Embedded Systems Lab Experiment 10

Serial Peripherial Interface in ATmega 32

To program two ATmega 32 microcontroller as a master and a slave for SPI communication between them to control a 7-segment display

Jitbitan Baroi
3-31-2024

Exp 10: Serial Peripherial Interface in AT Mega 32

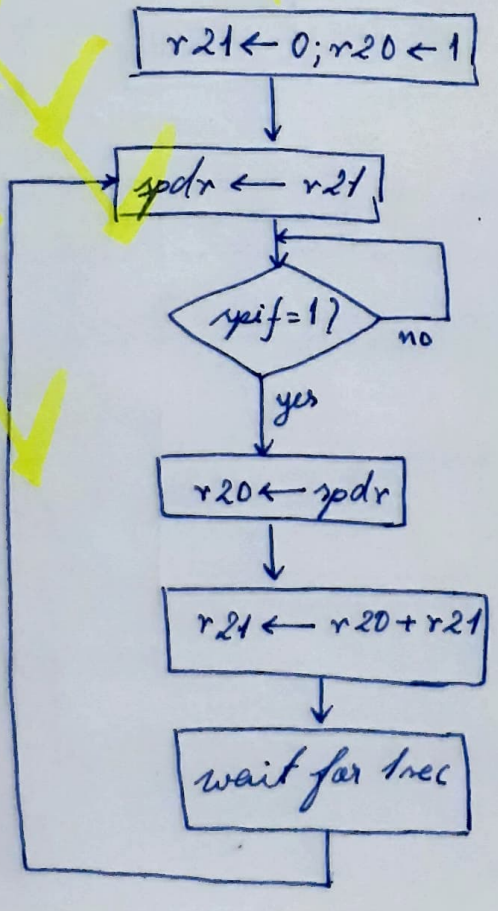
•> **Objective:** To program two AT Mega 32 microcontrollers as a master and slave for SPI communication.

•> **Apparatus reqd.:**

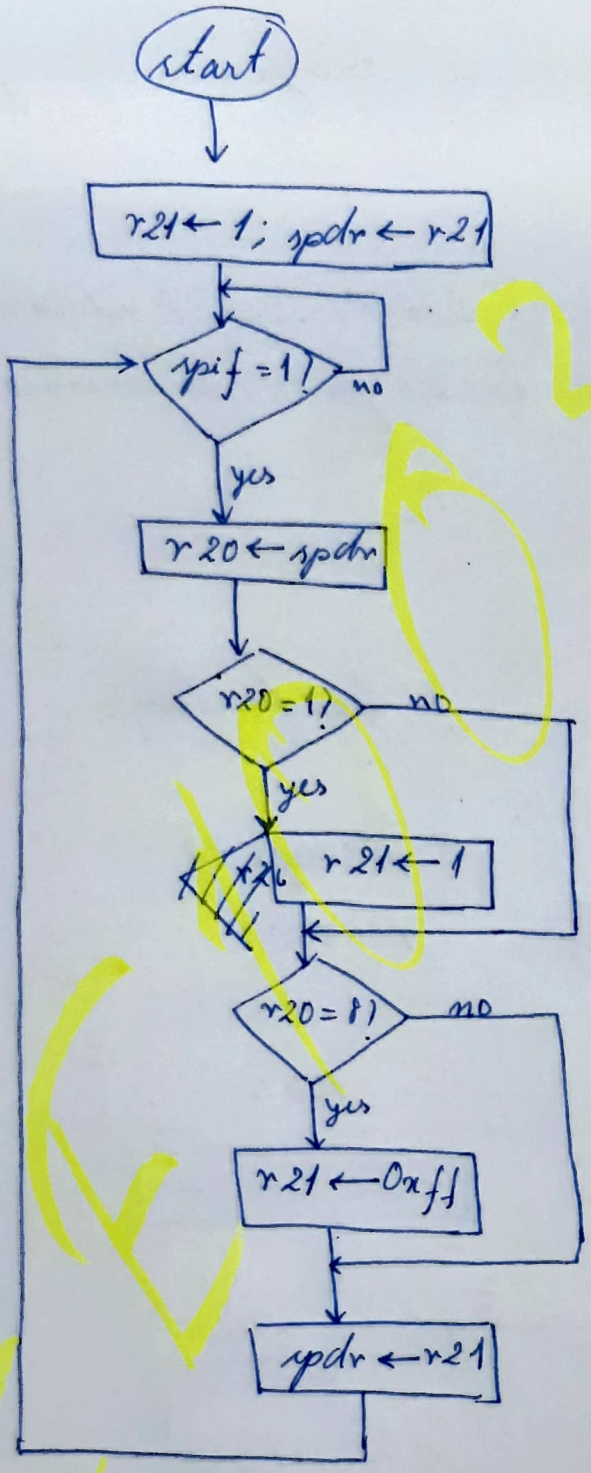
Name	Specification	Quantity
1. µC	AT Mega 32	2
2. 7 segment display	TAR5503	1
3. Jumper wires	—	12
4. Resitors	100 Ω	7

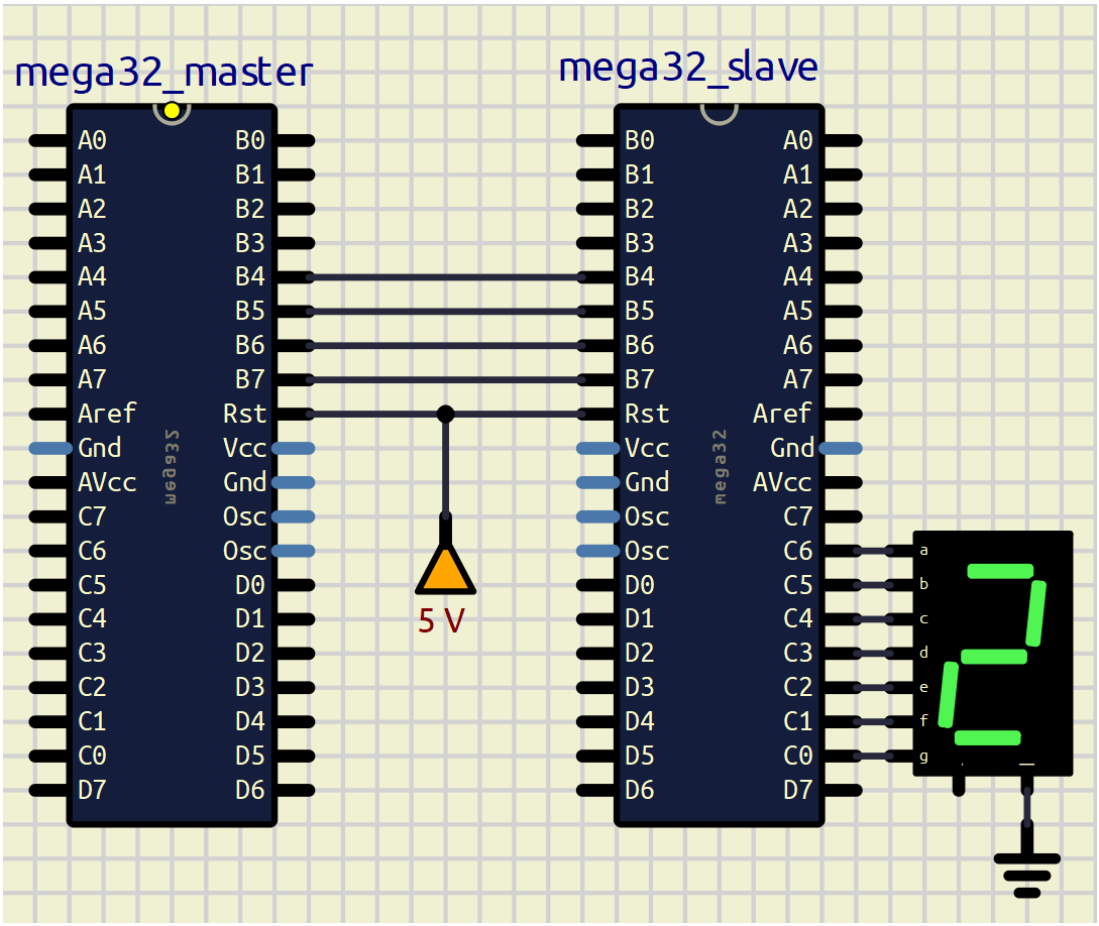
•> **Program flow:**

(i) **master:**



(ii) slave:


start
r21 ← 1; spdr ← r21
spif = 1?
no
yes
r20 ← spdr
r20 = 1?
no
yes
r21 ← 1
r20 = 81
no
yes
r21 ← 0xff
spdr ← r21



Schematic used for SPI Communication with master and slave boards mentioned

•> Code:

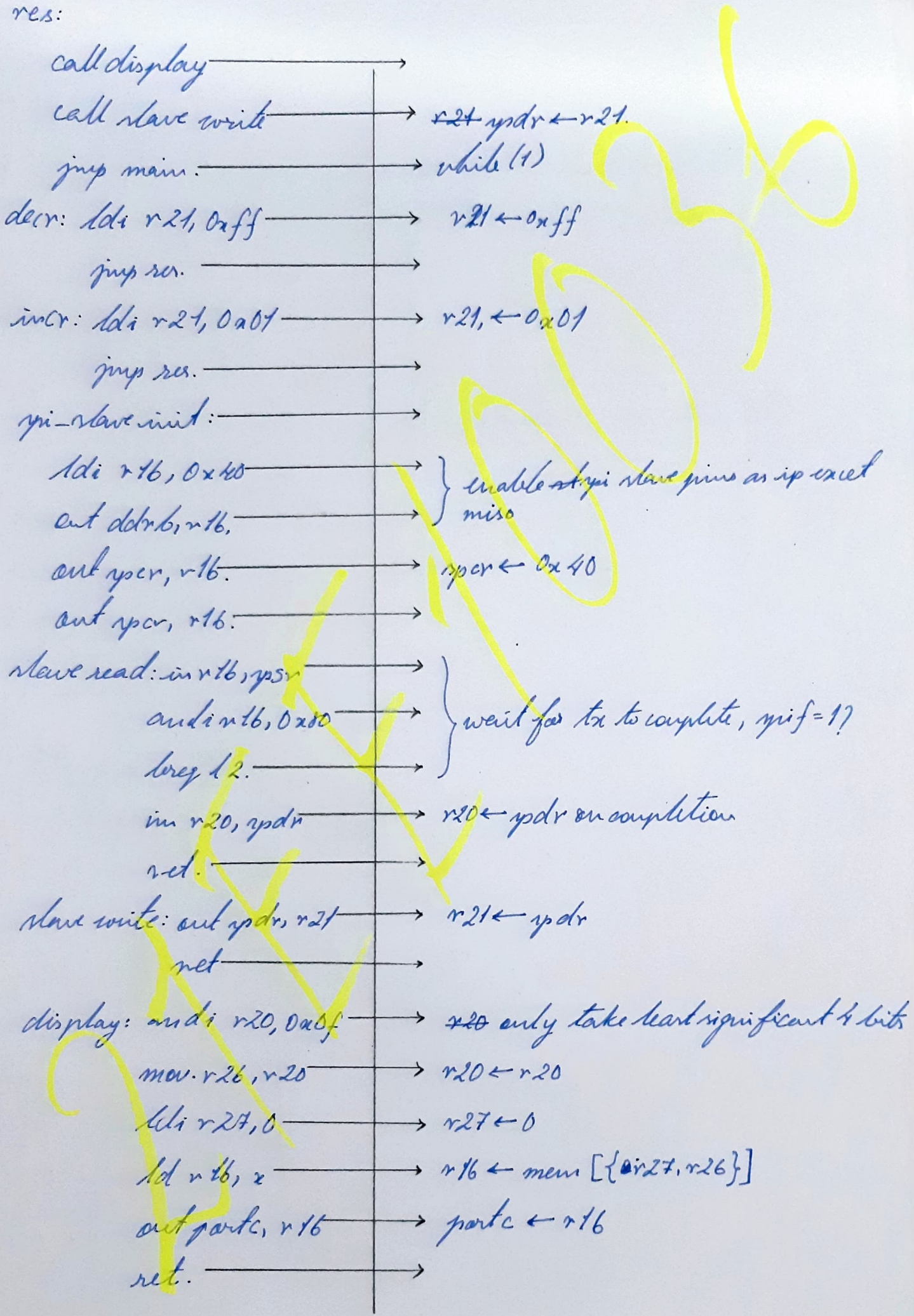
(i) master:

```
// basic imports
call spi_master_init      -> enable spi with proper settings
    ldi r16,0xff          -> } portc as output
    out ddrc,r16          -> }
    ldi r21,0             -> } r20 <- 1, r21 <- 0
    ldi r20,1             -> }
    call delay            -> wait for slave to init properly
main:
    call master_write     -> exchange data
    call master_read      -> r20 <- spdr
    add r21,r20           -> r21 <- r20 + r21
    call delay            -> wait for slave to complete
    jmp main              -> while(1)
spi_master_init:
    ldi r16,0xb0          -> } ip, op pins in portb are set
    out ddrb,r16          -> } according
    sbi portb,pinb4       -> disable SS slave.
    ldi r16,0x51          -> } spcr <- 0x51
    out spcr,r16          -> }
    ret.
master_read:
    in r20,spdr           -> r20 <- spdr
    ret.
```

```
master_write:
    cbi portb, pinb4        → enable slave
    out ~~r24, spdr~~ spdr, r24   → spdr ← r24, begin Tx
l1: in r16, spsr            → } wait for Tx complete, spif=1
    andi r16, 0x80          →
    breq l1                 →
    in r16, spdr            → r16 ← spdr, spif = 0
    sbi portb, pinb4        → disable slave.
    ret.
//a delay block.            → a delay fn from lab1, led blink(1s,

(ii) Slave:
//basic_inports             →
//led 7 segment codes       →
call spi_slave_init         → enable spi in slave mode.
ldi r16, 0xff               → }
out ddrc, r16.              → } portc as output
ldi r21, 1                  → r21 ← 1
call slave_write            → spdr ← r24

main:                       →
    call slave_read         → wait for Tx completion.
    cpi r20, 1              → } if r20 = 1? jmp to incr
    breq incr               →
    cpi r20, 8              → } if r20 = 8? jmp to decr.
    breq decr.              →
```


res:
call display
call slave write
jmp main.
while (1)
decr: ldi r21, 0xff
r21 ← 0xff
jmp res.
incr: ldi r21, 0x01
r21, ← 0x01
jmp res.
spi_slave init:
ldi r16, 0x40
out ddrb, r16.
enable spi slave pins as ip except miso
out spcr, r16.
spcr ← 0x40
out spcr, r16:
slave read: in r16, spsr
andi r16, 0x80
breq l2.
wait for tx to complete, spif = 1?
in r20, spdr
r20 ← spdr on completion
ret.
slave write: out spdr, r21
r21 ← spdr
ret
display: andi r20, 0x0f
r20 only take least significant 4 bits
mov. r26, r20
r20 ← r20
ldi r27, 0
r27 ← 0
ld r16, x
r16 ← mem [{r27, r26}]
out portc, r16
portc ← r16
ret.