ECS Assignment Report

By

Shubham Kumar

18IE10021

Course: Instrumentation Engineering

Department: Electrical Engineering

Undergraduate student at IIT Kgp

3rd semester, Autumn sem

Assignment Problem : Write programs in assembly and C to simultaneously generate different frequencies of square waveforms (10n Hz; n=0, 1, 2 ,3, 4, 5,6,7,...) (that is, 1 Hz, 10 Hz, 100 Hz, 1000Hz, 10000 Hz, ...) at different pins of PORT C output using the timers 0 , 1 and 2. Identify the limitations/extent of capability of using all of these timers for generating pulses. That is, up to how many different frequencies can you generate simultaneously using these timers and justify, why?. The generated waveforms can have shifts in phase/delay among each other by a maximum of 0.1 seconds. State the assumptions, if any.

Solution:

1. Clearly, there is limit on the number of frequencies we can generate using 8MHz clock. The smallest time delay possible is due to one clock pulse i.e. 0.125μs. Therefore, theoretically we can generate any

time delay greater than 0.125µs. So, only feasible frequency seems to be 1 Hz, 10 Hz, 100 Hz, 1000 Hz, … 1000000 Hz.

2. But we have to generate time delays using timers 0, 1, 2 , so there will be several intermediate instruction which will interfere with delay, especially when delay to be generated is small, they are source of significant errors.
   For eg.:

```
.include "m32def.inc"
ldi r16, -1
ldi r17, 1
ldi r18, 0
;delay starts here
out tcnt0, r16
out tccr0, r17
again : in r20, tifr
        sbrs r20, tov0
        rjmp again
out tccr0, r18
out tifr, r17
       end : rjmp end
```

   Takes 11 clock cycle for minimum possible delay using timer 0, which is 1.375µs, even if tccr0 is loaded with -1, one tick.

   The example removes any possibility of 1000000 Hz.

Assumption: While generating waves 10% tolerance in time delays is assumed.

3. It turns out that we can generate only 6 possible frequencies, those are 1 Hz to 100kHz.

Explanation of code logic:

- Logic of code is simple. In both C and assembly same logic is used.
- We generate delay for maximum possible frequency (minimum possible time period).
- Then using nested loop for each wave to be generated is used, where high frequency waves lies in.
- This logic is used considering the fact that frequencies are differing by a constant multiple of 10.

Implementation Details: All clock are implemented in Normal mode, no prescaler.

Calculation and Optimization:

- Theoretical calculations: For generating 100kHz, we need to have 80 clock ticks, considering 8MHz clock. So, we need to set TCNT register with -80(in binary ofcourse).
- Optimization calculations: Extra instructions in programs create significant error over i.e. 10%. To make the error below 10%, we feed -65 instead of -80, to compensate for extra instruction delays.
- We have not created separated delay subroutine, and have directly inlined in the code. For 2 overlapping reasons,
  - Only one time we need to call it.
  - Inlining reduces the use of call and ret instructions and make delay more precise.

  Also, we have used several registers to hold frequently used constant in assembly code. Reason behind is reduce the frequent loading of registers with constants and then use it, as it creates unwanted delays.

Assembly and C Differences :

- It has been noted using disassembly tool that my Assembly code generates more precise delays.
- The difference is more significant for timer 1 than those of timer 0 and 2 as timer 1 is 16 bit.

Code along with instruction's clock cycles are attached in the following pages :

Clock cycles are provided for simpler analysis, rather than actual time.
To find actual time, Total no. of clock cycles can always be multiplied with 0.125us.

Further code verification is possible with simulator, in ATMEL Studio.

For C code analysis Diassembly in simulator and .lss file was used.

MOVE TO NEXT PAGE,
THIS PAGE IS BLANK

TIMER 0 .asm

C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer0\timer0\main.asm 1

```
1  .include "m32def.inc"                                                    ;clock cycles
2  ldi r31, high(ramend)                                                    ; 1
3  out sph, r31                                                             ; 1
4  ldi r31, low(ramend)                                                     ; 1
5  out spl, r31                                                             ; 1
6
7  ldi r16, -65                                                             ; 1
8  ldi r18, 0                                                               ; 1
9  ldi r17, 1<<0                                                            ; 1
10 ldi r19, 1<<1                                                            ; 1
11 ldi r20, 1<<2                                                            ; 1
12 ldi r21, 1<<3                                                            ; 1
13 ldi r22, 1<<4                                                            ; 1
14 ldi r23, 1<<5                                                            ; 1
15
16 ldi r31, 0xFF                                                            ; 1
17 out ddrc, r31                                                            ; 1
18
19 start : ldi r24, 10                                                      ; 1
20         loop1 : ldi r25, 10                                              ; 1
21                 loop2 : ldi r26, 10                                      ; 1
22                         loop3 : ldi r27, 10                              ; 1
23                                 loop4 : ldi r28, 10                      ; 1
24                                         loop5 :     out tcnt0, r16       ; 1
25                                                     out tccr0, r17       ; 1
26                                                     again : in r20, tifr ; 1
27                                                             sbrs r20, tov0 ; 1 / 2
28                                                             rjmp again   ; 2
29                                                     out tccr0, r18       ; 1
30                                                     out tifr, r17        ; 1
31                                             in r31, portc                ; 1
32                                             eor r31, r17                 ; 1
33                                             out portc, r31               ; 1
34                                             dec r28                      ; 1
35                                             brne loop5                   ; 1 / 2
```

```
36                                      in r31, portc                   ; 1
37                                      eor r31, r19                    ; 1
38                                      out portc, r31                  ; 1
39                                      dec r27                         ; 1
40                                      brne loop4                      ; 1 / 2
41                                  in r31, portc                       ; 1
42                                  eor r31, r20                        ; 1
43                                  out portc, r31                      ; 1
44                                  dec r26                             ; 1
45                                  brne loop3                          ; 1 / 2
46                              in r31, portc                           ; 1
47                              eor r31, r21                            ; 1
48                              out portc, r31                          ; 1
49                              dec r25                                 ; 1
50                              brne loop2                              ; 1 / 2
51                          in r31, portc                               ; 1
52                          eor r31, r22                                ; 1
53                          out portc, r31                              ; 1
54                          dec r24                                     ; 1
55                          brne loop1                                  ; 1 / 2
56                  in r31, portc                                       ; 1
57                  eor r31, r23                                        ; 1
58                  out portc, r31                                      ; 1
59                  rjmp start                                          ; 2
60
61
```

TIMER 0 .C

C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer0_c\timer0_c\main.c 1

```c
#include "avr/io.h"

int main() {

    DDRC = 0xFF;

    while(1) {
         int a1 = 10;
            while(a1) {
                int a2 = 10;
                while(a2) {
                    int a3 = 10;
                    while(a3) {
                        int a4 = 10;
                        while(a4) {
                            int a5 = 10;
                            while(a5) {
                                    TCNT0 = -65;
                                    TCCR0 = 0x01;
                                    while(!(TIFR & (1<<TOV0)));
                                    TCCR0 = 0;
                                    TIFR = 0x01;
                                PORTC ^= 1<<0;
                                --a5;
                            }
                            PORTC ^= 1<<1;
                            --a4;
                        }
                        PORTC ^= 1<<2;
                        --a3;
                    }
                    PORTC ^= 1<<3;
                    --a2;
                }
                PORTC ^= 1<<4;
```

```
                --a1;
            }
            PORTC ^= 1<<5;
        }
    return 0;
}
```

.lss file for diassembly analysis of timer 0 .c code


```
timer0_c.elf:     file format elf32-avr

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000fc  00000000  00000000  00000054  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  00800060  00800060  00000150  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .comment      00000030  00000000  00000000  00000150  2**0
                  CONTENTS, READONLY
  3 .note.gnu.avr.deviceinfo 0000003c  00000000  00000000  00000180  2**2
                  CONTENTS, READONLY
  4 .debug_aranges 00000020  00000000  00000000  000001bc  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_info   000005d2  00000000  00000000  000001dc  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_abbrev 0000050d  00000000  00000000  000007ae  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   0000026c  00000000  00000000  00000cbb  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_frame  00000024  00000000  00000000  00000f28  2**2
                  CONTENTS, READONLY, DEBUGGING
  9 .debug_str    000002eb  00000000  00000000  00000f4c  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .debug_loc    000001e5  00000000  00000000  00001237  2**0
                  CONTENTS, READONLY, DEBUGGING
 11 .debug_ranges 000000d8  00000000  00000000  0000141c  2**0
                  CONTENTS, READONLY, DEBUGGING

Disassembly of section .text:

00000000 <__vectors>:
   0:   0c 94 2a 00     jmp 0x54    ; 0x54 <__ctors_end>
   4:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
```

```
   8:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
   c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  10:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  14:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  18:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  1c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  20:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  24:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  28:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  2c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  30:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  34:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  38:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  3c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  40:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  44:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  48:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  4c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  50:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>

00000054 <__ctors_end>:
  54:   11 24           eor r1, r1
  56:   1f be           out 0x3f, r1    ; 63
  58:   cf e5           ldi r28, 0x5F   ; 95
  5a:   d8 e0           ldi r29, 0x08   ; 8
  5c:   de bf           out 0x3e, r29   ; 62
  5e:   cd bf           out 0x3d, r28   ; 61
  60:   0e 94 36 00     call    0x6c    ; 0x6c <main>
  64:   0c 94 7c 00     jmp 0xf8    ; 0xf8 <_exit>

00000068 <__bad_interrupt>:
  68:   0c 94 00 00     jmp 0   ; 0x0 <__vectors>

0000006c <main>:
#include "avr/io.h"
```

```
int main() {

    DDRC = 0xFF;
  6c:   8f ef           ldi r24, 0xFF   ; 255
  6e:   84 bb           out 0x14, r24   ; 20
                        int a4 = 10;
                        while(a4) {
                            int a5 = 10;
                            while(a5) {
                                    TCNT0 = -65;
                                    TCCR0 = 0x01;
  70:   91 e0           ldi r25, 0x01   ; 1
                                    TCCR0 = 0;
                                    TIFR = 0x01;
                                PORTC ^= 1<<0;
                                --a5;
                            }
                            PORTC ^= 1<<1;
  72:   e2 e0           ldi r30, 0x02   ; 2
                    while(a3) {
                        int a4 = 10;
                        while(a4) {
                            int a5 = 10;
                            while(a5) {
                                    TCNT0 = -65;
  74:   5f eb           ldi r21, 0xBF   ; 191
                                    TCCR0 = 0x01;
  76:   49 2f           mov r20, r25
                                --a5;
                            }
                            PORTC ^= 1<<1;
                            --a4;
                        }
                        PORTC ^= 1<<2;
```

```
  78:   68 94           set
  7a:   dd 24           eor r13, r13
  7c:   d2 f8           bld r13, 2
                        --a3;
                    }
                    PORTC ^= 1<<3;
  7e:   68 94           set
  80:   ee 24           eor r14, r14
  82:   e3 f8           bld r14, 3
                    --a2;
                }
                PORTC ^= 1<<4;
  84:   68 94           set
  86:   ff 24           eor r15, r15
  88:   f4 f8           bld r15, 4
                --a1;
            }
            PORTC ^= 1<<5;
  8a:   f0 e2           ldi r31, 0x20   ; 32
#include "avr/io.h"

int main() {
  8c:   aa e0           ldi r26, 0x0A   ; 10
  8e:   b0 e0           ldi r27, 0x00   ; 0
  90:   2c c0           rjmp    .+88        ; 0xea <main+0x7e>
                    while(a3) {
                        int a4 = 10;
                        while(a4) {
                            int a5 = 10;
                            while(a5) {
                                    TCNT0 = -65;
  92:   52 bf           out 0x32, r21   ; 50
                                    TCCR0 = 0x01;
  94:   43 bf           out 0x33, r20   ; 51
                                    while(!(TIFR & (1<<TOV0)));
```

```
  96:   08 b6           in  r0, 0x38    ; 56
  98:   00 fe           sbrs    r0, 0
  9a:   fd cf           rjmp    .-6         ; 0x96 <main+0x2a>
                            TCCR0 = 0;
  9c:   13 be           out 0x33, r1    ; 51
                            TIFR = 0x01;
  9e:   98 bf           out 0x38, r25   ; 56
                        PORTC ^= 1<<0;
  a0:   85 b3           in  r24, 0x15   ; 21
  a2:   89 27           eor r24, r25
  a4:   85 bb           out 0x15, r24   ; 21
  a6:   21 50           subi    r18, 0x01   ; 1
  a8:   31 09           sbc r19, r1
                int a3 = 10;
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
  aa:   99 f7           brne    .-26        ; 0x92 <main+0x26>
                            TCCR0 = 0;
                            TIFR = 0x01;
                        PORTC ^= 1<<0;
                        --a5;
                        }
                        PORTC ^= 1<<1;
  ac:   85 b3           in  r24, 0x15   ; 21
  ae:   8e 27           eor r24, r30
  b0:   85 bb           out 0x15, r24   ; 21
  b2:   61 50           subi    r22, 0x01   ; 1
  b4:   71 09           sbc r23, r1
            int a2 = 10;
            while(a2) {
                int a3 = 10;
                while(a3) {
```

```
                        int a4 = 10;
                        while(a4) {
  b6:   19 f0           breq    .+6         ; 0xbe <main+0x52>
#include "avr/io.h"

int main() {
  b8:   2a e0           ldi r18, 0x0A   ; 10
  ba:   30 e0           ldi r19, 0x00   ; 0
  bc:   ea cf           rjmp    .-44        ; 0x92 <main+0x26>
                            --a5;
                        }
                        PORTC ^= 1<<1;
                        --a4;
                    }
                    PORTC ^= 1<<2;
  be:   85 b3           in  r24, 0x15   ; 21
  c0:   8d 25           eor r24, r13
  c2:   85 bb           out 0x15, r24   ; 21
  c4:   01 50           subi    r16, 0x01   ; 1
  c6:   11 09           sbc r17, r1
       int a1 = 10;
          while(a1) {
              int a2 = 10;
              while(a2) {
                  int a3 = 10;
                  while(a3) {
  c8:   19 f0               breq    .+6         ; 0xd0 <main+0x64>
#include "avr/io.h"

int main() {
  ca:   6a e0           ldi r22, 0x0A   ; 10
  cc:   70 e0           ldi r23, 0x00   ; 0
  ce:   f4 cf           rjmp    .-24        ; 0xb8 <main+0x4c>
                            --a4;
                        }
```

```
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
  d0:   85 b3           in  r24, 0x15   ; 21
  d2:   8e 25           eor r24, r14
  d4:   85 bb           out 0x15, r24   ; 21
  d6:   21 97           sbiw    r28, 0x01   ; 1

    while(1) {
         int a1 = 10;
            while(a1) {
                int a2 = 10;
                while(a2) {
  d8:   19 f0           breq    .+6         ; 0xe0 <main+0x74>
#include "avr/io.h"

int main() {
  da:   0a e0           ldi r16, 0x0A   ; 10
  dc:   10 e0           ldi r17, 0x00   ; 0
  de:   f5 cf           rjmp    .-22        ; 0xca <main+0x5e>
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
  e0:   85 b3           in  r24, 0x15   ; 21
  e2:   8f 25           eor r24, r15
  e4:   85 bb           out 0x15, r24   ; 21
  e6:   11 97           sbiw    r26, 0x01   ; 1

    DDRC = 0xFF;

    while(1) {
```

```
         int a1 = 10;
            while(a1) {
  e8:   19 f0           breq    .+6         ; 0xf0 <main+0x84>
#include "avr/io.h"

int main() {
  ea:   ca e0           ldi r28, 0x0A   ; 10
  ec:   d0 e0           ldi r29, 0x00   ; 0
  ee:   f5 cf           rjmp    .-22        ; 0xda <main+0x6e>
                    --a2;
                }
                PORTC ^= 1<<4;
                --a1;
            }
            PORTC ^= 1<<5;
  f0:   85 b3           in  r24, 0x15   ; 21
  f2:   8f 27           eor r24, r31
  f4:   85 bb           out 0x15, r24   ; 21
        }
  f6:   ca cf           rjmp    .-108       ; 0x8c <main+0x20>

000000f8 <_exit>:
  f8:   f8 94           cli

000000fa <__stop_program>:
  fa:   ff cf           rjmp    .-2         ; 0xfa <__stop_program>
```

TIMER 1 .asm

C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer1\timer1\main.asm 1

```
 1  .include "m32def.inc"                                                    ;clock cycles
 2
 3  ldi r31, high(ramend)                                                    ; 1
 4  out sph, r31                                                             ; 1
 5  ldi r31, low(ramend)                                                     ; 1
 6  out spl, r31                                                             ; 1
 7
 8  ldi r16, -65                                                             ; 1
 9  ldi r18, 0                                                               ; 1
10  ldi r17, 1<<0                                                            ; 1
11  ldi r19, 1<<1                                                            ; 1
12  ldi r20, 1<<2                                                            ; 1
13  ldi r21, 1<<3                                                            ; 1
14  ldi r22, 1<<4                                                            ; 1
15  ldi r23, 1<<5                                                            ; 1
16  ldi r29, low(65536-65)                                                   ; 1
17  ldi r30, high(65536-65)                                                  ; 1
18
19  ldi r31, 0xFF                                                            ; 1
20  out ddrc, r31                                                            ; 1
21
22  out tccr1a, r18                                                          ; 1
23
24  start : ldi r24, 10                                                      ; 1
25          loop1 : ldi r25, 10                                              ; 1
26                  loop2 : ldi r26, 10                                      ; 1
27                          loop3 : ldi r27, 10                              ; 1
28                                  loop4 : ldi r28, 10                      ; 1
29                                          loop5 :     out tcnt1h, r30      ; 1
30                                                      out tcnt1l, r29      ; 1
31                                                      out tccr1b, r17      ; 1
32                                                      again : in r20, tifr ; 1
33                                                              sbrs r20, tov1 ; 1 / 2
34                                                              rjmp again     ; 2
35                                                      out tccr1b, r18      ; 1
```

```
36                                              out tifr, r20                   ; 1
37                                          in r31, portc                       ; 1
38                                          eor r31, r17                        ; 1
39                                          out portc, r31                      ; 1
40                                          dec r28                             ; 1
41                                          brne loop5                          ; 1 / 2
42                                      in r31, portc                           ; 1
43                                      eor r31, r19                            ; 1
44                                      out portc, r31                          ; 1
45                                      dec r27                                 ; 1
46                                      brne loop4                              ; 1 / 2
47                                  in r31, portc                               ; 1
48                                  eor r31, r20                                ; 1
49                                  out portc, r31                              ; 1
50                                  dec r26                                     ; 1
51                                  brne loop3                                  ; 1 / 2
52                              in r31, portc                                   ; 1
53                              eor r31, r21                                    ; 1
54                              out portc, r31                                  ; 1
55                              dec r25                                         ; 1
56                              brne loop2                                      ; 1 / 2
57                          in r31, portc                                       ; 1
58                          eor r31, r22                                        ; 1
59                          out portc, r31                                      ; 1
60                          dec r24                                             ; 1
61                          brne loop1                                          ; 1 / 2
62                  in r31, portc                                               ; 1
63                  eor r31, r23                                                ; 1
64                  out portc, r31                                              ; 1
65                  rjmp start                                                  ; 2
66
67
```

TIMER 1 .C


```c
#include "avr/io.h"

int main() {

    DDRC = 0xFF;

    while(1) {
        int a1 = 10;
        while(a1) {
            int a2 = 10;
            while(a2) {
                int a3 = 10;
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
                            TCNT1H = 0xFF;
                            TCNT1L = 0xBF;
                            TCCR1A = 0;
                            TCCR1B = 0x01;
                            while(!(TIFR & (1<<TOV1)));
                            TCCR1B = 0;
                            TIFR = 1<<TOV1;
                            PORTC ^= 1<<0;
                            --a5;
                        }
                        PORTC ^= 1<<1;
                        --a4;
                    }
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
            --a1;
        }
        PORTC ^= 1<<5;
    }
    return 0;
}
```

.lss file for diassembly analysis of timer 1 .c code


```
timer1_c.elf:     file format elf32-avr

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         00000100  00000000  00000000  00000054  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  00800060  00800060  00000154  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .comment      00000030  00000000  00000000  00000154  2**0
                  CONTENTS, READONLY
  3 .note.gnu.avr.deviceinfo 0000003c  00000000  00000000  00000184  2**2
                  CONTENTS, READONLY
  4 .debug_aranges 00000020  00000000  00000000  000001c0  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_info   000005d2  00000000  00000000  000001e0  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_abbrev 0000050d  00000000  00000000  000007b2  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   00000278  00000000  00000000  00000cbf  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_frame  00000024  00000000  00000000  00000f38  2**2
                  CONTENTS, READONLY, DEBUGGING
  9 .debug_str    000002eb  00000000  00000000  00000f5c  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .debug_loc    000001e5  00000000  00000000  00001247  2**0
                  CONTENTS, READONLY, DEBUGGING
 11 .debug_ranges 000000e0  00000000  00000000  0000142c  2**0
                  CONTENTS, READONLY, DEBUGGING

Disassembly of section .text:

00000000 <__vectors>:
   0:   0c 94 2a 00     jmp 0x54    ; 0x54 <__ctors_end>
   4:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
```

```
   8:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
   c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  10:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  14:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  18:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  1c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  20:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  24:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  28:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  2c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  30:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  34:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  38:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  3c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  40:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  44:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  48:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  4c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  50:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>

00000054 <__ctors_end>:
  54:   11 24           eor r1, r1
  56:   1f be           out 0x3f, r1    ; 63
  58:   cf e5           ldi r28, 0x5F   ; 95
  5a:   d8 e0           ldi r29, 0x08   ; 8
  5c:   de bf           out 0x3e, r29   ; 62
  5e:   cd bf           out 0x3d, r28   ; 61
  60:   0e 94 36 00     call    0x6c    ; 0x6c <main>
  64:   0c 94 7e 00     jmp 0xfc    ; 0xfc <_exit>

00000068 <__bad_interrupt>:
  68:   0c 94 00 00     jmp 0   ; 0x0 <__vectors>

0000006c <main>:
#include "avr/io.h"
```

```
int main() {

    DDRC = 0xFF;
  6c:   8f ef           ldi r24, 0xFF   ; 255
  6e:   84 bb           out 0x14, r24   ; 20
                        int a5 = 10;
                        while(a5) {
                            TCNT1H = 0xFF;
                            TCNT1L = 0xBF;
                            TCCR1A = 0;
                            TCCR1B = 0x01;
  70:   91 e0           ldi r25, 0x01   ; 1
                            while(!(TIFR & (1<<TOV1)));
                            TCCR1B = 0;
                            TIFR = 1<<TOV1;
  72:   44 e0           ldi r20, 0x04   ; 4
                            PORTC ^= 1<<0;
                            --a5;
                        }
                        PORTC ^= 1<<1;
  74:   68 94           set
  76:   ff 24           eor r15, r15
  78:   f1 f8           bld r15, 1
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
                            TCNT1H = 0xFF;
  7a:   7f ef           ldi r23, 0xFF   ; 255
                            TCNT1L = 0xBF;
  7c:   6f eb           ldi r22, 0xBF   ; 191
                            TCCR1A = 0;
                            TCCR1B = 0x01;
```

```
  7e:   59 2f           mov r21, r25
#include "avr/io.h"

int main() {
  80:   aa e0           ldi r26, 0x0A   ; 10
  82:   b0 e0           ldi r27, 0x00   ; 0
                        --a4;
                    }
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
  84:   68 94           set
  86:   dd 24           eor r13, r13
  88:   d3 f8           bld r13, 3
                --a2;
            }
            PORTC ^= 1<<4;
  8a:   68 94           set
  8c:   ee 24           eor r14, r14
  8e:   e4 f8           bld r14, 4
  90:   2d c0           rjmp    .+90        ; 0xec <main+0x80>
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
                            TCNT1H = 0xFF;
  92:   7d bd           out 0x2d, r23   ; 45
                            TCNT1L = 0xBF;
  94:   6c bd           out 0x2c, r22   ; 44
                            TCCR1A = 0;
  96:   1f bc           out 0x2f, r1    ; 47
                            TCCR1B = 0x01;
  98:   5e bd           out 0x2e, r21   ; 46
```

```
                        while(!(TIFR & (1<<TOV1)));
  9a:   08 b6           in  r0, 0x38    ; 56
  9c:   02 fe           sbrs    r0, 2
  9e:   fd cf           rjmp    .-6         ; 0x9a <main+0x2e>
                        TCCR1B = 0;
  a0:   1e bc           out 0x2e, r1    ; 46
                        TIFR = 1<<TOV1;
  a2:   48 bf           out 0x38, r20   ; 56
                        PORTC ^= 1<<0;
  a4:   85 b3           in  r24, 0x15   ; 21
  a6:   89 27           eor r24, r25
  a8:   85 bb           out 0x15, r24   ; 21
  aa:   21 50           subi    r18, 0x01   ; 1
  ac:   31 09           sbc r19, r1
              int a3 = 10;
              while(a3) {
                  int a4 = 10;
                  while(a4) {
                      int a5 = 10;
                      while(a5) {
  ae:   89 f7           brne    .-30        ; 0x92 <main+0x26>
                          TCCR1B = 0;
                          TIFR = 1<<TOV1;
                          PORTC ^= 1<<0;
                          --a5;
                      }
                      PORTC ^= 1<<1;
  b0:   85 b3           in  r24, 0x15   ; 21
  b2:   8f 25           eor r24, r15
  b4:   85 bb           out 0x15, r24   ; 21
  b6:   31 97           sbiw    r30, 0x01   ; 1
          int a2 = 10;
          while(a2) {
              int a3 = 10;
              while(a3) {
```

```
                    int a4 = 10;
                    while(a4) {
  b8:   19 f0           breq    .+6         ; 0xc0 <main+0x54>
#include "avr/io.h"

int main() {
  ba:   2a e0           ldi r18, 0x0A   ; 10
  bc:   30 e0           ldi r19, 0x00   ; 0
  be:   e9 cf           rjmp    .-46        ; 0x92 <main+0x26>
                        --a5;
                    }
                    PORTC ^= 1<<1;
                    --a4;
                }
                PORTC ^= 1<<2;
  c0:   85 b3           in  r24, 0x15   ; 21
  c2:   84 27           eor r24, r20
  c4:   85 bb           out 0x15, r24   ; 21
  c6:   01 50           subi    r16, 0x01   ; 1
  c8:   11 09           sbc r17, r1
      int a1 = 10;
      while(a1) {
          int a2 = 10;
          while(a2) {
              int a3 = 10;
              while(a3) {
  ca:   19 f0           breq    .+6         ; 0xd2 <main+0x66>
#include "avr/io.h"

int main() {
  cc:   ea e0           ldi r30, 0x0A   ; 10
  ce:   f0 e0           ldi r31, 0x00   ; 0
  d0:   f4 cf           rjmp    .-24        ; 0xba <main+0x4e>
                    --a4;
                }
```

```
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
  d2:   85 b3           in  r24, 0x15   ; 21
  d4:   8d 25           eor r24, r13
  d6:   85 bb           out 0x15, r24   ; 21
  d8:   21 97           sbiw    r28, 0x01   ; 1

    while(1) {
        int a1 = 10;
        while(a1) {
            int a2 = 10;
            while(a2) {
  da:   19 f0           breq    .+6         ; 0xe2 <main+0x76>
#include "avr/io.h"

int main() {
  dc:   0a e0           ldi r16, 0x0A   ; 10
  de:   10 e0           ldi r17, 0x00   ; 0
  e0:   f5 cf           rjmp    .-22        ; 0xcc <main+0x60>
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
  e2:   85 b3           in  r24, 0x15   ; 21
  e4:   8e 25           eor r24, r14
  e6:   85 bb           out 0x15, r24   ; 21
  e8:   11 97           sbiw    r26, 0x01   ; 1

    DDRC = 0xFF;

    while(1) {
```

```
        int a1 = 10;
        while(a1) {
  ea:   19 f0           breq    .+6         ; 0xf2 <main+0x86>
#include "avr/io.h"

int main() {
  ec:   ca e0           ldi r28, 0x0A   ; 10
  ee:   d0 e0           ldi r29, 0x00   ; 0
  f0:   f5 cf           rjmp    .-22        ; 0xdc <main+0x70>
                --a2;
            }
            PORTC ^= 1<<4;
            --a1;
        }
        PORTC ^= 1<<5;
  f2:   25 b3           in  r18, 0x15   ; 21
  f4:   80 e2           ldi r24, 0x20   ; 32
  f6:   82 27           eor r24, r18
  f8:   85 bb           out 0x15, r24   ; 21
    }
  fa:   c2 cf           rjmp    .-124       ; 0x80 <main+0x14>

000000fc <_exit>:
  fc:   f8 94           cli

000000fe <__stop_program>:
  fe:   ff cf           rjmp    .-2         ; 0xfe <__stop_program>
```

Timer 2 .asm

C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer2\timer2\main.asm 1

```
1  .include "m32def.inc"                                                ;clock cycles
2
3  ldi r31, high(ramend)                                                ; 1
4  out sph, r31                                                         ; 1
5  ldi r31, low(ramend)                                                 ; 1
6  out spl, r31                                                         ; 1
7
8  ldi r16, -65                                                         ; 1
9  ldi r18, 0                                                           ; 1
10 ldi r17, 1<<0                                                        ; 1
11 ldi r19, 1<<1                                                        ; 1
12 ldi r20, 1<<2                                                        ; 1
13 ldi r21, 1<<3                                                        ; 1
14 ldi r22, 1<<4                                                        ; 1
15 ldi r23, 1<<5                                                        ; 1
16 ldi r29, 1<<6                                                        ; 1
17
18 ldi r31, 0xFF                                                        ; 1
19 out ddrc, r31                                                        ; 1
20
21 start : ldi r24, 10                                                  ; 1
22         loop1 : ldi r25, 10                                          ; 1
23                 loop2 : ldi r26, 10                                  ; 1
24                         loop3 : ldi r27, 10                          ; 1
25                                 loop4 : ldi r28, 10                  ; 1
26                                         loop5 :     out tcnt2, r16   ; 1
27                                                     out tccr2, r17   ; 1
28                                                     again : in r20, tifr ; 1
29                                                             sbrs r20, tov2 ; 1 / 2
30                                                             rjmp again ; 2
31                                                     out tccr2, r18   ; 1
32                                                     out tifr, r29    ; 1
33                                                 in r31, portc        ; 1
34                                                 eor r31, r17         ; 1
35                                                 out portc, r31       ; 1
```

```
36                                          dec r28                     ; 1
37                                          brne loop5                  ; 1 / 2
38                                      in r31, portc                   ; 1
39                                      eor r31, r19                    ; 1
40                                      out portc, r31                  ; 1
41                                      dec r27                         ; 1
42                                      brne loop4                      ; 1 / 2
43                                  in r31, portc                       ; 1
44                                  eor r31, r20                        ; 1
45                                  out portc, r31                      ; 1
46                                  dec r26                             ; 1
47                                  brne loop3                          ; 1 / 2
48                              in r31, portc                           ; 1
49                              eor r31, r21                            ; 1
50                              out portc, r31                          ; 1
51                              dec r25                                 ; 1
52                              brne loop2                              ; 1 / 2
53                          in r31, portc                               ; 1
54                          eor r31, r22                                ; 1
55                          out portc, r31                              ; 1
56                          dec r24                                     ; 1
57                          brne loop1                                  ; 1 / 2
58                  in r31, portc                                       ; 1
59                  eor r31, r23                                        ; 1
60                  out portc, r31                                      ; 1
61                  rjmp start                                          ; 2
62
63
```

timer 2, .c file


```c
#include "avr/io.h"

int main() {

    DDRC = 0xFF;

    while(1) {
        int a1 = 10;
        while(a1) {
            int a2 = 10;
            while(a2) {
                int a3 = 10;
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
                            TCNT2 = -65;
                            TCCR2 = 0x01;
                            while(!(TIFR & (1<<TOV2)));
                            TCCR2 = 0;
                            TIFR = 1<<TOV2;
                            PORTC ^= 1<<0;
                            --a5;
                        }
                        PORTC ^= 1<<1;
                        --a4;
                    }
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
```

```
            --a1;
        }
        PORTC ^= 1<<5;
    }
    return 0;
}
```

.lss file for diassembly analysis of timer 2 .c code


```
timer2_c.elf:     file format elf32-avr

Sections:
Idx Name          Size      VMA       LMA       File off  Algn
  0 .text         000000fc  00000000  00000000  00000054  2**1
                  CONTENTS, ALLOC, LOAD, READONLY, CODE
  1 .data         00000000  00800060  00800060  00000150  2**0
                  CONTENTS, ALLOC, LOAD, DATA
  2 .comment      00000030  00000000  00000000  00000150  2**0
                  CONTENTS, READONLY
  3 .note.gnu.avr.deviceinfo 0000003c  00000000  00000000  00000180  2**2
                  CONTENTS, READONLY
  4 .debug_aranges 00000020  00000000  00000000  000001bc  2**0
                  CONTENTS, READONLY, DEBUGGING
  5 .debug_info   000005d2  00000000  00000000  000001dc  2**0
                  CONTENTS, READONLY, DEBUGGING
  6 .debug_abbrev 0000050d  00000000  00000000  000007ae  2**0
                  CONTENTS, READONLY, DEBUGGING
  7 .debug_line   0000026c  00000000  00000000  00000cbb  2**0
                  CONTENTS, READONLY, DEBUGGING
  8 .debug_frame  00000024  00000000  00000000  00000f28  2**2
                  CONTENTS, READONLY, DEBUGGING
  9 .debug_str    000002eb  00000000  00000000  00000f4c  2**0
                  CONTENTS, READONLY, DEBUGGING
 10 .debug_loc    000001e5  00000000  00000000  00001237  2**0
                  CONTENTS, READONLY, DEBUGGING
 11 .debug_ranges 000000d8  00000000  00000000  0000141c  2**0
                  CONTENTS, READONLY, DEBUGGING

Disassembly of section .text:

00000000 <__vectors>:
   0:   0c 94 2a 00     jmp 0x54    ; 0x54 <__ctors_end>
   4:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
```

```
   8:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
   c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  10:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  14:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  18:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  1c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  20:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  24:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  28:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  2c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  30:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  34:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  38:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  3c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  40:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  44:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  48:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  4c:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>
  50:   0c 94 34 00     jmp 0x68    ; 0x68 <__bad_interrupt>

00000054 <__ctors_end>:
  54:   11 24           eor r1, r1
  56:   1f be           out 0x3f, r1    ; 63
  58:   cf e5           ldi r28, 0x5F   ; 95
  5a:   d8 e0           ldi r29, 0x08   ; 8
  5c:   de bf           out 0x3e, r29   ; 62
  5e:   cd bf           out 0x3d, r28   ; 61
  60:   0e 94 36 00     call    0x6c    ; 0x6c <main>
  64:   0c 94 7c 00     jmp 0xf8    ; 0xf8 <_exit>

00000068 <__bad_interrupt>:
  68:   0c 94 00 00     jmp 0   ; 0x0 <__vectors>

0000006c <main>:
#include "avr/io.h"
```

```
int main() {

    DDRC = 0xFF;
  6c:   8f ef           ldi r24, 0xFF   ; 255
  6e:   84 bb           out 0x14, r24   ; 20
                int a4 = 10;
                while(a4) {
                    int a5 = 10;
                    while(a5) {
                        TCNT2 = -65;
                        TCCR2 = 0x01;
  70:   91 e0           ldi r25, 0x01   ; 1
                        while(!(TIFR & (1<<TOV2)));
                        TCCR2 = 0;
                        TIFR = 1<<TOV2;
  72:   60 e4           ldi r22, 0x40   ; 64
                        PORTC ^= 1<<0;
                        --a5;
                    }
                    PORTC ^= 1<<1;
  74:   72 e0           ldi r23, 0x02   ; 2
            while(a3) {
                int a4 = 10;
                while(a4) {
                    int a5 = 10;
                    while(a5) {
                        TCNT2 = -65;
  76:   5f eb           ldi r21, 0xBF   ; 191
                        TCCR2 = 0x01;
  78:   49 2f           mov r20, r25
                        --a5;
                    }
                    PORTC ^= 1<<1;
                    --a4;
```

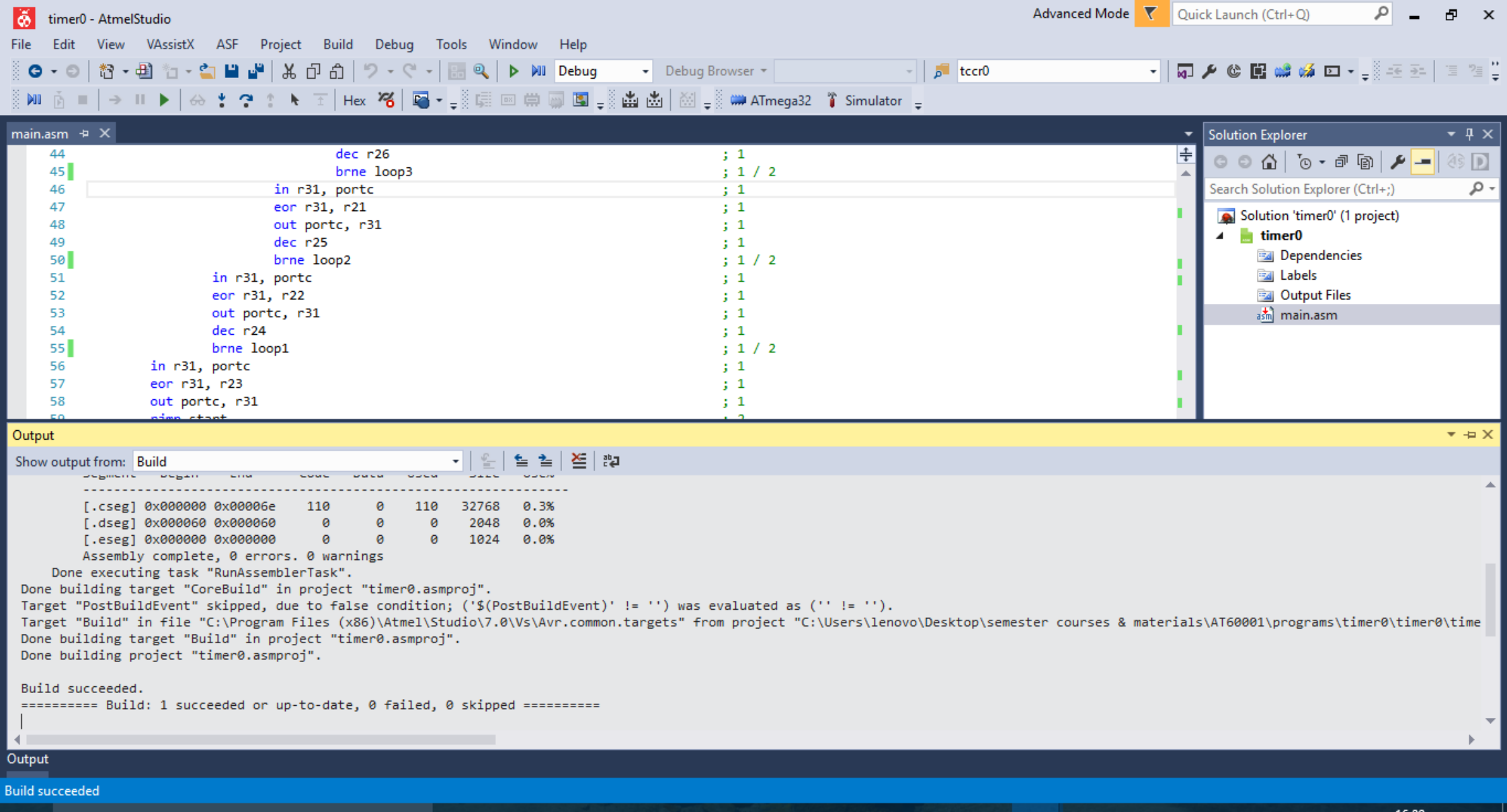
```
                    }
                    PORTC ^= 1<<2;
  7a:   68 94           set
  7c:   ee 24           eor r14, r14
  7e:   e2 f8           bld r14, 2
                    --a3;
                }
                PORTC ^= 1<<3;
  80:   68 94           set
  82:   ff 24           eor r15, r15
  84:   f3 f8           bld r15, 3
#include "avr/io.h"

int main() {
  86:   aa e0           ldi r26, 0x0A   ; 10
  88:   b0 e0           ldi r27, 0x00   ; 0
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
  8a:   68 94           set
  8c:   dd 24           eor r13, r13
  8e:   d4 f8           bld r13, 4
  90:   2b c0           rjmp    .+86        ; 0xe8 <main+0x7c>
                while(a3) {
                    int a4 = 10;
                    while(a4) {
                        int a5 = 10;
                        while(a5) {
                            TCNT2 = -65;
  92:   54 bd           out 0x24, r21   ; 36
                            TCCR2 = 0x01;
  94:   45 bd           out 0x25, r20   ; 37
```

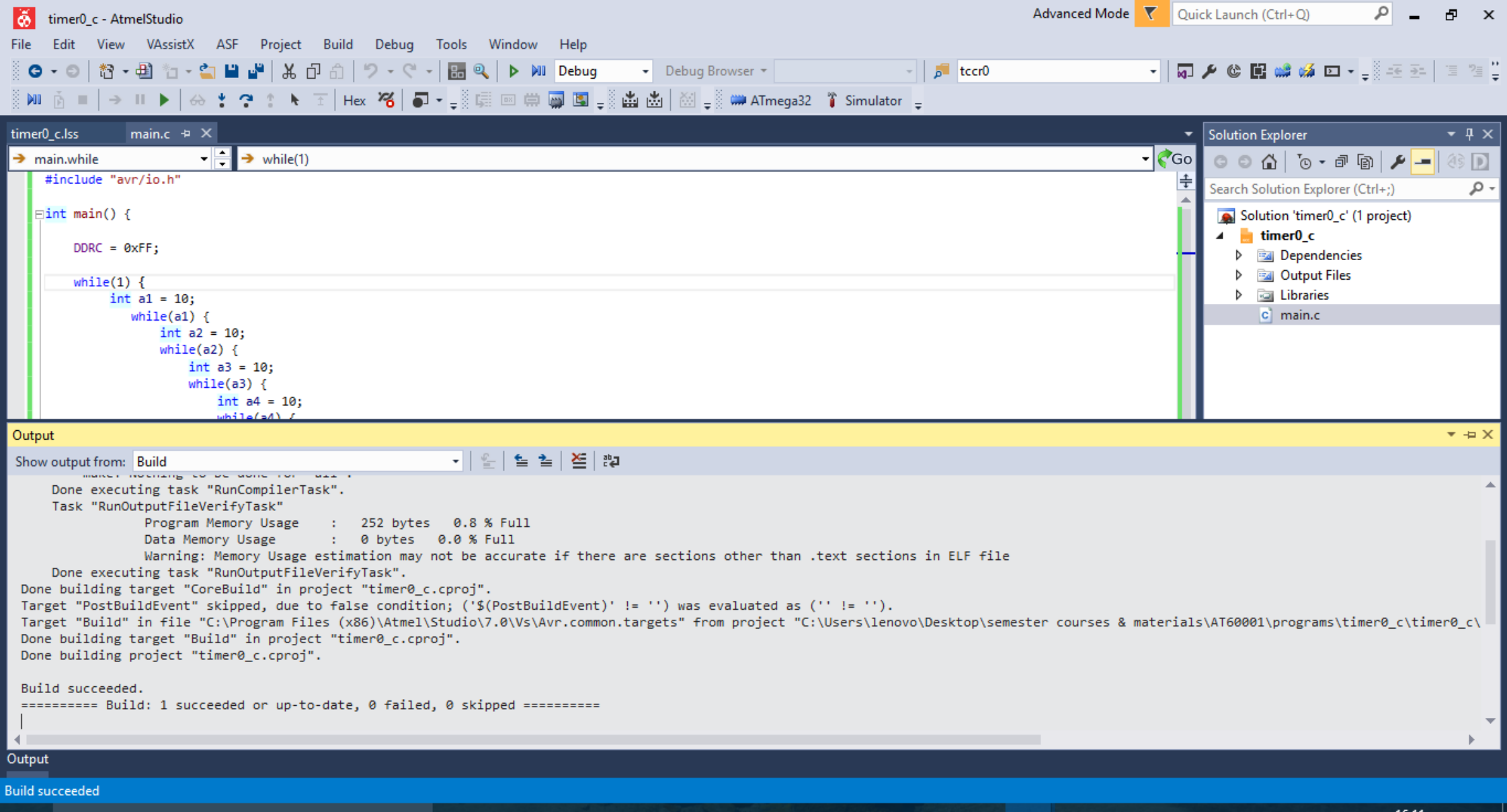
```
                        while(!(TIFR & (1<<TOV2)));
  96:   08 b6           in  r0, 0x38    ; 56
  98:   06 fe           sbrs    r0, 6
  9a:   fd cf           rjmp    .-6         ; 0x96 <main+0x2a>
                        TCCR2 = 0;
  9c:   15 bc           out 0x25, r1    ; 37
                        TIFR = 1<<TOV2;
  9e:   68 bf           out 0x38, r22   ; 56
                        PORTC ^= 1<<0;
  a0:   85 b3           in  r24, 0x15   ; 21
  a2:   89 27           eor r24, r25
  a4:   85 bb           out 0x15, r24   ; 21
  a6:   21 50           subi    r18, 0x01   ; 1
  a8:   31 09           sbc r19, r1
              int a3 = 10;
              while(a3) {
                  int a4 = 10;
                  while(a4) {
                      int a5 = 10;
                      while(a5) {
  aa:   99 f7           brne    .-26        ; 0x92 <main+0x26>
                          TCCR2 = 0;
                          TIFR = 1<<TOV2;
                          PORTC ^= 1<<0;
                          --a5;
                      }
                      PORTC ^= 1<<1;
  ac:   85 b3           in  r24, 0x15   ; 21
  ae:   87 27           eor r24, r23
  b0:   85 bb           out 0x15, r24   ; 21
  b2:   31 97           sbiw    r30, 0x01   ; 1
          int a2 = 10;
          while(a2) {
              int a3 = 10;
              while(a3) {
```

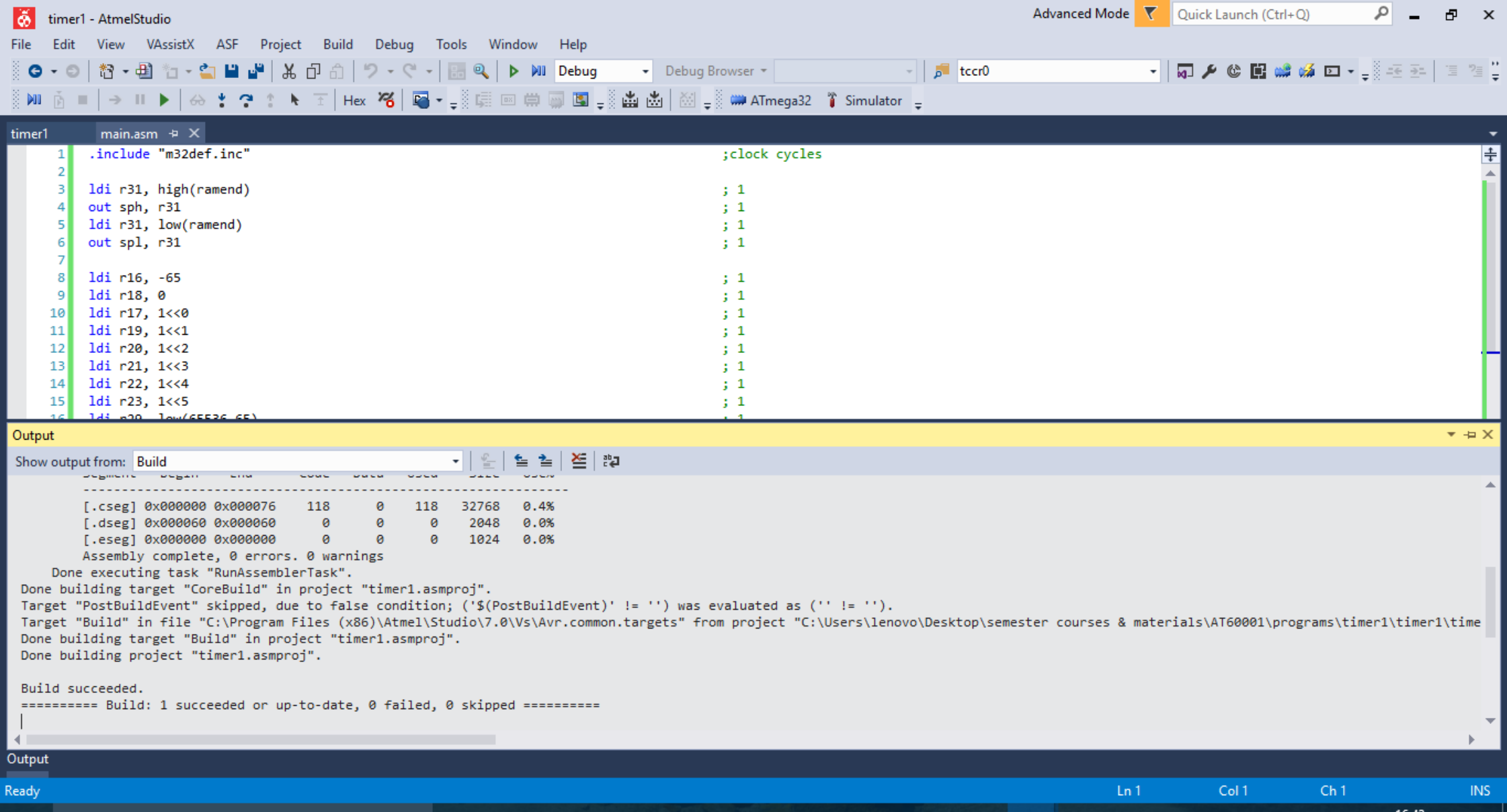
```
                    int a4 = 10;
                    while(a4) {
  b4:   19 f0           breq    .+6         ; 0xbc <main+0x50>
#include "avr/io.h"

int main() {
  b6:   2a e0           ldi r18, 0x0A   ; 10
  b8:   30 e0           ldi r19, 0x00   ; 0
  ba:   eb cf           rjmp    .-42        ; 0x92 <main+0x26>
                        --a5;
                    }
                    PORTC ^= 1<<1;
                    --a4;
                }
                PORTC ^= 1<<2;
  bc:   85 b3           in  r24, 0x15   ; 21
  be:   8e 25           eor r24, r14
  c0:   85 bb           out 0x15, r24   ; 21
  c2:   01 50           subi    r16, 0x01   ; 1
  c4:   11 09           sbc r17, r1
      int a1 = 10;
      while(a1) {
          int a2 = 10;
          while(a2) {
              int a3 = 10;
              while(a3) {
  c6:   19 f0           breq    .+6         ; 0xce <main+0x62>
#include "avr/io.h"

int main() {
  c8:   ea e0           ldi r30, 0x0A   ; 10
  ca:   f0 e0           ldi r31, 0x00   ; 0
  cc:   f4 cf           rjmp    .-24        ; 0xb6 <main+0x4a>
                    --a4;
                }
```

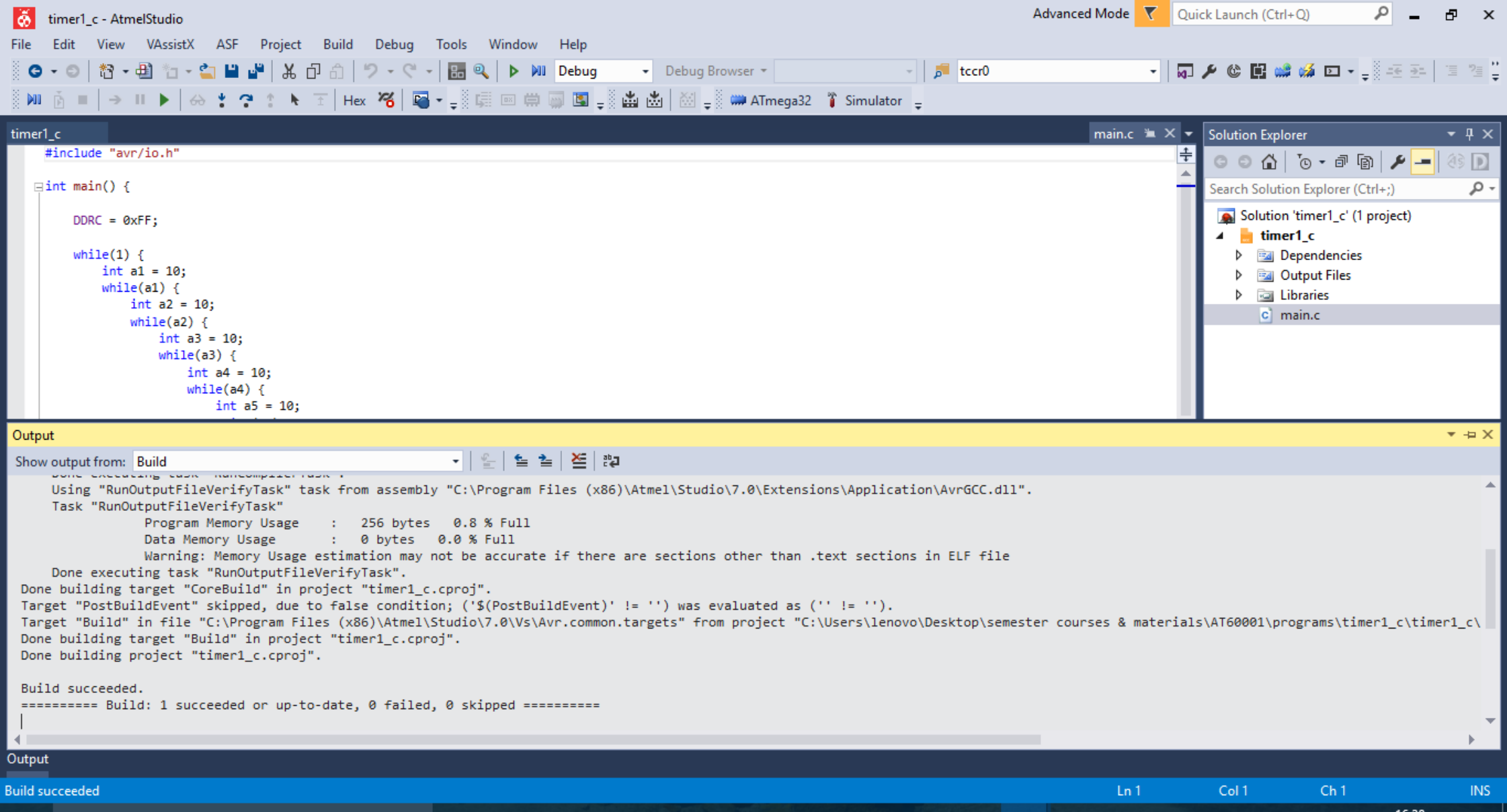
```
                    PORTC ^= 1<<2;
                    --a3;
                }
                PORTC ^= 1<<3;
  ce:   85 b3           in  r24, 0x15   ; 21
  d0:   8f 25           eor r24, r15
  d2:   85 bb           out 0x15, r24   ; 21
  d4:   21 97           sbiw    r28, 0x01   ; 1

    while(1) {
        int a1 = 10;
        while(a1) {
            int a2 = 10;
            while(a2) {
  d6:   19 f0           breq    .+6         ; 0xde <main+0x72>
#include "avr/io.h"

int main() {
  d8:   0a e0           ldi r16, 0x0A   ; 10
  da:   10 e0           ldi r17, 0x00   ; 0
  dc:   f5 cf           rjmp    .-22        ; 0xc8 <main+0x5c>
                    --a3;
                }
                PORTC ^= 1<<3;
                --a2;
            }
            PORTC ^= 1<<4;
  de:   85 b3           in  r24, 0x15   ; 21
  e0:   8d 25           eor r24, r13
  e2:   85 bb           out 0x15, r24   ; 21
  e4:   11 97           sbiw    r26, 0x01   ; 1

    DDRC = 0xFF;

    while(1) {
```

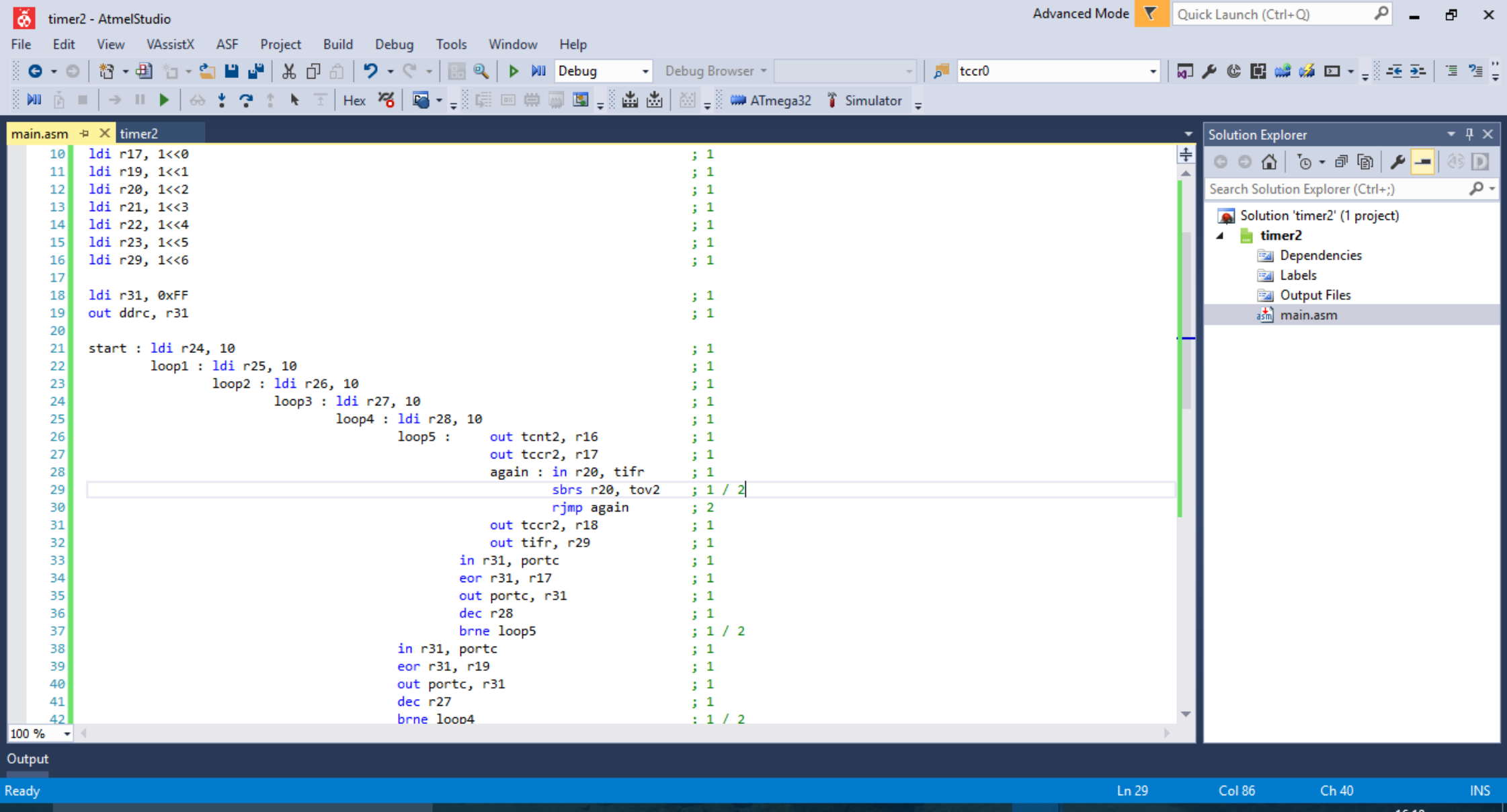
```
        int a1 = 10;
        while(a1) {
  e6:   19 f0           breq    .+6         ; 0xee <main+0x82>
#include "avr/io.h"

int main() {
  e8:   ca e0           ldi r28, 0x0A   ; 10
  ea:   d0 e0           ldi r29, 0x00   ; 0
  ec:   f5 cf           rjmp    .-22        ; 0xd8 <main+0x6c>
                --a2;
            }
            PORTC ^= 1<<4;
            --a1;
        }
        PORTC ^= 1<<5;
  ee:   25 b3           in  r18, 0x15   ; 21
  f0:   80 e2           ldi r24, 0x20   ; 32
  f2:   82 27           eor r24, r18
  f4:   85 bb           out 0x15, r24   ; 21
    }
  f6:   c7 cf           rjmp    .-114       ; 0x86 <main+0x1a>

000000f8 <_exit>:
  f8:   f8 94           cli

000000fa <__stop_program>:
  fa:   ff cf           rjmp    .-2         ; 0xfa <__stop_program>
```

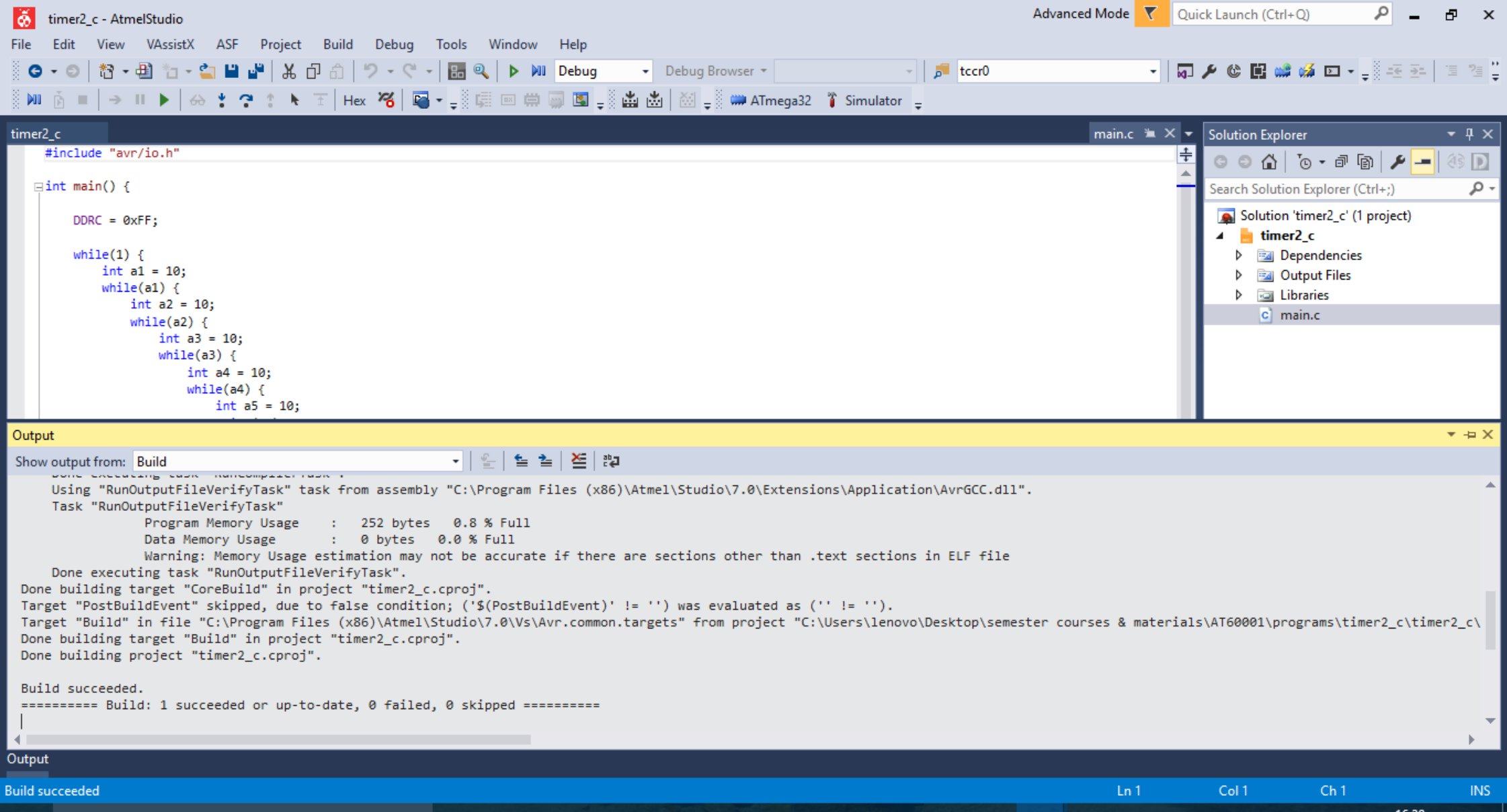

timer0 - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
main.asm
44 dec r26 ; 1
45 brne loop3 ; 1 / 2
46 in r31, portc ; 1
47 eor r31, r21 ; 1
48 out portc, r31 ; 1
49 dec r25 ; 1
50 brne loop2 ; 1 / 2
51 in r31, portc ; 1
52 eor r31, r22 ; 1
53 out portc, r31 ; 1
54 dec r24 ; 1
55 brne loop1 ; 1 / 2
56 in r31, portc ; 1
57 eor r31, r23 ; 1
58 out portc, r31 ; 1
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'timer0' (1 project)
timer0
Dependencies
Labels
Output Files
main.asm
Output
Show output from: Build
[.cseg] 0x000000 0x00006e 110 0 110 32768 0.3%
[.dseg] 0x000060 0x000060 0 0 0 2048 0.0%
[.eseg] 0x000000 0x000000 0 0 0 1024 0.0%
Assembly complete, 0 errors. 0 warnings
Done executing task "RunAssemblerTask".
Done building target "CoreBuild" in project "timer0.asmproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer0\timer0\time
Done building target "Build" in project "timer0.asmproj".
Done building project "timer0.asmproj".
Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
Output
Build succeeded


timer0_c - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
timer0_c.lss
main.c
main.while
while(1)
Go
#include "avr/io.h"
int main() {
DDRC = 0xFF;
while(1) {
int a1 = 10;
while(a1) {
int a2 = 10;
while(a2) {
int a3 = 10;
while(a3) {
int a4 = 10;
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'timer0_c' (1 project)
timer0_c
Dependencies
Output Files
Libraries
main.c
Output
Show output from: Build
Done executing task "RunCompilerTask".
Task "RunOutputFileVerifyTask"
Program Memory Usage : 252 bytes 0.8 % Full
Data Memory Usage : 0 bytes 0.0 % Full
Warning: Memory Usage estimation may not be accurate if there are sections other than .text sections in ELF file
Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "timer0_c.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer0_c\timer0_c\
Done building target "Build" in project "timer0_c.cproj".
Done building project "timer0_c.cproj".
Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
Output
Build succeeded


timer1 - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
timer1
main.asm
1 .include "m32def.inc" ;clock cycles
2
3 ldi r31, high(ramend) ; 1
4 out sph, r31 ; 1
5 ldi r31, low(ramend) ; 1
6 out spl, r31 ; 1
7
8 ldi r16, -65 ; 1
9 ldi r18, 0 ; 1
10 ldi r17, 1<<0 ; 1
11 ldi r19, 1<<1 ; 1
12 ldi r20, 1<<2 ; 1
13 ldi r21, 1<<3 ; 1
14 ldi r22, 1<<4 ; 1
15 ldi r23, 1<<5 ; 1
Output
Show output from: Build
[.cseg] 0x000000 0x000076 118 0 118 32768 0.4%
[.dseg] 0x000060 0x000060 0 0 0 2048 0.0%
[.eseg] 0x000000 0x000000 0 0 0 1024 0.0%
Assembly complete, 0 errors. 0 warnings
Done executing task "RunAssemblerTask".
Done building target "CoreBuild" in project "timer1.asmproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer1\timer1\time
Done building target "Build" in project "timer1.asmproj".
Done building project "timer1.asmproj".
Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
Output
Ready
Ln 1
Col 1
Ch 1
INS


timer1_c - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
timer1_c
main.c
#include "avr/io.h"
int main() {
DDRC = 0xFF;
while(1) {
int a1 = 10;
while(a1) {
int a2 = 10;
while(a2) {
int a3 = 10;
while(a3) {
int a4 = 10;
while(a4) {
int a5 = 10;
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'timer1_c' (1 project)
timer1_c
Dependencies
Output Files
Libraries
main.c
Output
Show output from: Build
Using "RunOutputFileVerifyTask" task from assembly "C:\Program Files (x86)\Atmel\Studio\7.0\Extensions\Application\AvrGCC.dll".
Task "RunOutputFileVerifyTask"
Program Memory Usage : 256 bytes 0.8 % Full
Data Memory Usage : 0 bytes 0.0 % Full
Warning: Memory Usage estimation may not be accurate if there are sections other than .text sections in ELF file
Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "timer1_c.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer1_c\timer1_c\
Done building target "Build" in project "timer1_c.cproj".
Done building project "timer1_c.cproj".
Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
Output
Build succeeded
Ln 1
Col 1
Ch 1
INS


timer2 - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
main.asm
timer2
10 ldi r17, 1<<0 ; 1
11 ldi r19, 1<<1 ; 1
12 ldi r20, 1<<2 ; 1
13 ldi r21, 1<<3 ; 1
14 ldi r22, 1<<4 ; 1
15 ldi r23, 1<<5 ; 1
16 ldi r29, 1<<6 ; 1
17
18 ldi r31, 0xFF ; 1
19 out ddrc, r31 ; 1
20
21 start : ldi r24, 10 ; 1
22 loop1 : ldi r25, 10 ; 1
23 loop2 : ldi r26, 10 ; 1
24 loop3 : ldi r27, 10 ; 1
25 loop4 : ldi r28, 10 ; 1
26 loop5 : out tcnt2, r16 ; 1
27 out tccr2, r17 ; 1
28 again : in r20, tifr ; 1
29 sbrs r20, tov2 ; 1 / 2
30 rjmp again ; 2
31 out tccr2, r18 ; 1
32 out tifr, r29 ; 1
33 in r31, portc ; 1
34 eor r31, r17 ; 1
35 out portc, r31 ; 1
36 dec r28 ; 1
37 brne loop5 ; 1 / 2
38 in r31, portc ; 1
39 eor r31, r19 ; 1
40 out portc, r31 ; 1
41 dec r27 ; 1
42 brne loop4 ; 1 / 2
100 %
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'timer2' (1 project)
timer2
Dependencies
Labels
Output Files
main.asm
Output
Ready
Ln 29
Col 86
Ch 40
INS


timer2_c - AtmelStudio
Advanced Mode
Quick Launch (Ctrl+Q)
File Edit View VAssistX ASF Project Build Debug Tools Window Help
Debug
Debug Browser
tccr0
Hex
ATmega32
Simulator
timer2_c
main.c
#include "avr/io.h"
int main() {
DDRC = 0xFF;
while(1) {
int a1 = 10;
while(a1) {
int a2 = 10;
while(a2) {
int a3 = 10;
while(a3) {
int a4 = 10;
while(a4) {
int a5 = 10;
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'timer2_c' (1 project)
timer2_c
Dependencies
Output Files
Libraries
main.c
Output
Show output from: Build
Using "RunOutputFileVerifyTask" task from assembly "C:\Program Files (x86)\Atmel\Studio\7.0\Extensions\Application\AvrGCC.dll".
Task "RunOutputFileVerifyTask"
Program Memory Usage : 252 bytes 0.8 % Full
Data Memory Usage : 0 bytes 0.0 % Full
Warning: Memory Usage estimation may not be accurate if there are sections other than .text sections in ELF file
Done executing task "RunOutputFileVerifyTask".
Done building target "CoreBuild" in project "timer2_c.cproj".
Target "PostBuildEvent" skipped, due to false condition; ('$(PostBuildEvent)' != '') was evaluated as ('' != '').
Target "Build" in file "C:\Program Files (x86)\Atmel\Studio\7.0\Vs\Avr.common.targets" from project "C:\Users\lenovo\Desktop\semester courses & materials\AT60001\programs\timer2_c\timer2_c\
Done building target "Build" in project "timer2_c.cproj".
Done building project "timer2_c.cproj".
Build succeeded.
========== Build: 1 succeeded or up-to-date, 0 failed, 0 skipped ==========
Output
Build succeeded
Ln 1
Col 1
Ch 1
INS