and set
                        prescaler to div 64
 71  00004a e005                        ldi r16, ADC_PRESC_DIV64_gc |
       ADC_REFSEL_INTREF_gc
 72  00004b 9300 0602                   sts ADC0_CTRLC, r16
 73
 74                                     ;set resolution to 10 bit and enable adc
 75  00004d e001                        ldi r16, ADC_RESSEL_10BIT_gc |
       ADC_ENABLE_bm;
 76  00004e 9300 0600                   sts ADC0_CTRLA, r16
 77
 78                                     ;start conversion
 79  000050 e001                        ldi r16, ADC_STCONV_bm;
 80  000051 9300 0608                   sts ADC0_COMMAND, r16
 81
 82                                     ;enable interrupts
 83  000053 9478                        sei
 84  000054 940c 0065                   jmp wait_for_post
 85
 86                                    ;*****************************************
                        ****************************
 87                                    ;*
```

```
 88                                     ;* "post_display" - title
 89                                     ;*
 90                                     ;* Description: toggles value for all PORTC
                        pins.  Since PORTC is used to multiplex the led display,
                        this will
 91                                     ;* turn the LED display on and off
 92                                     ;* Author:  Judah Ben-Eliezer
 93                                     ;* Version:    1.0
 94                                     ;* Last updated:   11/17
 95                                     ;* Target:  ATmega4809
 96                                     ;* Number of words:    13
 97                                     ;* Number of cycles:   6
 98                                     ;* Low registers modified:
 99                                     ;* High registers modified:
100                                     ;* Parameters:  none
101                                     ;* Returns:         none
102                                     ;*
103                                     ;* Notes:
104                                     ;*
105                                     ;*****************************************
                        ******************************
106                                     post_display_ISR:
107 000056 930f                            push r16
108 000057 b70f                            in r16, CPU_SREG
109 000058 930f                            push r16
110 000059 931f                            push r17
111
112 00005a ef1f                            ldi r17, $FF
113 00005b b109                            in r16, VPORTC_OUT
114 00005c 2701                            eor r16, r17
115 00005d b909                            out VPORTC_OUT, r16
116
117                                        ;ldi r16, TCA_SINGLE_OVF_bm ;clear OVF flag
118                                        ;sts TCA0_SINGLE_INTFLAGS, r16
119
120 00005e 911f                            pop r17
121 00005f 910f                            pop r16
122 000060 bf0f                            out CPU_SREG, r16
123 000061 910f                            pop r16
124
125 000062 9478                            sei
126 000063 940c 0067                       jmp main_loop
127
128                                     wait_for_post:
129 000065 0000                            nop
130 000066 cffe                            rjmp wait_for_post
131
132                                     main_loop:
133 000067 d02a                            rcall multiplex_display
```

```
000068 d03e                         rcall mux_digit_delay
000069 cffd                         rjmp main_loop

                                ;*********************************************
                            ****************************
                                ;*
                                ;* "read_ISR" - title
                                ;*
                                ;* Description: loads ADC0_RES into r17:r16
                            and calls bin16_to_led
                                ;*
                                ;* Author:  Judah Ben-Eliezer
                                ;* Version:    1.0
                                ;* Last updated:   11/17/2020
                                ;* Target:  ATmega4809
                                ;* Number of words:
                                ;* Number of cycles:
                                ;* Low registers modified:  none
                                ;* High registers modified:        r17:r16
                                ;*
                                ;* Parameters:  ADC0_RES
                                ;* Returns:        r17:r16
                                ;*
                                ;* Notes:
                                ;*
                                ;*********************************************
                            ****************************
                               read_ISR:
00006a 930f                        push r16
00006b b70f                        in r16, CPU_SREG
00006c 930f                        push r16
00006d 2f0b                        mov r16, XH
00006e 930f                        push r16
00006f 2f0a                        mov r16, XL
000070 930f                        push r16
000071 2f0f                        mov r16, ZH
000072 930f                        push r16
000073 2f0e                        mov r16, ZL
000074 930f                        push r16
000075 931f                        push r17
000076 932f                        push r18
000077 933f                        push r19

000078 9110 0611                   lds r17, ADC0_RESH
00007a 9100 0610                   lds r16, ADC0_RESL
00007c d031                        rcall bin16_to_led

                                   ;reset interrupt flag
00007d e001                        ldi r16, ADC_RESRDY_bm;
```

```
180  00007e 9300 060a                    sts ADC0_INTCTRL, r16
181
182                                      ;restart conversion
183  000080 e001                         ldi r16, ADC_STCONV_bm;
184  000081 9300 0608                    sts ADC0_COMMAND, r16
185
186  000083 913f                         pop r19
187  000084 912f                         pop r18
188  000085 911f                         pop r17
189  000086 910f                         pop r16
190  000087 2fe0                         mov ZL, r16
191  000088 910f                         pop r16
192  000089 2ff0                         mov ZH, r16
193  00008a 910f                         pop r16
194  00008b 2fa0                         mov XL, r16
195  00008c 910f                         pop r16
196  00008d 2fb0                         mov XH, r16
197  00008e 910f                         pop r16
198  00008f bf0f                         out CPU_SREG, r16
199  000090 910f                         pop r16
200
201  000091 9518                         reti
202
203                                      ;******************************************** ↵
                     ******************************
204                                      ;*
205                                      ;* "multiplex_display" - title
206                                      ;*
207                                      ;* Description:    outputs values from      ↵
                     led_display array to 7 segment display on PORTD driven by   ↵
                     highest two bits of PORTC
208                                      ;*
209                                      ;* Author: Judah Ben-Eliezer
210                                      ;* Version:    1.0
211                                      ;* Last updated:   11/10/2020
212                                      ;* Target: ATmega4809
213                                      ;* Number of words:
214                                      ;* Number of cycles:
215                                      ;* Low registers modified:
216                                      ;* High registers modified:
217                                      ;*
218                                      ;* Parameters:
219                                      ;* Returns:
220                                      ;*
221                                      ;* Notes:
222                                      ;*
223                                      ;******************************************** ↵
                     ******************************
224                                      multiplex_display:
```

```
000092 e2d8                         ldi YH, HIGH(led_display)
000093 e0c0                         ldi YL, LOW(led_display)
000094 9110 2804                    lds r17, digit_num
000096 7013                         andi r17, $03
000097 2f41                         mov r20, r17
000098 0fc1                         add YL, r17
000099 8128                         ld r18, Y
00009a e850                         ldi r21, $80
00009b 9543                         inc r20
                                loop:
00009c 9556                         lsr r21
00009d 954a                         dec r20
00009e f7e9                         brne loop
00009f 0f55                         lsl r21
0000a0 9550                         com r21
0000a1 b959                         out VPORTC_OUT, r21
0000a2 b92d                         out VPORTD_OUT, r18
0000a3 9513                         inc r17
0000a4 9310 2804                    sts digit_num, r17
0000a6 9508                         ret

                                ;****************************************** ↵
                            ******************************
                                ;*
                                ;* "mux_digit_delay" - title
                                ;*
                                ;* Description: delays 0.1 * r23
                                ;*
                                ;* Author:  Judah Ben-Eliezer
                                ;* Version:    1.0
                                ;* Last updated:
                                ;* Target:
                                ;* Number of words:
                                ;* Number of cycles:
                                ;* Low registers modified:
                                ;* High registers modified:
                                ;*
                                ;* Parameters:
                                ;* Returns:
                                ;*
                                ;* Notes:
                                ;*
                                ;****************************************** ↵
                            ******************************
                                mux_digit_delay:
0000a7 e078                         ldi r23, $08 ; 0.1 * r23 = delay
                                outer_loop:
0000a8 e086                         ldi r24, $06
                                inner_loop:
```

```
272  0000a9 958a                        dec r24
273  0000aa f7f1                        brne inner_loop
274  0000ab 957a                        dec r23
275  0000ac f7d9                        brne outer_loop
276  0000ad 9508                        ret
277
278                                     ;****************************************** ↵
                        ****************************
279                                     ;*
280                                     ;* "bin16_to_led" - title
281                                     ;*
282                                     ;* Description:    Converts bin16 input to     ↵
                        7seg output, from bcd_entries array to led_display array
283                                     ;*
284                                     ;* Author: Judah Ben-Eliezer
285                                     ;* Version:    1.0
286                                     ;* Last updated:   11/17/2020
287                                     ;* Target: ATmega4809
288                                     ;* Number of words:
289                                     ;* Number of cycles:
290                                     ;* Low registers modified:
291                                     ;* High registers modified:
292                                     ;*
293                                     ;* Parameters: r17:r16 16 bit binary number.
294                                     ;* Returns:    none
295                                     ;*
296                                     ;* Notes:
297                                     ;*
298                                     ;****************************************** ↵
                        ****************************
299
300                                    bin16_to_led:
301  0000ae e2b8                           ldi XH, HIGH(led_display)
302  0000af e0a0                           ldi XL, LOW(led_display)
303  0000b0 2f21                           mov r18, r17
304  0000b1 7f20                           andi r18, $F0
305  0000b2 9522                           swap r18
306  0000b3 d00f                           rcall hex_to_7seg
307  0000b4 932d                           st X+, r18
308  0000b5 2f21                           mov r18, r17
309  0000b6 702f                           andi r18, $0F
310  0000b7 d00b                           rcall hex_to_7seg
311  0000b8 932d                           st X+, r18
312  0000b9 2f20                           mov r18, r16
313  0000ba 7f20                           andi r18, $F0
314  0000bb 9522                           swap r18
315  0000bc d006                           rcall hex_to_7seg
316  0000bd 932d                           st X+, r18
317  0000be 2f20                           mov r18, r16
```

```
318  0000bf 702f                         andi r18, $0F
319  0000c0 d002                         rcall hex_to_7seg
320  0000c1 932c                         st X, r18
321  0000c2 9508                         ret
322
323                                 ;*************************************************
                        ******************************
324                                 ;*
325                                 ;* "hex_to_7seg" - Hexadecimal to Seven
                        Segment Conversion
326                                 ;*
327                                 ;* Description: Converts a right justified
                        hexadecimal digit to the seven
328                                 ;* segment pattern required to display it.
                        Pattern is right justified a
329                                 ;* through g. Pattern uses 0s to turn segments
                         on ON.
330                                 ;*
331                                 ;* Author:                      Ken Short
332                                 ;* Version:                         1.0

333                                 ;* Last updated:                101620
334                                 ;* Target:                      ATmega4809
335                                 ;* Number of words:                 8
336                                 ;* Number of cycles:            13
337                                 ;* Low registers modified:      none
338                                 ;* High registers modified:         r19, r18,
                        ZL, ZH
339                                 ;*
340                                 ;* Parameters: r18: right justified hex digit,
                         high nibble 0
341                                 ;* Returns: r18: segment values a through g
                        right justified
342                                 ;*
343                                 ;* Notes:
344                                 ;*
345                                 ;*************************************************
                        ******************************
346
347                                hex_to_7seg:
348  0000c3 702f                        andi r18, 0x0F              ;clear ms
       nibble
349  0000c4 e0f1                         ldi ZH, HIGH(hextable * 2)  ;set Z to
       point to start of table
350  0000c5 e9e6                         ldi ZL, LOW(hextable * 2)
351  0000c6 e030                         ldi r19, $00                ;add offset to
        Z pointer
352  0000c7 0fe2                         add ZL, r18
353  0000c8 1ff3                         adc ZH, r19
```

```
354  0000c9 9124                          lpm r18, Z                  ;load byte from table pointed to by Z
355  0000ca 9508                         ret
356
357                                      ;Table of segment values to display digits 0 - F
358                                      ;!!! seven values must be added - verify all values
359  0000cb 4f01
360  0000cc 0612
361  0000cd 244c
362  0000ce 0f20
363  0000cf 0400
364  0000d0 6008
365  0000d1 4231
366  0000d2 3830                     hextable: .db $01, $4F, $12, $06, $4C, $24, $20, $0F, $00, $04, $08, $60, $31, $42, $30, $38
367
368
369  RESOURCE USE INFORMATION
370  ------------------------
371
372  Notice:
373  The register and instruction counts are symbol table hit counts,
374  and hence implicitly used resources are not counted, eg, the
375  'lpm' instruction without operands implicitly uses r0 and z,
376  none of which are counted.
377
378  x,y,z are separate entities in the symbol table and are
379  counted separately from r26..r31 here.
380
381  .dseg memory usage only counts static data declared with .byte
382
383  "ATmega4809" register use summary:
384  x  :   4 y  :   1 z  :   1 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
385  r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   0
386  r13:   0 r14:   0 r15:   0 r16:  64 r17:  15 r18:  21 r19:   4 r20:   3
387  r21:   5 r22:   0 r23:   2 r24:   2 r25:   0 r26:   3 r27:   3 r28:   2
388  r29:   1 r30:   4 r31:   4
389  Registers used: 17 out of 35 (48.6%)
390
391  "ATmega4809" instruction use summary:
392  .lds  :   0 .sts  :   0 adc   :   1 add   :   2 adiw  :   0 and   :   0
393  andi  :   6 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
394  brcc  :   0 brcs  :   0 break :   0 breq  :   0 brge  :   0 brhc  :   0
395  brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
396  brne  :   3 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
397  brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :   1 cbr   :   0
398  clc   :   0 clh   :   0 cli   :   0 cln   :   0 clr   :   0 cls   :   0
```

```
399  clt   :   0 clv   :   0 clz   :   0 com   :   2 cp    :   0 cpc   :   0
400  cpi   :   0 cpse  :   0 dec   :   3 des   :   0 eor   :   1 fmul  :   0
401  fmuls :   0 fmulsu:   0 icall :   0 ijmp  :   0 in    :   3 inc   :   2
402  jmp   :   5 ld    :   1 ldd   :   0 ldi   :  24 lds   :   3 lpm   :   2
403  lsl   :   1 lsr   :   1 mov   :  13 movw  :   0 mul   :   0 muls  :   0
404  mulsu :   0 neg   :   0 nop   :   1 or    :   0 ori   :   0 out   :   9
405  pop   :  12 push  :  12 rcall :   7 ret   :   4 reti  :   1 rjmp  :   2
406  rol   :   0 ror   :   0 sbc   :   0 sbci  :   0 sbi   :   0 sbic  :   0
407  sbis  :   0 sbiw  :   0 sbr   :   0 sbrc  :   0 sbrs  :   0 sec   :   0
408  seh   :   0 sei   :   2 sen   :   0 ser   :   0 ses   :   0 set   :   0
409  sev   :   0 sez   :   0 sleep :   0 spm   :   0 st    :   4 std   :   0
410  sts   :  13 sub   :   0 subi  :   0 swap  :   2 tst   :   0 wdr   :   0
411
412  Instructions used: 30 out of 114 (26.3%)
413
414  "ATmega4809" memory use summary [bytes]:
415  Segment   Begin    End      Code   Data   Used    Size   Use%
416  ----------------------------------------------------------------
417  [.cseg] 0x000000 0x0001a6    326     16    342   49152   0.7%
418  [.dseg] 0x002800 0x002805      0      5      5    6144   0.1%
419  [.eseg] 0x000000 0x000000      0      0      0     256   0.0%
420
421  Assembly complete, 0 errors, 0 warnings
422
```

```
 1
 2  AVRASM ver. 2.2.7  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas
      \temp_meas\main.asm Tue Nov 17 19:25:58 2020
 3
 4  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel
      \ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
 5  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (439): warning: Register r16 already defined by the .DEF directive
 6  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (440): warning: Register r17 already defined by the .DEF directive
 7  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (485): warning: Register r16 already defined by the .DEF directive
 8  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (486): warning: Register r17 already defined by the .DEF directive
 9  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (487): warning: Register r18 already defined by the .DEF directive
10  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (488): warning: Register r19 already defined by the .DEF directive
11  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (489): warning: Register r18 already defined by the .DEF directive
12  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (490): warning: Register r19 already defined by the .DEF directive
13  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (491): warning: Register r20 already defined by the .DEF directive
14  E:\ESE_280\$MyDocuments$\Atmel Studio\7.0\lab_11\temp_meas\temp_meas\main.asm
      (9): Including file 'C:/Program Files (x86)\Atmel\Studio\7.0\Packs\atmel
      \ATmega_DFP\1.3.300\avrasm\inc\m4809def.inc'
15
16
17                                    ; temp_meas.asm
18                                    ;
19                                    ; Created: 11/17/2020 2:02:20 PM
20                                    ; Author : Judah Ben-Eliezer
21                                    ;
22
23                                    .list
24
25                                    .equ PERIOD_EXAMPLE_VALUE = 25
26
27                                    .dseg
28  002800                            bcd_entries: .byte 4
29  002804                            led_display: .byte 4
30  002808                            digit_num: .byte 1
31
32
33                                    .cseg
34
35                                    reset:
```

```
 36  000000 940c 002e                    jmp start
 37
 38                                  .org TCA0_OVF_vect
 39  00000e 940c 0056                    jmp post_display_ISR
 40
 41                                  .org ADC0_RESRDY_vect
 42  00002c 940c 006a                    jmp read_ISR
 43
 44                                  start:
 45                                      ;configure inputs and outputs
 46  00002e 9883                         cbi VPORTE_DIR, 3
 47  00002f ef0f                         ldi r16, $FF
 48  000030 b908                         out VPORTC_DIR, r16
 49  000031 b90c                         out VPORTD_DIR, r16
 50  000032 9500                         com r16
 51  000033 b909                         out VPORTC_OUT, r16
 52  000034 b90d                         out VPORTD_OUT, r16
 53
 54                                      ;configure TCA0
 55  000035 e000                         ldi r16, TCA_SINGLE_WGMODE_NORMAL_gc        ↵
       ;WGMODE normal
 56  000036 9300 0a01                    sts TCA0_SINGLE_CTRLB, r16
 57
 58                                      ;enable overflow interrupt
 59  000038 e001                         ldi r16, TCA_SINGLE_OVF_bm
 60  000039 9300 0a0a                    sts TCA0_SINGLE_INTCTRL, r16
 61
 62                                      ;load period low byte then high byte
 63  00003b e109                         ldi r16, LOW(PERIOD_EXAMPLE_VALUE)
 64  00003c 9300 0a26                    sts TCA0_SINGLE_PER, r16
 65  00003e e000                         ldi r16, HIGH(PERIOD_EXAMPLE_VALUE)
 66  00003f 9300 0a27                    sts TCA0_SINGLE_PER + 1, r16
 67
 68                                      ;set clock and start timer
 69  000041 e00d                         ldi r16, TCA_SINGLE_CLKSEL_DIV256_gc |      ↵
       TCA_SINGLE_ENABLE_bm
 70  000042 9300 0a00                    sts TCA0_SINGLE_CTRLA, r16
 71
 72                                      ;set voltage reference
 73  000044 e200                         ldi r16, VREF_ADC0REFSEL_2V5_gc
 74  000045 9300 00a0                    sts VREF_CTRLA, r16
 75
 76                                      ;select PE1/ AIN9
 77  000047 e00b                         ldi r16, ADC_MUXPOS_AIN11_gc
 78  000048 9300 0606                    sts ADC0_MUXPOS, r16
 79
 80                                      ;enable internal reference and set          ↵
                       prescaler to div 64
 81  00004a e005                         ldi r16, ADC_PRESC_DIV64_gc |               ↵
```

```
      ADC_REFSEL_INTREF_gc
 82  00004b 9300 0602                   sts ADC0_CTRLC, r16
 83
 84                                      ;set resolution to 10 bit and enable adc
 85  00004d e001                        ldi r16, ADC_RESSEL_10BIT_gc |
      ADC_ENABLE_bm;
 86  00004e 9300 0600                   sts ADC0_CTRLA, r16
 87
 88                                      ;start conversion
 89  000050 e001                        ldi r16, ADC_STCONV_bm;
 90  000051 9300 0608                   sts ADC0_COMMAND, r16
 91
 92                                      ;enable interrupts
 93  000053 9478                        sei
 94  000054 940c 0065                   jmp wait_for_post
 95
 96                                  ;*****************************************
                     ******************************
 97                                  ;*
 98                                  ;* "post_display" - title
 99                                  ;*
100                                  ;* Description: toggles value for all PORTC
                     pins.  Since PORTC is used to multiplex the led display,
                     this will
101                                  ;* turn the LED display on and off
102                                  ;* Author:  Judah Ben-Eliezer
103                                  ;* Version:    1.0
104                                  ;* Last updated:   11/17
105                                  ;* Target:  ATmega4809
106                                  ;* Number of words:    13
107                                  ;* Number of cycles:   6
108                                  ;* Low registers modified:
109                                  ;* High registers modified:
110                                  ;* Parameters:  none
111                                  ;* Returns:        none
112                                  ;*
113                                  ;* Notes:
114                                  ;*
115                                  ;*****************************************
                     ******************************
116                                 post_display_ISR:
117  000056 930f                        push r16
118  000057 b70f                        in r16, CPU_SREG
119  000058 930f                        push r16
120  000059 931f                        push r17
121
122  00005a ef1f                        ldi r17, $FF
123  00005b b109                        in r16, VPORTC_OUT
124  00005c 2701                        eor r16, r17
```

```
125  00005d b909                          out VPORTC_OUT, r16
126
127                                       ;ldi r16, TCA_SINGLE_OVF_bm ;clear OVF flag
128                                       ;sts TCA0_SINGLE_INTFLAGS, r16
129
130  00005e 911f                          pop r17
131  00005f 910f                          pop r16
132  000060 bf0f                          out CPU_SREG, r16
133  000061 910f                          pop r16
134
135  000062 9478                          sei
136  000063 940c 0067                     jmp main_loop
137
138                                    wait_for_post:
139  000065 0000                          nop
140  000066 cffe                          rjmp wait_for_post
141
142                                    main_loop:
143  000067 d037                          rcall multiplex_display
144  000068 d04b                          rcall mux_digit_delay
145  000069 cffd                          rjmp main_loop
146
147                                    ;*********************************************
                                    ****************************
148                                    ;*
149                                    ;* "read_ISR" - title
150                                    ;*
151                                    ;* Description: loads ADC0_RES into r17:r16
                                    and calls bin16_to_BCD
152                                    ;*
153                                    ;* Author:  Judah Ben-Eliezer
154                                    ;* Version:     1.0
155                                    ;* Last updated:    11/17/2020
156                                    ;* Target:  ATmega4809
157                                    ;* Number of words:
158                                    ;* Number of cycles:
159                                    ;* Low registers modified:  none
160                                    ;* High registers modified:         r17:r16
161                                    ;*
162                                    ;* Parameters:  ADC0_RES
163                                    ;* Returns:         r17:r16
164                                    ;*
165                                    ;* Notes:
166                                    ;*
167                                    ;*********************************************
                                    ****************************
168                                    read_ISR:
169  00006a 930f                          push r16
170  00006b b70f                          in r16, CPU_SREG
```

```
171  00006c 930f                        push r16
172  00006d 2f0b                        mov r16, XH
173  00006e 930f                        push r16
174  00006f 2f0a                        mov r16, XL
175  000070 930f                        push r16
176  000071 2f0f                        mov r16, ZH
177  000072 930f                        push r16
178  000073 2f0e                        mov r16, ZL
179  000074 930f                        push r16
180  000075 931f                        push r17
181  000076 932f                        push r18
182  000077 933f                        push r19
183
184  000078 e039                        ldi r19, $09    ;load multiplier 2500 into ↵
       r19:r18 for multiplication
185  000079 ec24                        ldi r18, $C4
186  00007a 9110 0611                   lds r17, ADC0_RESH
187  00007c 9100 0610                   lds r16, ADC0_RESL
188  00007e d07f                        rcall mpy16u
189  00007f 9526                        lsr r18     ;finish division by 1024 by     ↵
       taking middle two bytes and shifting right twice
190  000080 9517                        ror r17
191  000081 9526                        lsr r18
192  000082 9517                        ror r17
193  000083 2f01                        mov r16, r17    ; move output to r17:r16
194  000084 2f12                        mov r17, r18
195  000085 5f04                        subi r16, $F4   ;subtract 500 from result
196  000086 4011                        sbci r17, $01
197  000087 d054                        rcall bin16_to_BCD
198  000088 d032                        rcall packed_to_bcd_entries
199  000089 d041                        rcall bcd_to_led
200
201                                     ;reset interrupt flag
202  00008a e001                        ldi r16, ADC_RESRDY_bm;
203  00008b 9300 060a                   sts ADC0_INTCTRL, r16
204
205                                     ;restart conversion
206  00008d e001                        ldi r16, ADC_STCONV_bm;
207  00008e 9300 0608                   sts ADC0_COMMAND, r16
208
209  000090 913f                        pop r19
210  000091 912f                        pop r18
211  000092 911f                        pop r17
212  000093 910f                        pop r16
213  000094 2fe0                        mov ZL, r16
214  000095 910f                        pop r16
215  000096 2ff0                        mov ZH, r16
216  000097 910f                        pop r16
217  000098 2fa0                        mov XL, r16
```

```
218  000099 910f                        pop r16
219  00009a 2fb0                        mov XH, r16
220  00009b 910f                        pop r16
221  00009c bf0f                        out CPU_SREG, r16
222  00009d 910f                        pop r16
223
224  00009e 9518                        reti
225
226                                     ;************************************************
                                    ****************************
227                                     ;*
228                                     ;* "multiplex_display" - title
229                                     ;*
230                                     ;* Description:    outputs values from
                                    led_display array to 7 segment display on PORTD driven by
                                    highest two bits of PORTC
231                                     ;*
232                                     ;* Author: Judah Ben-Eliezer
233                                     ;* Version:    1.0
234                                     ;* Last updated:   11/10/2020
235                                     ;* Target: ATmega4809
236                                     ;* Number of words:
237                                     ;* Number of cycles:
238                                     ;* Low registers modified:
239                                     ;* High registers modified:
240                                     ;*
241                                     ;* Parameters:
242                                     ;* Returns:
243                                     ;*
244                                     ;* Notes:
245                                     ;*
246                                     ;************************************************
                                    ****************************
247                                     multiplex_display:
248  00009f e2d8                            ldi YH, HIGH(led_display)
249  0000a0 e0c4                            ldi YL, LOW(led_display)
250  0000a1 9110 2808                       lds r17, digit_num
251  0000a3 7013                            andi r17, $03
252  0000a4 2f41                            mov r20, r17
253  0000a5 0fc1                            add YL, r17
254  0000a6 8128                            ld r18, Y
255  0000a7 e850                            ldi r21, $80
256  0000a8 9543                            inc r20
257                                     loop:
258  0000a9 9556                            lsr r21
259  0000aa 954a                            dec r20
260  0000ab f7e9                            brne loop
261  0000ac 0f55                            lsl r21
262  0000ad 9550                            com r21
```

```
263  0000ae b959                          out VPORTC_OUT, r21
264  0000af b92d                          out VPORTD_OUT, r18
265  0000b0 9513                          inc r17
266  0000b1 9310 2808                     sts digit_num, r17
267  0000b3 9508                          ret
268
269                                       ;*****************************************
                         ******************************
270                                       ;*
271                                       ;* "mux_digit_delay" - title
272                                       ;*
273                                       ;* Description: delays 0.1 * r23
274                                       ;*
275                                       ;* Author:  Judah Ben-Eliezer
276                                       ;* Version:    1.0
277                                       ;* Last updated:
278                                       ;* Target:
279                                       ;* Number of words:
280                                       ;* Number of cycles:
281                                       ;* Low registers modified:
282                                       ;* High registers modified:
283                                       ;*
284                                       ;* Parameters:
285                                       ;* Returns:
286                                       ;*
287                                       ;* Notes:
288                                       ;*
289                                       ;*****************************************
                         ******************************
290                                       mux_digit_delay:
291  0000b4 e078                             ldi r23, $08 ; 0.1 * r23 = delay
292                                       outer_loop:
293  0000b5 e086                             ldi r24, $06
294                                       inner_loop:
295  0000b6 958a                             dec r24
296  0000b7 f7f1                             brne inner_loop
297  0000b8 957a                             dec r23
298  0000b9 f7d9                             brne outer_loop
299  0000ba 9508                             ret
300
301                                       ;*****************************************
                         ******************************
302                                       ;*
303                                       ;* "packed_to_bcd_entries" - title
304                                       ;*
305                                       ;* Description:    Converts bcd input to 7seg
                         output, from bcd_entries array to led_display array
306                                       ;*
307                                       ;* Author:
```

```
308                                 ;* Version:
309                                 ;* Last updated:
310                                 ;* Target:
311                                 ;* Number of words:
312                                 ;* Number of cycles:
313                                 ;* Low registers modified:
314                                 ;* High registers modified:
315                                 ;*
316                                 ;* Parameters:
317                                 ;* Returns:
318                                 ;*
319                                 ;* Notes:
320                                 ;*
321                                 ;*********************************************
                            ****************************
322                                 packed_to_bcd_entries:
323  0000bb e2b8                       ldi XH, HIGH(bcd_entries)
324  0000bc e0a0                       ldi XL, LOW(bcd_entries)
325  0000bd 2f37                       mov r19, r23
326  0000be 7f30                       andi r19, $F0
327  0000bf 9532                       swap r19
328  0000c0 933d                       st X+, r19
329  0000c1 2f37                       mov r19, r23
330  0000c2 703f                       andi r19, $0F
331  0000c3 933d                       st X+, r19
332  0000c4 2f36                       mov r19, r22
333  0000c5 7f30                       andi r19, $F0
334  0000c6 9532                       swap r19
335  0000c7 933d                       st X+, r19
336  0000c8 2f36                       mov r19, r22
337  0000c9 703f                       andi r19, $0F
338  0000ca 933c                       st X, r19
339
340
341                                 ;*********************************************
                            ****************************
342                                 ;*
343                                 ;* "bcd_to_led" - title
344                                 ;*
345                                 ;* Description:    Converts bcd input to 7seg
                            output, from bcd_entries array to led_display array
346                                 ;*
347                                 ;* Author:
348                                 ;* Version:
349                                 ;* Last updated:
350                                 ;* Target:
351                                 ;* Number of words:
352                                 ;* Number of cycles:
353                                 ;* Low registers modified:
```

```
354                                 ;* High registers modified:
355                                 ;*
356                                 ;* Parameters:
357                                 ;* Returns:
358                                 ;*
359                                 ;* Notes:
360                                 ;*
361                                 ;*****************************************
                            *****************************
362
363                                 bcd_to_led:
364  0000cb e2b8                       ldi XH, HIGH(bcd_entries)
365  0000cc e0a0                       ldi XL, LOW(bcd_entries)
366  0000cd 931c                       st X, r17
367  0000ce e044                       ldi r20, $04
368                                 conversion_loop:
369  0000cf 954a                       dec r20
370  0000d0 e2b8                       ldi XH, HIGH(bcd_entries)
371  0000d1 e0a0                       ldi XL, LOW(bcd_entries)
372  0000d2 e2d8                       ldi YH, HIGH(led_display)
373  0000d3 e0c4                       ldi YL, LOW(led_display)
374  0000d4 0fa4                       add XL, r20
375  0000d5 0fc4                       add YL, r20
376  0000d6 912c                       ld r18, X
377  0000d7 d035                       rcall hex_to_7seg
378  0000d8 8328                       st Y, r18
379  0000d9 3040                       cpi r20, $00
380  0000da f7a1                       brne conversion_loop
381  0000db 9508                       ret
382
383                                 ;*****************************************
                            *****************************
384                                 ;*
385                                 ;* "bin16_to_BCD" - 16-bit Binary to BCD
                            Conversion
386                                 ;*
387                                 ;* Description: Converts a 16-bit unsigned
                            binary number to a five digit
388                                 ;* packed BCD number. Uses subroutine div16u
                            from Atmel application note AVR200
389                                 ;*
390                                 ;* Author:                 Ken Short
391                                 ;* Version:                    0.0
392                                 ;* Last updated:           111320
393                                 ;* Target:                 ATmega4809
394                                 ;* Number of words:
395                                 ;* Number of cycles:
396                                 ;* Low registers modified:  r14, r15
397                                 ;* High registers modified: r16, r17, r18,
```

```
                        r19, r20, r22, r23, r24
398                                 ;*
399                                 ;* Parameters: r17:r16 16-bit unsigned right
                        justified number to be converted.
400                                 ;* Returns:        r24:r23:r22 five digit
                        packed BCD result.
401                                 ;*
402                                 ;* Notes:
403                                 ;* Subroutine uses repeated division by 10 to
                        perform conversion.
404                                 ;*****************************************
                        ****************************
405                                 bin16_to_BCD:
406 0000dc e030                        ldi r19, 0          ;high byte of divisor
      for div16u
407 0000dd e02a                        ldi r18, 10         ;low byte of the
      divisor for div16u
408
409 0000de d00c                        rcall div16u        ;divide original binary
       number by 10
410 0000df 2d6e                        mov r22, r14        ;result is BCD digit 0
      (least significant digit)
411 0000e0 d00a                        rcall div16u        ;divide result from
      first division by 10, gives digit 1
412 0000e1 94e2                        swap r14            ;swap digit 1 for
      packing
413 0000e2 296e                        or r22, r14         ;pack
414
415 0000e3 d007                        rcall div16u        ;divide result from
      second division by 10, gives digit 2
416 0000e4 2d7e                        mov r23, r14        ;place in r23
417 0000e5 d005                        rcall div16u        ;divide result from
      third division by 10, gives digit 3
418 0000e6 94e2                        swap r14            ;swap digit 3 for
      packing
419 0000e7 297e                        or r23, r14         ;pack
420
421 0000e8 d002                        rcall div16u        ;divide result from
      fourth division by 10, gives digit 4
422 0000e9 2d8e                        mov r24, r14        ;place in r24
423
424 0000ea 9508                        ret
425
426
427                                 ;Subroutine div16u is from Atmel application
                        note AVR200
428
429                                 ;*****************************************
                        ****************************
```

```
430                                 ;*
431                                 ;* "div16u" - 16/16 Bit Unsigned Division
432                                 ;*
433                                 ;* This subroutine divides the two 16-bit numbers
434                                 ;*# "dd16uH:dd16uL" (dividend) and "dv16uH:dv16uL" (divisor).
435                                 ;* The result is placed in "dres16uH:dres16uL" and the remainder in
436                                 ;* "drem16uH:drem16uL".
437                                 ;*
438                                 ;* Number of words  :19
439                                 ;* Number of cycles    :235/251 (Min/Max)
440                                 ;* Low registers used  :2 (drem16uL,drem16uH)
441                                 ;* High registers used  :5 (dres16uL/dd16uL,dres16uH/dd16uH,dv16uL,dv16uH,
442                                 ;*             dcnt16u)
443                                 ;*
444                                 ;*********************************************************************
445
446                                 ;***** Subroutine Register Variables
447
448                                 .def   drem16uL=r14
449                                 .def   drem16uH=r15
450                                 .def   dres16uL=r16
451                                 .def   dres16uH=r17
452                                 .def   dd16uL  =r16
453                                 .def   dd16uH  =r17
454                                 .def   dv16uL  =r18
455                                 .def   dv16uH  =r19
456                                 .def   dcnt16u =r20
457
458                                 ;***** Code
459
460 0000eb 24ee                    div16u:    clr drem16uL    ;clear remainder Low byte
461 0000ec 18ff                       sub drem16uH,drem16uH;clear remainder High byte and carry
462 0000ed e141                       ldi dcnt16u,17  ;init loop counter
463 0000ee 1f00                    d16u_1:    rol dd16uL      ;shift left dividend
464 0000ef 1f11                       rol dd16uH
465 0000f0 954a                       dec dcnt16u     ;decrement counter
466 0000f1 f409                       brne    d16u_2      ;if done
467 0000f2 9508                       ret         ;    return
468 0000f3 1cee                    d16u_2:    rol drem16uL    ;shift dividend into remainder
469 0000f4 1cff                       rol drem16uH
```

```
470  0000f5 1ae2                sub drem16uL,dv16uL ;remainder = remainder - divisor
471  0000f6 0af3                sbc drem16uH,dv16uH ;
472  0000f7 f420                brcc    d16u_3      ;if result negative
473  0000f8 0ee2                add drem16uL,dv16uL ;    restore remainder
474  0000f9 1ef3                adc drem16uH,dv16uH
475  0000fa 9488                clc         ;    clear carry to be shifted into result
476  0000fb cff2                rjmp    d16u_1      ;else
477  0000fc 9408               d16u_3:     sec         ;    set carry to be shifted into result
478  0000fd cff0                rjmp    d16u_1
479
480                             ;***************************************************************************
481                             ;*
482                             ;* "mpy16u" - 16x16 Bit Unsigned Multiplication
483                             ;*
484                             ;* This subroutine multiplies the two 16-bit register variables
485                             ;* mp16uH:mp16uL and mc16uH:mc16uL.
486                             ;* The result is placed in m16u3:m16u2:m16u1:m16u0.
487                             ;*
488                             ;* Number of words  :14 + return
489                             ;* Number of cycles     :153 + return
490                             ;* Low registers used   :None
491                             ;* High registers used  :7 (mp16uL,mp16uH,mc16uL/m16u0,mc16uH/m16u1,m16u2,
492                             ;*                          m16u3,mcnt16u)
493                             ;*
494                             ;***************************************************************************
495
496                             ;***** Subroutine Register Variables
497
498                             .def    mc16uL  =r16        ;multiplicand low byte
499                             .def    mc16uH  =r17        ;multiplicand high byte
500                             .def    mp16uL  =r18        ;multiplier low byte
501                             .def    mp16uH  =r19        ;multiplier high byte
502                             .def    m16u0   =r18        ;result byte 0 (LSB)
503                             .def    m16u1   =r19        ;result byte 1
504                             .def    m16u2   =r20        ;result byte 2
```

```
505                                 .def    m16u3   =r21        ;result byte 3 (MSB)
506                                 .def    mcnt16u =r22        ;loop counter
507
508                                 ;***** Code
509
510 0000fe 2755                     mpy16u:     clr m16u3       ;clear 2 highest bytes of result
511 0000ff 2744                        clr m16u2
512 000100 e160                        ldi mcnt16u,16  ;init loop counter
513 000101 9536                        lsr mp16uH
514 000102 9527                        ror mp16uL
515
516 000103 f410                     m16u_1:     brcc    noad8       ;if bit 0 of multiplier set
517 000104 0f40                        add m16u2,mc16uL    ;add multiplicand Low to byte 2 of res
518 000105 1f51                        adc m16u3,mc16uH    ;add multiplicand high to byte 3 of res
519 000106 9557                     noad8: ror m16u3       ;shift right result byte 3
520 000107 9547                        ror m16u2       ;rotate right result byte 2
521 000108 9537                        ror m16u1       ;rotate result byte 1 and multiplier High
522 000109 9527                        ror m16u0       ;rotate result byte 0 and multiplier Low
523 00010a 956a                        dec mcnt16u     ;decrement loop counter
524 00010b f7b9                        brne    m16u_1      ;if not done, loop more
525 00010c 9508                        ret
526
527
528
529
530                                 ;***********************************************************************
531                                 ;*
532                                 ;* "hex_to_7seg" - Hexadecimal to Seven Segment Conversion
533                                 ;*
534                                 ;* Description: Converts a right justified hexadecimal digit to the seven
535                                 ;* segment pattern required to display it. Pattern is right justified a
536                                 ;* through g. Pattern uses 0s to turn segments on ON.
537                                 ;*
538                                 ;* Author:                      Ken Short
539                                 ;* Version:                         1.0
```

```
540                                 ;* Last updated:               101620
541                                 ;* Target:                     ATmega4809
542                                 ;* Number of words:                8
543                                 ;* Number of cycles:           13
544                                 ;* Low registers modified:     none
545                                 ;* High registers modified:        r19, r18,
                        ZL, ZH
546                                 ;*
547                                 ;* Parameters: r18: right justified hex digit,
                         high nibble 0
548                                 ;* Returns: r18: segment values a through g
                        right justified
549                                 ;*
550                                 ;* Notes:
551                                 ;*
552                                 ;*****************************************
                        ****************************
553
554                                 hex_to_7seg:
555 00010d 702f                       andi r18, 0x0F              ;clear ms
      nibble
556 00010e e0f2                        ldi ZH, HIGH(hextable * 2)  ;set Z to
      point to start of table
557 00010f e2ea                        ldi ZL, LOW(hextable * 2)
558 000110 e030                        ldi r19, $00                ;add offset to
       Z pointer
559 000111 0fe2                        add ZL, r18
560 000112 1ff3                        adc ZH, r19
561 000113 9124                        lpm r18, Z                  ;load byte
      from table pointed to by Z
562 000114 9508                       ret
563
564                                    ;Table of segment values to display digits
                         0 - F
565                                    ;!!! seven values must be added - verify
                        all values
566 000115 4f01
567 000116 0612
568 000117 244c
569 000118 0f20
570 000119 0400
571 00011a 6008
572 00011b 4231
573 00011c 3830                     hextable: .db $01, $4F, $12, $06, $4C, $24,
      $20, $0F, $00, $04, $08, $60, $31, $42, $30, $38
574
575
576 RESOURCE USE INFORMATION
577 ------------------------
```

```
578
579 Notice:
580 The register and instruction counts are symbol table hit counts,
581 and hence implicitly used resources are not counted, eg, the
582 'lpm' instruction without operands implicitly uses r0 and z,
583 none of which are counted.
584
585 x,y,z are separate entities in the symbol table and are
586 counted separately from r26..r31 here.
587
588 .dseg memory usage only counts static data declared with .byte
589
590 "ATmega4809" register use summary:
591 x  :   6 y  :   2 z  :   1 r0 :   0 r1 :   0 r2 :   0 r3 :   0 r4 :   0
592 r5 :   0 r6 :   0 r7 :   0 r8 :   0 r9 :   0 r10:   0 r11:   0 r12:   0
593 r13:   0 r14:  11 r15:   5 r16:  66 r17:  21 r18:  18 r19:  24 r20:  13
594 r21:   8 r22:   6 r23:   6 r24:   3 r25:   0 r26:   6 r27:   5 r28:   4
595 r29:   2 r30:   4 r31:   4
596 Registers used: 20 out of 35 (57.1%)
597
598 "ATmega4809" instruction use summary:
599 .lds  :   0 .sts  :   0 adc   :   3 add   :   6 adiw  :   0 and   :   0
600 andi  :   6 asr   :   0 bclr  :   0 bld   :   0 brbc  :   0 brbs  :   0
601 brcc  :   2 brcs  :   0 break :   0 breq  :   0 brge  :   0 brhc  :   0
602 brhs  :   0 brid  :   0 brie  :   0 brlo  :   0 brlt  :   0 brmi  :   0
603 brne  :   6 brpl  :   0 brsh  :   0 brtc  :   0 brts  :   0 brvc  :   0
604 brvs  :   0 bset  :   0 bst   :   0 call  :   0 cbi   :   1 cbr   :   0
605 clc   :   1 clh   :   0 cli   :   0 cln   :   0 clr   :   3 cls   :   0
606 clt   :   0 clv   :   0 clz   :   0 com   :   2 cp    :   0 cpc   :   0
607 cpi   :   1 cpse  :   0 dec   :   6 des   :   0 eor   :   1 fmul  :   0
608 fmuls :   0 fmulsu:   0 icall :   0 ijmp  :   0 in    :   3 inc   :   2
609 jmp   :   5 ld    :   2 ldd   :   0 ldi   :  37 lds   :   3 lpm   :   2
610 lsl   :   1 lsr   :   4 mov   :  18 movw  :   0 mul   :   0 muls  :   0
611 mulsu :   0 neg   :   0 nop   :   1 or    :   2 ori   :   0 out   :   9
612 pop   :  12 push  :  12 rcall :  12 ret   :   7 reti  :   1 rjmp  :   4
613 rol   :   4 ror   :   7 sbc   :   1 sbci  :   1 sbi   :   0 sbic  :   0
614 sbis  :   0 sbiw  :   0 sbr   :   0 sbrc  :   0 sbrs  :   0 sec   :   1
615 seh   :   0 sei   :   2 sen   :   0 ser   :   0 ses   :   0 set   :   0
616 sev   :   0 sez   :   0 sleep :   0 spm   :   0 st    :   6 std   :   0
617 sts   :  13 sub   :   2 subi  :   1 swap  :   4 tst   :   0 wdr   :   0
618
619 Instructions used: 42 out of 114 (36.8%)
620
621 "ATmega4809" memory use summary [bytes]:
622 Segment   Begin    End      Code   Data   Used    Size   Use%
623 ----------------------------------------------------------------
624 [.cseg] 0x000000 0x00023a    474     16    490   49152   1.0%
625 [.dseg] 0x002800 0x002809      0      9      9    6144   0.1%
626 [.eseg] 0x000000 0x000000      0      0      0     256   0.0%
```

```
627
628  Assembly complete, 0 errors, 9 warnings
629
```

Verification Strategy:

For part 1:

Set a breakpoint at the first instruction of the main_loop to make sure the post_display subroutine executes before it. Use the IO window to check the TCA0_INTFLAGS overflow flag to make sure it is set when the subroutine is called. The counter is set to approximately 1 second, so its operation should be easily visible on the 7seg display. Verify with the oscilloscope that the timing is correct

For part 2:

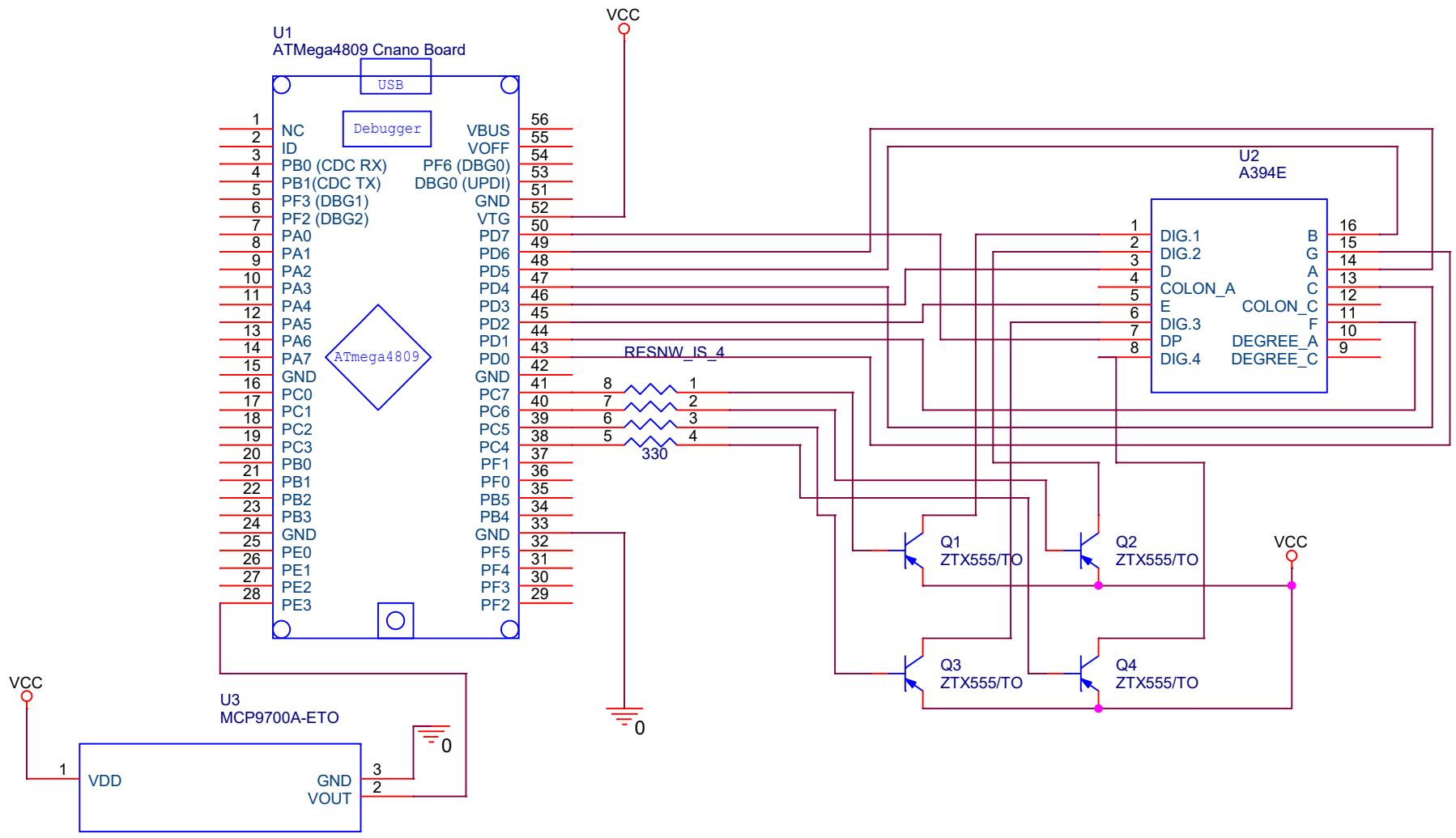
Set the trimpot to zero and verify that the 7seg display is showing zero. Then slowly increase the trimpot in 0.5V increments up to 3.0V. Verify with a calculator that the conversion is correct for each increment. Ensure that the trimpot outputs the desired voltage by connecting its output to the multimeter or the oscilloscope.

For part 3:

Start out by measuring the ambient room temperature and use a calculator to convert from hexadecimal. Make sure that the calculated temperature makes sense. Heat the thermometer up with your hand and record the temperature. Cool it down with a cold can and record the temperature again. Convert all the values to decimal and then to Celsius. Make sure they make sense.

For part 4:

Similar to part 3, measure the temperature at room temperature, body temperature, and cold can temperature. Make sure the temperature measurements are correct.



Title	temp_meas by Judah Ben-Eliezer	
Size A	Document Number <Doc>	Rev <RevCo
Date: Tuesday, October 20, 2020	Sheet 1 of 1	