**Cng 336**

**Lab 4 Prework**

**Umut Yıldırım - 2315612**

**Ulaş Cem Erten - 2256006**

**4.2**

**2.**

USART: ;initialize registers

ldi r16,0x00; all bits are 0

out UCSR0A,r16;

ldi r16,0x18 ; just rxen and txen is 1

out UCSR0B,r16;

ldi r16,0x86 ; rw , ucszn1 and ucszn0 is 1

sts UCSR0C,r16

ldi r16,0x00; ; all bit are 0

sts UBRR0H,r16

ldi r16,0x33; baudrate is 51

out UBRR0L,r16;

ret

**3.**

SENDCHAR:

Loop\_1: sbis UCSR0A, UDRE0 ; wait until udre0 set

rjmp Loop\_1

out UDR0, R16 ; if set send the value to terminal

ret

**4.**

SENDSTR:

ldi XL,LOW(0x0200) ; starting adress is 0x200

ldi XH,HIGH(0x0200) ; same

ldi r18, '$' ; compare bit

Loop\_2: ld r16, X+ ; store x to r16 and increment x

cp r16, r18 ; compare

breq Loop\_3 ; if $ is not entered continue

call SENDCHAR ; if entered send char to terminal

jmp Loop\_2 ; jump until $ entered

Loop\_3: ret ; return

**5.**

RECVCHAR: ; beginning of receive char subroite

Loop\_4: sbis UCSR0A,RXC0 ; wait until rxc0 set

rjmp Loop\_4 ;jump

in r17, UDR0 ; put the char in terminal to r17

ret

**6.**

RECVSTR:; beginning of receive str function

ldi YL,LOW(0x0400) ; starting adress is 0x400

ldi YH,HIGH(0x0400) ; same

ldi r18,'$' ; compare character

Loop\_5: call RECVCHAR ; receive char from terminal

cp r17,r18 ; check if last bit $

breq Loop\_6 ; if yes, end

st Y+, r17 ; else store r17 into y pointer then inc y+

jmp Loop\_5 ; continue

Loop\_6: ldi r20,'\n' ; this is for next line

st Y+,r20 ; store next line

ldi r20,'\r' ; this is also for next line

st Y+,r20 ; same

st Y+,r18 ;add $ too

ret ;return

**7.**

welcome: ; beginning of welcome subroutine

ldi ZL, LOW(MSG1<<1) ; put MSG1 's adress to Z pointer

ldi ZH, HIGH(MSG1<<1) ; same

ldi XL,LOW(0x0200) ; put storing adress to x pointer

ldi XH,HIGH(0x0200) ; same

ldi r18, '$' ; load compare character

Loop\_7: lpm r19, Z+ ; store the char at Z into r19 then inc Z

st X+,r19 ; store the char r19 to X then inc X

cp r19,r18 ; compare r18 and r19

breq Loop\_8 ; if yes then break

jmp Loop\_7 ; else continue

Loop\_8: call SENDSTR ; call sendstr func

call RECVSTR ; call recvstr func

ldi ZL, LOW(MSG2<<1) ; put hello's adress to Z pointer

ldi ZH, HIGH(MSG2<<1) ; same

ldi XH, HIGH(0x200) ; put the name's adress into X pointer

ldi XL, LOW(0x200); same

Loop\_9: lpm r19,Z+ ; Put the char at Z into r19 then inc Z

st X+,r19 ; put the char at r19 into X then inc X

cp r19,r18 ; compare

breq Loop\_10 ;if r19 is $ then break

jmp Loop\_9 ;else continue

Loop\_10: call SENDSTR ; call send str func

ldi XL,LOW(0x0200) ;name is at 0x200

ldi XH,HIGH(0x0200);same

ldi YL,LOW(0x0400);recieved string is at 0x400

ldi YH,HIGH(0x0400);same

cont: ld r20, Y+ ; put the value to r20 then inc Y

st X+,r20 ; load r20 into x then inc x

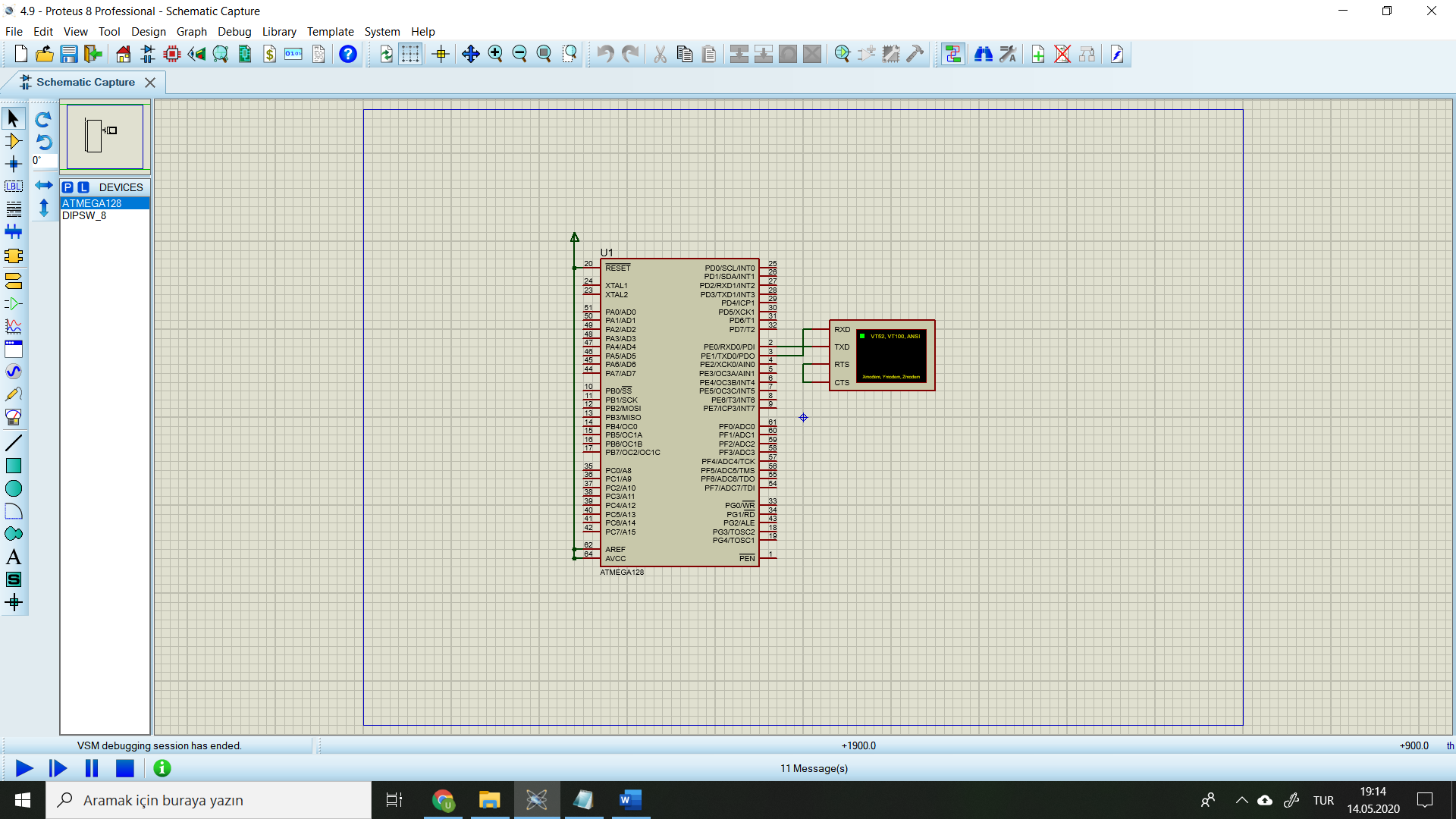
cp r20, r18 ;compare if r20 is $

breq Loop\_11 ; if yes end

jmp cont ; else continue

Loop\_11: call SENDSTR ; send str to terminal

ret ; end

**8)** 

**9)**

.include "m128def.inc"

.ORG 0x0000

.MACRO INITSTACK ; initialize stack

LDI R16,HIGH(RAMEND)

OUT SPH,R16

LDI R16,LOW(RAMEND)

OUT SPL,R16

.ENDMACRO

INITSTACK

call USART ; call usart subroutine

END:call welcome ; call welcome subroutine

rjmp END ; jump and ask again

USART: ;initialize registers

ldi r16,0x00; all bits are 0

out UCSR0A,r16;

ldi r16,0x18 ; just rxen and txen is 1

out UCSR0B,r16;

ldi r16,0x86 ; rw , ucszn1 and ucszn0 is 1

sts UCSR0C,r16

ldi r16,0x00; ; all bit are 0

sts UBRR0H,r16

ldi r16,0x33; baudrate is 51

out UBRR0L,r16;

ret

SENDCHAR:

Loop\_1: sbis UCSR0A, UDRE0 ; wait until udre0 set

rjmp Loop\_1

out UDR0, R16 ; if set send the value to terminal

ret

SENDSTR:

ldi XL,LOW(0x0200) ; starting adress is 0x200

ldi XH,HIGH(0x0200) ; same

ldi r18, '$' ; compare bit

Loop\_2: ld r16, X+ ; store x to r16 and increment x

cp r16, r18 ; compare

breq Loop\_3 ; if $ is not entered continue

call SENDCHAR ; if entered send char to terminal

jmp Loop\_2 ; jump until $ entered

Loop\_3: ret ; return

RECVCHAR: ; beginning of receive char subroite

Loop\_4: sbis UCSR0A,RXC0 ; wait until rxc0 set

rjmp Loop\_4 ;jump

in r17, UDR0 ; put the char in terminal to r17

ret

RECVSTR:; beginning of receive str function

ldi YL,LOW(0x0400) ; starting adress is 0x400

ldi YH,HIGH(0x0400) ; same

ldi r18,'$' ; compare character

Loop\_5: call RECVCHAR ; receive char from terminal

cp r17,r18 ; check if last bit $

breq Loop\_6 ; if yes, end

st Y+, r17 ; else store r17 into y pointer then inc y+

jmp Loop\_5 ; continue

Loop\_6: ldi r20,'\n' ; this is for next line

st Y+,r20 ; store next line

ldi r20,'\r' ; this is also for next line

st Y+,r20 ; same

st Y+,r18 ;add $ too

ret ;return

welcome: ; beginning of welcome subroutine

ldi ZL, LOW(MSG1<<1) ; put MSG1 's adress to Z pointer

ldi ZH, HIGH(MSG1<<1) ; same

ldi XL,LOW(0x0200) ; put storing adress to x pointer

ldi XH,HIGH(0x0200) ; same

ldi r18, '$' ; load compare character

Loop\_7: lpm r19, Z+ ; store the char at Z into r19 then inc Z

st X+,r19 ; store the char r19 to X then inc X

cp r19,r18 ; compare r18 and r19

breq Loop\_8 ; if yes then break

jmp Loop\_7 ; else continue

Loop\_8: call SENDSTR ; call sendstr func

call RECVSTR ; call recvstr func

ldi ZL, LOW(MSG2<<1) ; put hello's adress to Z pointer

ldi ZH, HIGH(MSG2<<1) ; same

ldi XH, HIGH(0x200) ; put the name's adress into X pointer

ldi XL, LOW(0x200); same

Loop\_9: lpm r19,Z+ ; Put the char at Z into r19 then inc Z

st X+,r19 ; put the char at r19 into X then inc X

cp r19,r18 ; compare

breq Loop\_10 ;if r19 is $ then break

jmp Loop\_9 ;else continue

Loop\_10: call SENDSTR ; call send str func

ldi XL,LOW(0x0200) ;name is at 0x200

ldi XH,HIGH(0x0200);same

ldi YL,LOW(0x0400);recieved string is at 0x400

ldi YH,HIGH(0x0400);same

cont: ld r20, Y+ ; put the value to r20 then inc Y

st X+,r20 ; load r20 into x then inc x

cp r20, r18 ;compare if r20 is $

breq Loop\_11 ; if yes end

jmp cont ; else continue

Loop\_11: call SENDSTR ; send str to terminal

ret ; end

.ORG 0x500 ;starting adress

MSG1: .DB "What is your name ?",'\n','\r','$' ; first data

MSG2: .DB "Hello $" ; second data

**10.**

INTEN:

SEI

ret

**11.**

Timer:

LDI R16,0xFF

out ddra,R16

out ddrb,r16

LDI R20,0x0

OUT TCCR1A,R20 ;timer starts from 0;

OUT TCNT1H,R20 ;timer starts from 0;

OUT TCNT1L,R20 ;timer starts from 0;

LDI R20,0x1E

OUT OCR1AH,R20

LDI R20,0x84;prescaler is clk/1024 1/(8\*10^6/(1024))\*x=1 s

OUT OCR1AL,R20 ;loaded will cause interrupt to occur when counter reaches 1e84 since we are using compare match and using a prescaler to slowdown the counter

LDI R20,0x0D

OUT TCCR1B,R20

ldi R21,0

ldi R22,0

LDI R20,(1<<OCIE1A);enable interrupt for compare flag for timre1 A

OUT TIMSK,R20

SEI

ret

**12.**

.include "m128def.inc"

.org 0x0000

rjmp main

.org 0x0018

rjmp IsrTim

main:

ldi r16,high(RAMEND)

out SPH,r16

ldi r16,low(RAMEND)

out SPL,r16

call USART

call INTEN ; enable interrupt

call timer ; call timer

END:call WELCOME

rjmp END

USART:

ldi r16,0x00;

out UCSR0A,r16;

ldi r16,0x18

out UCSR0B,r16;

ldi r16,0x86

sts UCSR0C,r16

ldi r16,0x00;

sts UBRR0H,r16

ldi r16,0x33;

out UBRR0L,r16;

ret

SENDCHAR:

Loop\_1: sbis UCSR0A, UDRE0

rjmp Loop\_1

out UDR0, R16

ret

SENDSTR:

ldi XL,LOW(0x0200)

ldi XH,HIGH(0x0200)

ldi r18, '$'

Loop\_2: ld r16, X+

cp r16, r18

breq Loop\_3

call SENDCHAR

jmp Loop\_2

Loop\_3: ret

RECVCHAR:

Loop\_4: sbis UCSR0A,RXC0

rjmp Loop\_4

in r17, UDR0

ret

RECVSTR:

ldi YL,LOW(0x0400)

ldi YH,HIGH(0x0400)

ldi r18,'$'

Loop\_5: call RECVCHAR

cp r17,r18

breq Loop\_6

st Y+, r17

jmp Loop\_5

Loop\_6: ldi r20,'\n'

st Y+,r20

ldi r20,'\r'

st Y+,r20

st Y+,r18

ret

WELCOME:

ldi ZL, LOW(MSG1<<1)

ldi ZH, HIGH(MSG1<<1)

ldi XL,LOW(0x0200)

ldi XH,HIGH(0x0200)

ldi r18, '$'

Loop\_7: lpm r19, Z+

st X+,r19

cp r19,r18

breq Loop\_8

jmp Loop\_7

Loop\_8: call SENDSTR

call RECVSTR

ldi ZL, LOW(MSG2<<1)

ldi ZH, HIGH(MSG2<<1)

ldi XH, HIGH(0x200)

ldi XL, LOW(0x200)

Loop\_9: lpm r19,Z+

st X+,r19

cp r19,r18

breq Loop\_10

jmp Loop\_9

Loop\_10: call SENDSTR

ldi XL,LOW(0x0200)

ldi XH,HIGH(0x0200)

ldi YL,LOW(0x0400)

ldi YH,HIGH(0x0400)

cont: ld r20, Y+

st X+,r20

cp r20, r18

breq Loop\_11

jmp cont

Loop\_11: call SENDSTR

ret

.ORG 0x500

MSG1: .DB "What is your name ?",'\n','\r','$'

MSG2: .DB "Hello $"

Timer:

LDI R16,0xFF

out ddra,R16

out ddrb,r16

LDI R20,0x0

OUT TCCR1A,R20 ;timer starts from 0;

OUT TCNT1H,R20 ;timer starts from 0;

OUT TCNT1L,R20 ;timer starts from 0;

LDI R20,0x1E

OUT OCR1AH,R20

LDI R20,0x84;prescaler is clk/1024 1/(8\*10^6/(1024))\*x=1 s

OUT OCR1AL,R20 ;loaded will cause interrupt to occur when counter reaches 1e84 since we are using compare match and using a prescaler to slowdown the counter

LDI R20,0x0D

OUT TCCR1B,R20

ldi R21,0

ldi R22,0

LDI R20,(1<<OCIE1A);enable interrupt for compare flag for timre1 A

OUT TIMSK,R20

SEI

ret

IsrTim:;we are holding our values for ports here on r21,r22 and incrementing r21 if when we increment 21 it becomes 0 that means there is an overflow on register and we should increment r22 which shows the upper 8 bits

clc

Inc R21

brne still

inc R22

still:

out portA,R21

out portB,R22

reti

INTEN:

SEI

Ret

**13)**

IsrRec:;receive char

in r18, UDR0 ; copy UDR to R17

st X+,r18

cpi r18,'$'

brne keep\_going

ldi r16,0

out UCSR0B,r16;disable interrupt

keep\_going:reti

**14)**

welcome:

ldi ZL, LOW(MSG1<<1)

ldi ZH, HIGH(MSG1<<1)

ldi XL,LOW(0x0200)

ldi XH,HIGH(0x0200)

ldi r18, '$'

L7: lpm r19, Z+

st X+,r19

cp r19,r18

breq L8

jmp L7

L8: ldi XL,LOW(0x200)

ldi XH,HIGH(0x200)

ldi R16,(1<<TXEN0)|(1<<UDRIE0);enabling interrupt to write to memory

out ucsr0b,r16;

ldi ZL,LOW(MSG2<<1)

ldi ZH,HIGH(MSG2<<1)

ldi XL,LOW(0x300)

ldi XH,HIGH(0x300)

L9: lpm r19,Z+

cpi r19,'$'

breq print

st X+,r19

jmp L9

print: ldi R16,(1<<RXEN0)|(1<<RXCIE0);

ldi XL,LOW(0x300)

ldi XH,HIGH(0x300)

ldi R16,(1<<TXEN0)|(1<<UDRIE0);enable interrupt to sent data

ret

**15)**

.include "m128def.inc"

RJMP MAIN

.ORG 0x24

RJMP IsrRec

.org 0x26

RJMP IsrTr

MAIN:ldi r16,high(RAMEND)

out SPH,r16

ldi r16,low(RAMEND)

out SPL,r16

LDI R16,(1<<UCSZ01)|(1<<UCSZ00); 8 bit data, no parity, 1 stop bit

sts UCSR0C, R16

LDI R16,0x33 ; 9600 baud rate

OUT UBRR0L, R16 ; XTAL = 8 MHz

LDI R16, 0xFF

OUT DDRB, R16

out DDRA, r16 ; set PORTB as output

SEI ; enable interrupts globally

ldi ZL, LOW(MSG1<<1)

ldi ZH, HIGH(MSG1<<1)

ldi ZL, LOW(MSG1<<1);

ldi ZH, HIGH(MSG1<<1);

ldi YL, LOW(0x200);

ldi YH, HIGH(0x200);

call welcome

WAIT: RJMP WAIT ; stay here until a byte arrives

welcome:

ldi ZL, LOW(MSG1<<1)

ldi ZH, HIGH(MSG1<<1)

ldi XL,LOW(0x0200)

ldi XH,HIGH(0x0200)

ldi r18, '$'

L7: lpm r19, Z+

st X+,r19

cp r19,r18

breq L8

jmp L7

L8: ldi XL,LOW(0x200)

ldi XH,HIGH(0x200)

ldi R16,(1<<TXEN0)|(1<<UDRIE0);enabling interrupt to write to memory

out ucsr0b,r16;

ldi ZL,LOW(MSG2<<1)

ldi ZH,HIGH(MSG2<<1)

ldi XL,LOW(0x300)

ldi XH,HIGH(0x300)

L9: lpm r19,Z+

cpi r19,'$'

breq print

st X+,r19

jmp L9

print: ldi R16,(1<<RXEN0)|(1<<RXCIE0);

ldi XL,LOW(0x300)

ldi XH,HIGH(0x300)

ldi R16,(1<<TXEN0)|(1<<UDRIE0);enable interrupt to sent data

ret

IsrRec:;receive char

in r18, UDR0 ; copy UDR to R17

st X+,r18

cpi r18,'$'

brne keep\_going

ldi r16,0

out UCSR0B,r16;disable interrupt

keep\_going:reti

IsrTr:;send to terminal

ld r17,X+

out portb,r17

out UDR0,r17

cpi r17,'$'

brne keep\_trans

ldi r16,0

out ucsr0b,r16;disable interrupt

keep\_trans:RETI

.ORG 0x500

MSG1: .DB "What is your name ?",'\n','\r','$'

MSG2: .DB "Hello $"

;LDI R16,(1<<RXEN0)|(1<<RXCIE0); enable receiver and RXC0 interrupt

;ldi XL,LOW(0x400)

;ldi XH,HIGH(0x400)

;OUT UCSR0B, R16