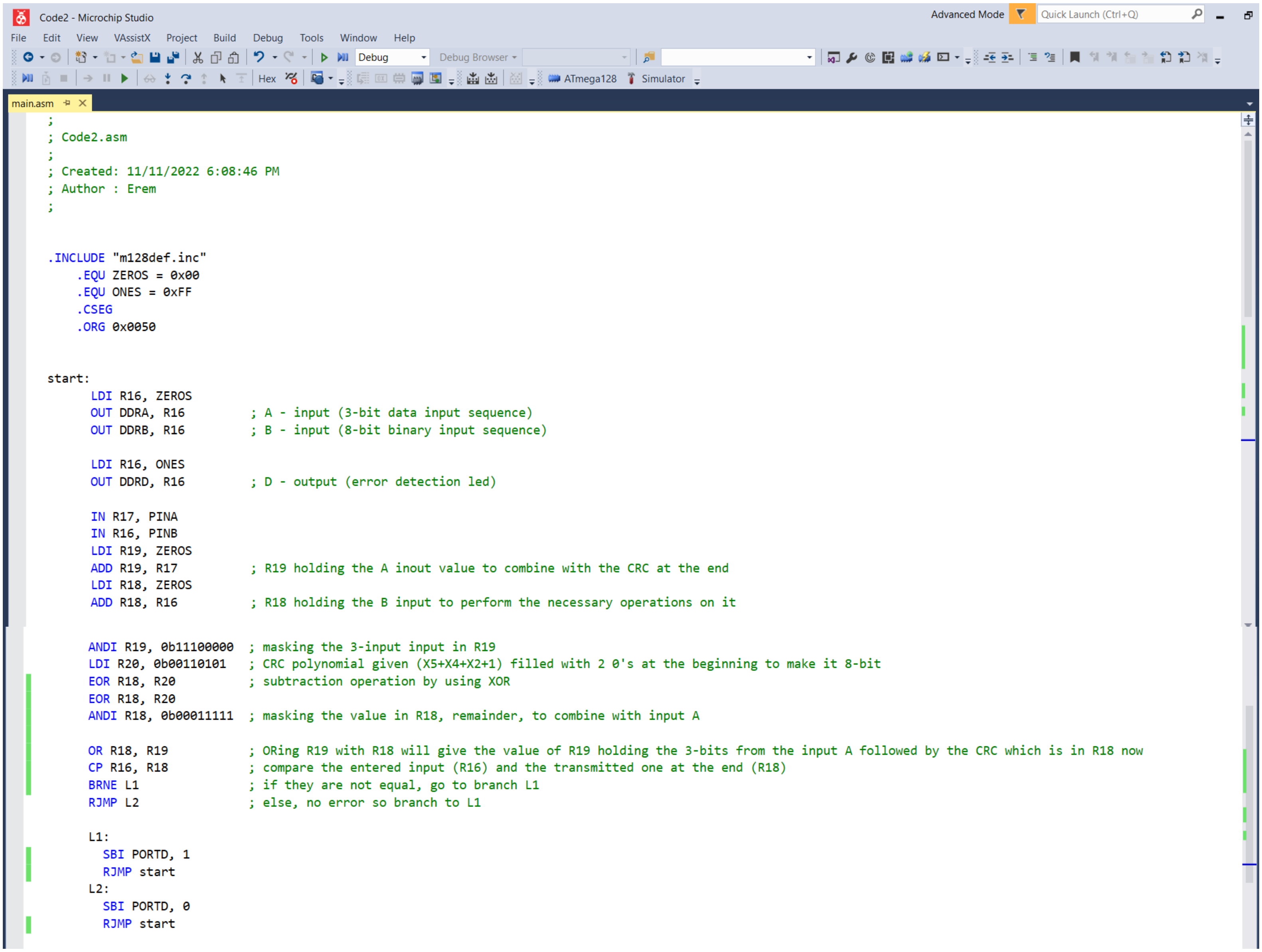
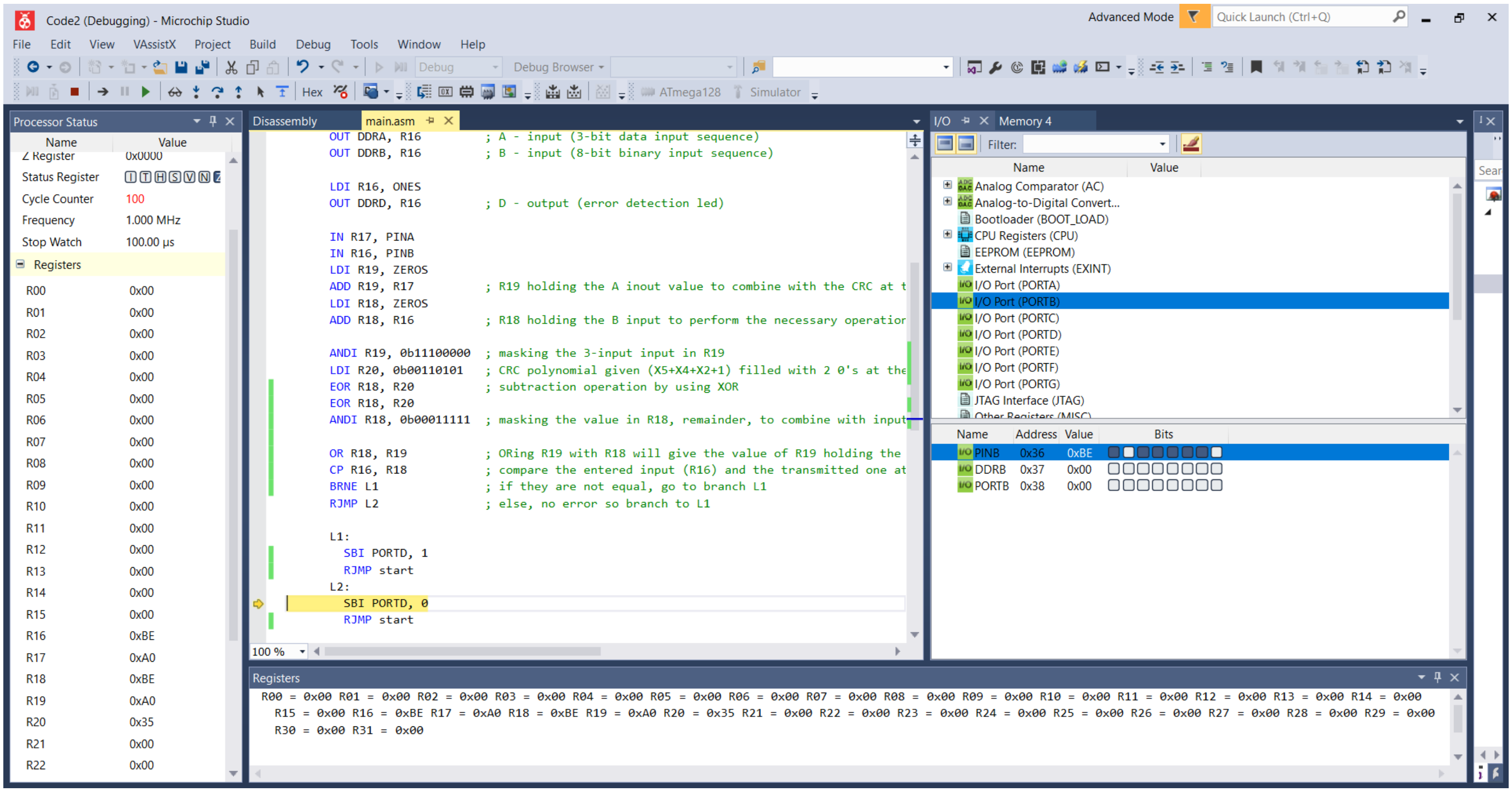
Fatma Erem AKSOY
2315075



```
;
; Code2.asm
;
; Created: 11/11/2022 6:08:46 PM
; Author : Erem
;


.INCLUDE "m128def.inc"
    .EQU ZEROS = 0x00
    .EQU ONES = 0xFF
    .CSEG
    .ORG 0x0050


start:
      LDI R16, ZEROS
      OUT DDRA, R16         ; A - input (3-bit data input sequence)
      OUT DDRB, R16         ; B - input (8-bit binary input sequence)

      LDI R16, ONES
      OUT DDRD, R16         ; D - output (error detection led)

      IN R17, PINA
      IN R16, PINB
      LDI R19, ZEROS
      ADD R19, R17          ; R19 holding the A inout value to combine with the CRC at the end
      LDI R18, ZEROS
      ADD R18, R16          ; R18 holding the B input to perform the necessary operations on it


      ANDI R19, 0b11100000  ; masking the 3-input input in R19
      LDI R20, 0b00110101   ; CRC polynomial given (X5+X4+X2+1) filled with 2 0's at the beginning to make it 8-bit
      EOR R18, R20          ; subtraction operation by using XOR
      EOR R18, R20
      ANDI R18, 0b00011111  ; masking the value in R18, remainder, to combine with input A

      OR R18, R19           ; ORing R19 with R18 will give the value of R19 holding the 3-bits from the input A followed by the CRC which is in R18 now
      CP R16, R18           ; compare the entered input (R16) and the transmitted one at the end (R18)
      BRNE L1               ; if they are not equal, go to branch L1
      RJMP L2               ; else, no error so branch to L1

      L1:
        SBI PORTD, 1
        RJMP start
      L2:
        SBI PORTD, 0
        RJMP start
```

Debugging Outputs:

For inputs A=101, B=10111110 (no error case 1):



Name	Value
Z Register	0x0000
Status Register	ITHSVNZ
Cycle Counter	100
Frequency	1.000 MHz
Stop Watch	100.00 µs
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00
R09	0x00
R10	0x00
R11	0x00
R12	0x00
R13	0x00
R14	0x00
R15	0x00
R16	0xBE
R17	0xA0
R18	0xBE
R19	0xA0
R20	0x35
R21	0x00
R22	0x00

```
OUT DDRA, R16          ; A - input (3-bit data input sequence)
OUT DDRB, R16          ; B - input (8-bit binary input sequence)

LDI R16, ONES
OUT DDRD, R16          ; D - output (error detection led)

IN R17, PINA
IN R16, PINB
LDI R19, ZEROS
ADD R19, R17           ; R19 holding the A inout value to combine with the CRC at t
LDI R18, ZEROS
ADD R18, R16           ; R18 holding the B input to perform the necessary operatior

ANDI R19, 0b11100000  ; masking the 3-input input in R19
LDI R20, 0b00110101   ; CRC polynomial given (X5+X4+X2+1) filled with 2 0's at the
EOR R18, R20          ; subtraction operation by using XOR
EOR R18, R20
ANDI R18, 0b00011111  ; masking the value in R18, remainder, to combine with input

OR R18, R19           ; ORing R19 with R18 will give the value of R19 holding the
CP R16, R18           ; compare the entered input (R16) and the transmitted one at
BRNE L1               ; if they are not equal, go to branch L1
RJMP L2               ; else, no error so branch to L1

L1:
  SBI PORTD, 1
  RJMP start
L2:
  SBI PORTD, 0
  RJMP start
```

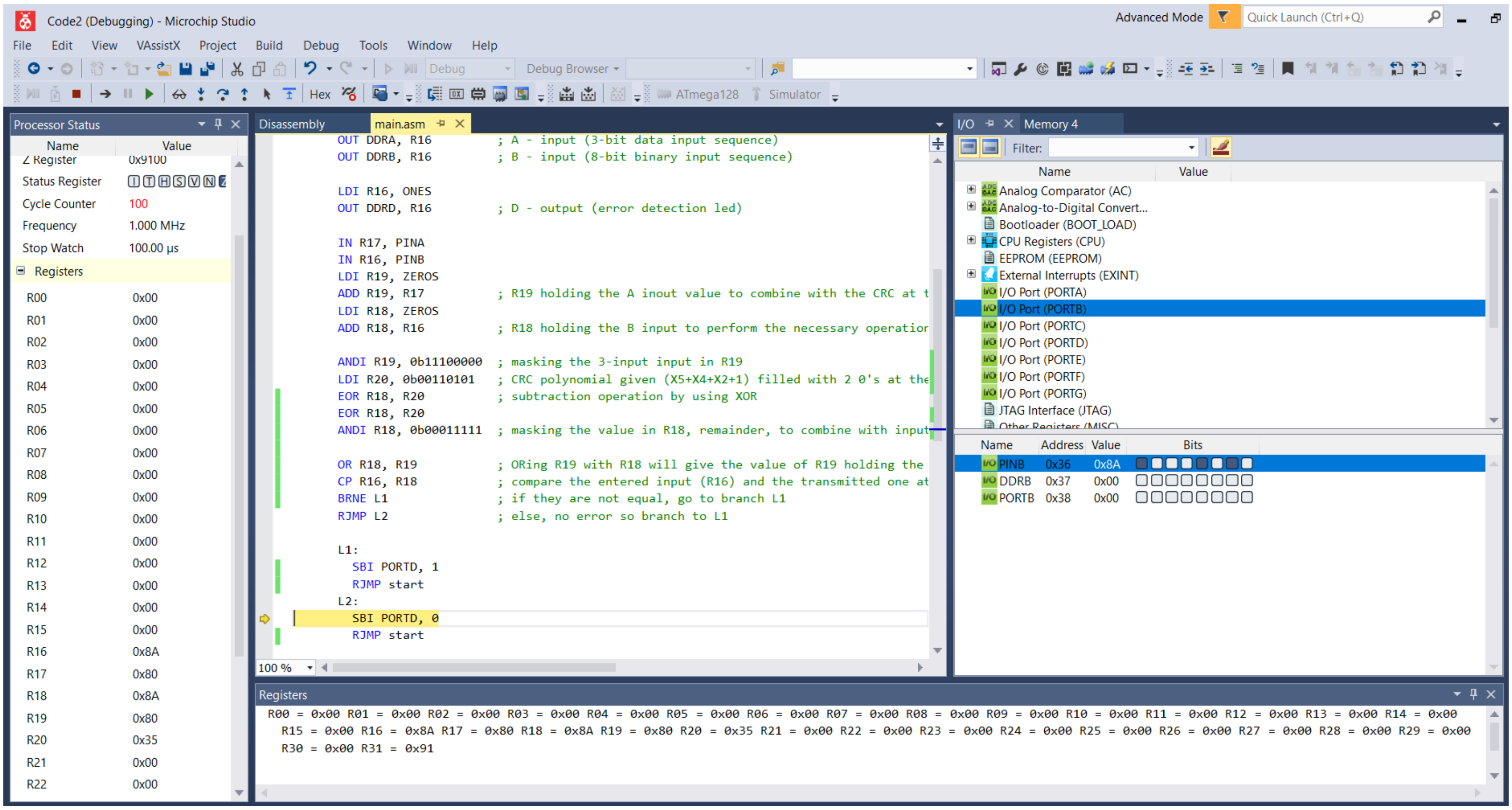
Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
Bootloader (BOOT_LOAD)	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTA)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
I/O Port (PORTE)	
I/O Port (PORTF)	
I/O Port (PORTG)	
JTAG Interface (JTAG)	
Other Registers (MISC)	

Name	Address	Value	Bits
PINB	0x36	0xBE	■□■■■■■□
DDRB	0x37	0x00	□□□□□□□□
PORTB	0x38	0x00	□□□□□□□□

```
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00 R14 = 0x00
  R15 = 0x00 R16 = 0xBE R17 = 0xA0 R18 = 0xBE R19 = 0xA0 R20 = 0x35 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00 R28 = 0x00 R29 = 0x00
  R30 = 0x00 R31 = 0x00
```

Register R16 is holding the value entered at the beginning as inout B, and R18 is holding the transmitted value at the end after appending the remainder after the 3-bit input A. It can be seen from the figure that the output is correct. After the comparison, as those values are equal, the probram branches to L2 so the led will be turned off which indicates that there is no error.

For inputs A=100, B=10001010 (no error case 2):



Name	Value
Z Register	0x9100
Status Register	I T H S V N Z
Cycle Counter	100
Frequency	1.000 MHz
Stop Watch	100.00 µs
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00
R09	0x00
R10	0x00
R11	0x00
R12	0x00
R13	0x00
R14	0x00
R15	0x00
R16	0x8A
R17	0x80
R18	0x8A
R19	0x80
R20	0x35
R21	0x00
R22	0x00

```
        OUT DDRA, R16          ; A - input (3-bit data input sequence)
        OUT DDRB, R16          ; B - input (8-bit binary input sequence)

        LDI R16, ONES
        OUT DDRD, R16          ; D - output (error detection led)

        IN R17, PINA
        IN R16, PINB
        LDI R19, ZEROS
        ADD R19, R17           ; R19 holding the A inout value to combine with the CRC at t
        LDI R18, ZEROS
        ADD R18, R16           ; R18 holding the B input to perform the necessary operatior

        ANDI R19, 0b11100000   ; masking the 3-input input in R19
        LDI R20, 0b00110101    ; CRC polynomial given (X5+X4+X2+1) filled with 2 0's at the
        EOR R18, R20           ; subtraction operation by using XOR
        EOR R18, R20
        ANDI R18, 0b00011111   ; masking the value in R18, remainder, to combine with input

        OR R18, R19            ; ORing R19 with R18 will give the value of R19 holding the
        CP R16, R18            ; compare the entered input (R16) and the transmitted one at
        BRNE L1                ; if they are not equal, go to branch L1
        RJMP L2                ; else, no error so branch to L1

        L1:
          SBI PORTD, 1
          RJMP start
        L2:
          SBI PORTD, 0
          RJMP start
```

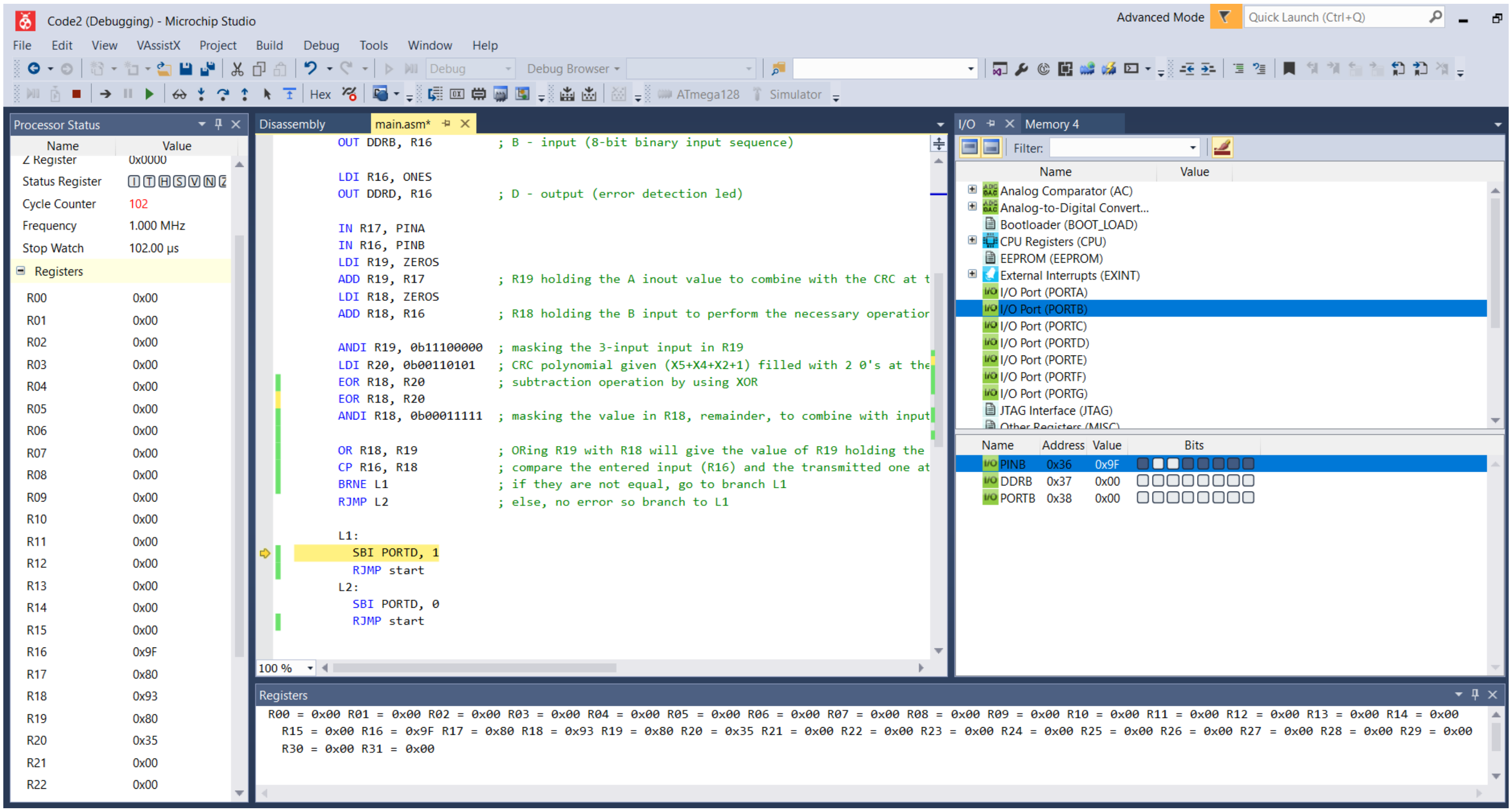
Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
Bootloader (BOOT_LOAD)	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTA)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
I/O Port (PORTE)	
I/O Port (PORTF)	
I/O Port (PORTG)	
JTAG Interface (JTAG)	
Other Registers (MISC)	

Name	Address	Value	Bits
PINB	0x36	0x8A	■□□□■□■□
DDRB	0x37	0x00	□□□□□□□□
PORTB	0x38	0x00	□□□□□□□□

```
 R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00 R14 = 0x00
   R15 = 0x00 R16 = 0x8A R17 = 0x80 R18 = 0x8A R19 = 0x80 R20 = 0x35 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00 R28 = 0x00 R29 = 0x00
   R30 = 0x00 R31 = 0x91
```

Same steps are taken and the result is no error in this case as the previous case. After generating the division and making necessary appends, the result at the end is what is expected, which is no error. So it branches to the L2, meaning that the error detection led will not be turned on.

For inputs A=100, B=10011111 (error case):



Name	Value
Z Register	0x0000
Status Register	I T H S V N Z
Cycle Counter	102
Frequency	1.000 MHz
Stop Watch	102.00 µs
Registers	
R00	0x00
R01	0x00
R02	0x00
R03	0x00
R04	0x00
R05	0x00
R06	0x00
R07	0x00
R08	0x00
R09	0x00
R10	0x00
R11	0x00
R12	0x00
R13	0x00
R14	0x00
R15	0x00
R16	0x9F
R17	0x80
R18	0x93
R19	0x80
R20	0x35
R21	0x00
R22	0x00

```
        OUT DDRB, R16           ; B - input (8-bit binary input sequence)

        LDI R16, ONES
        OUT DDRD, R16           ; D - output (error detection led)

        IN R17, PINA
        IN R16, PINB
        LDI R19, ZEROS
        ADD R19, R17            ; R19 holding the A inout value to combine with the CRC at t
        LDI R18, ZEROS
        ADD R18, R16            ; R18 holding the B input to perform the necessary operatior

        ANDI R19, 0b11100000  ; masking the 3-input input in R19
        LDI R20, 0b00110101   ; CRC polynomial given (X5+X4+X2+1) filled with 2 0's at the
        EOR R18, R20          ; subtraction operation by using XOR
        EOR R18, R20
        ANDI R18, 0b00011111  ; masking the value in R18, remainder, to combine with input

        OR R18, R19             ; ORing R19 with R18 will give the value of R19 holding the
        CP R16, R18             ; compare the entered input (R16) and the transmitted one at
        BRNE L1                 ; if they are not equal, go to branch L1
        RJMP L2                 ; else, no error so branch to L1

        L1:
          SBI PORTD, 1
          RJMP start
        L2:
          SBI PORTD, 0
          RJMP start
```

Name	Value
Analog Comparator (AC)	
Analog-to-Digital Convert...	
Bootloader (BOOT_LOAD)	
CPU Registers (CPU)	
EEPROM (EEPROM)	
External Interrupts (EXINT)	
I/O Port (PORTA)	
I/O Port (PORTB)	
I/O Port (PORTC)	
I/O Port (PORTD)	
I/O Port (PORTE)	
I/O Port (PORTF)	
I/O Port (PORTG)	
JTAG Interface (JTAG)	
Other Registers (MISC)	

Name	Address	Value	Bits
PINB	0x36	0x9F	■□□■■■■■
DDRB	0x37	0x00	□□□□□□□□
PORTB	0x38	0x00	□□□□□□□□

Registers

```
R00 = 0x00 R01 = 0x00 R02 = 0x00 R03 = 0x00 R04 = 0x00 R05 = 0x00 R06 = 0x00 R07 = 0x00 R08 = 0x00 R09 = 0x00 R10 = 0x00 R11 = 0x00 R12 = 0x00 R13 = 0x00 R14 = 0x00
  R15 = 0x00 R16 = 0x9F R17 = 0x80 R18 = 0x93 R19 = 0x80 R20 = 0x35 R21 = 0x00 R22 = 0x00 R23 = 0x00 R24 = 0x00 R25 = 0x00 R26 = 0x00 R27 = 0x00 R28 = 0x00 R29 = 0x00
  R30 = 0x00 R31 = 0x00
```

As seen in the figure, input B is not correct according to the division operation that we will be performing. The transmitted output after the division operation is supposed to give 10001010 so when we compare the value with the input entered, we see that it is an error case as the values are not equal. So the program will branch to the L1 this time, meaning that this is an error case and the error detection led will be turned on.