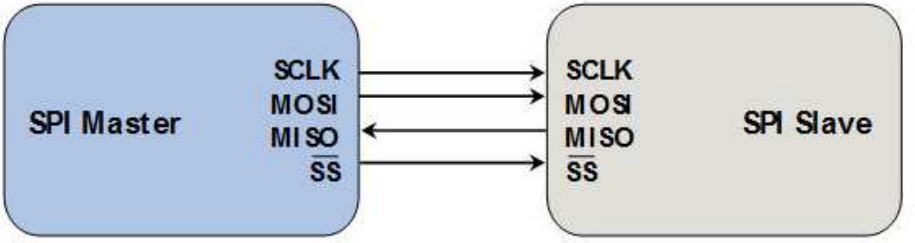
Rad sa SPI fleš memorijom

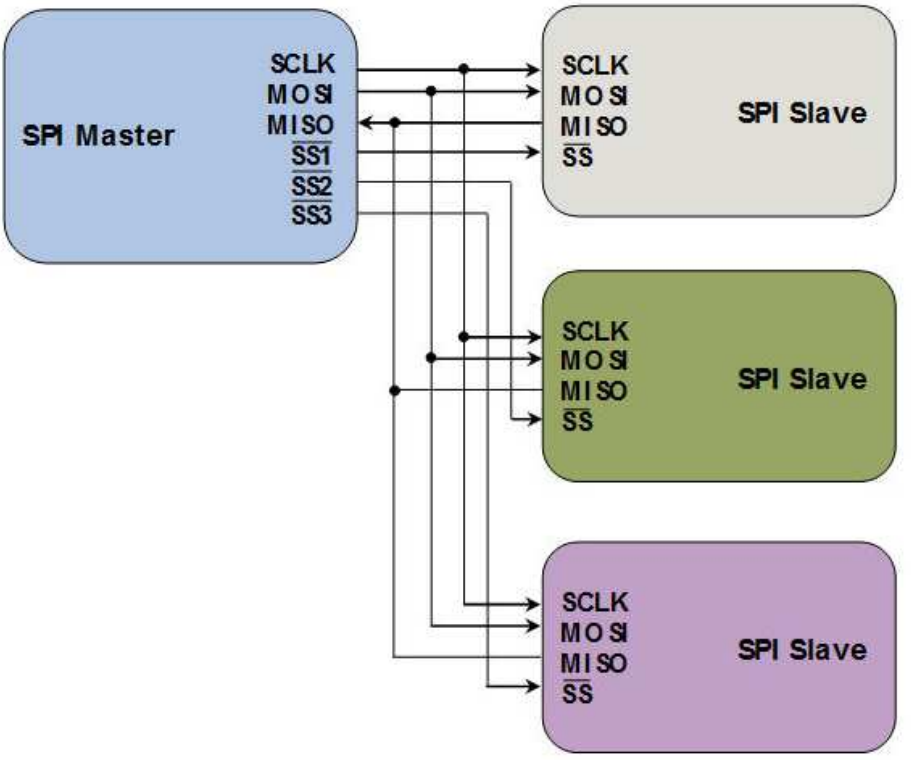
1. SPI komunikacija

SPI (eng. *The **S**erial **P**eripheral **I**nterface Bus*) predstavlja vid sinhrone serijske komunikacije koju je razvila Motorola [27]. SPI je *full duplex* komunikacija između master i *slave* uređaja. Za ovaj vid komunikacije potrebne su četiri fizičke linije SCLK (***S**erial **CL**oc**K***), MOSI (***M**aster **O**ut **S**lave **I**n*), MISO (***M**aster **I**n **S**lave **O**ut*) i CS (***C**hip **S**elect*) ili SS (***S**lave **S**elect*). SPI komunikacija između jednog mastera i jednog slejva je prikazana na slici 1. Tutorijal u vezi SPI je moguće pročitati u [28].



Slika 1. Povezivanje jednog master i jednog slave uređaja

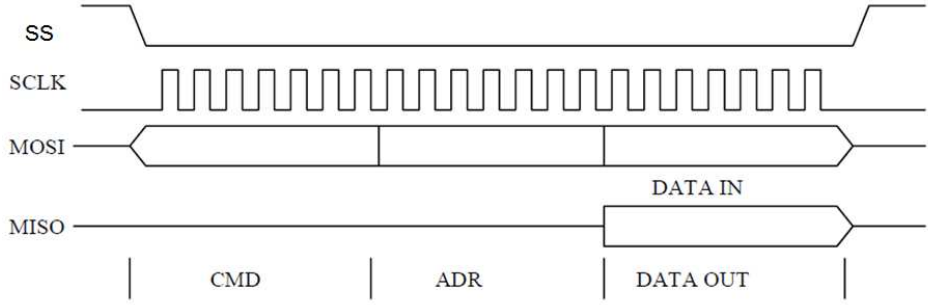
Moguće je povezati i više *slave* uređaja ali u tom slučaju je potrebno da na strani mastera postoji SS ili CS (*slave* ili *chip select*) za svaki *slave* uređaj, što je prikazano na slici 2.



Slika 2. Povezivanje jednog master i tri slave uređaja

Komunikaciju inicira master uređaj tako što spusti SS signal na logičku nulu, a zatim sledi prenos podataka preko MOSI linije. Ukoliko master uređaj ne očekuje odgovor od periferije onda će odmah nakon poslatog podatka signal SS vratiti na logičku jedinicu, a ukoliko periferija treba da odgovori ona će to učiniti preko MISO linije pa će nakon toga master uređaj SS postaviti na jedinicu. Prenos podataka preko MOSI i MISO linija je sinhron sa taktom SCLK.

U nastavku na slici 3 prikazan je primer komunikacije sa memorijom. CMD je komanda, a ADR je adresa sa koje se čita (upisuje). Ukoliko se upisuje u memoriju tada će master uređaj zanemariti ono što dolazi kao DATA OUT preko MISO linije, dok ako je upitanju čitanje iz memorije tada će memorija zanemariti DATA IN paket sa MOSI linije. CMD, ADR, DATA IN i DATA OUT su u paketima veličine po 8 bita.



Slika 3. Primer SPI komunikacije

Prednosti i mane u odnosu na druge serijske protokole

U nastavku su prikazane neke od prednosti i mana SPI u odnosu na druge načine serijske komunikacije (pre svega u odnosu na I2C).

Prednosti SPI su:

- full-duplex komunikacija
- veći protok podataka nego kod I2C magistrale
- nije ograničen na prenos 8-bitnih reči u slučaju bit prenosa
- proizvoljan izbor veličine poruke, sadržaja i namene
- jednostavan hardver
- manja potrošnje energije u odnosu na I2C usled manjeg broja kola
- slejv uređaj koristi takt glavnog uređaja nije mu potreban precizan oscilator
- primopredajnik nije potreban
- najviše jedna jedinstvena magistrala po uređaju, ostale su deljene

Mane SPI su:

- zahteva više pinova od I2C sprege
- nema unutrašnjeg adresiranja
- nema hardverske kontrole toka podataka
- nema nikakve potvrde od strane slejv uređaja
- sistemi sa više uređaja su retki i nezgrapni i uglavnom su ograničeni na jedan slejv
- bez formalnog standarda, potvrda usklađenosti nije moguća
- podržava samo kratke udaljenosti za prenos podatak za razliku od RS-232, RS-485 ili CAN standarda

2. Fleš memorije

U ovoj vežbi će biti korišćena fleš memorija AT45DB041D kompanije Atmel [29]. U pitanju je 4Mb memorija koja ima mogućnost rada na nivou sektora, blokova i strana. Neophodno je dobro proučiti *datasheet* ove memorije da bi bio jasan princip upisa/čitanja.

3. Podešavanje SPI na RPi

Prvo je potrebno aktivirati SPI magistralu pozivom u terminalu:

```
sudo raspi-config
```

U *Advanced options*, odabrati SPI. Potom odabrati da interfejs bude aktivan kao i da drajver (modul) bude učitan biranjem *Yes*, *Ok* i *Finish*.

Nakon toga podesiti učitavanje modula pozivom:

```
sudo nano /etc/modules
```

i dodavanjem:

```
spi-dev
```

Potrebno je uraditi i sledeću izmenu u `/boot/config.txt` datoteci na sledeći način:

```
sudo nano /boot/config.txt
```

Na kraju datoteke dodati liniju:

```
dtparam=spi=on
```

Uraditi restart RPi-a.

4. Komunikacija RPi sa AT45DB041D fleš memorijom

U narednom primeru je dato programsko rešenje napisano u programskom jeziku C upotrebom BCM2835 biblioteke za realizaciju SPI komunikacije između RPi i fleš memorije. Izled pločice koju treba koristiti dat je na slici 4.

Slika 4. Izgled pločice sa fleš memorijom

```
#include <bcm2835.h>
#include <stdio.h>

//za BCM biblioteku to je pin 8
#define PIN RPI_V2_GPIO_P1_24

void ee_write(unsigned int BufferOffset,unsigned char
data)
{
     bcm2835_gpio_write(PIN, LOW);
     bcm2835_spi_transfer(0x84);//komanda write buffer
     delayMicroseconds(6);
     bcm2835_spi_transfer(0xff);
     delayMicroseconds(12);
     bcm2835_spi_transfer((unsigned char)
                          (BufferOffset>>8));
     delayMicroseconds(12);
     bcm2835_spi_transfer((unsigned char)
                           BufferOffset);
     delayMicroseconds(12);
     bcm2835_spi_transfer(data);
     delayMicroseconds(12);
     bcm2835_gpio_write(PIN, HIGH);
}
```

```
unsigned char ee_read(unsigned int BufferOffset)
{
     unsigned char temp;

     bcm2835_gpio_write(PIN, LOW);
     delayMicroseconds(12);
     bcm2835_spi_transfer(0xD4); //komanda read buffer
     delayMicroseconds(12);
     bcm2835_spi_transfer(0xff);
     delayMicroseconds(12);
     bcm2835_spi_transfer((unsigned char)
                          (BufferOffset>>8));
     delayMicroseconds(12);
     bcm2835_spi_transfer((unsigned char)BufferOffset);
     delayMicroseconds(12);

     temp=bcm2835_spi_transfer(0xff);
     delayMicroseconds(12);
     bcm2835_gpio_write(PIN, HIGH);
     delayMicroseconds(12);
     return temp;
}

int main(int argc, char **argv)
{
     unsigned char num;
     if (!bcm2835_init())
        return 1;

     bcm2835_spi_begin();

bcm2835_spi_setBitOrder(BCM2835_SPI_BIT_ORDER_MSBFIRST);
bcm2835_spi_setDataMode(BCM2835_SPI_MODE3);
bcm2835_spi_setClockDivider(BCM2835_SPI_CLOCK_DIVIDER_256)
;
bcm2835_gpio_fsel(PIN, BCM2835_GPIO_FSEL_OUTP);
//bcm2835_spi_chipSelect(BCM2835_SPI_CS0);
bcm2835_gpio_write(PIN, HIGH);

     //citanje Device ID informacija -
     //strana 26 u datasheet-u
     char buf[] = {0x9F, 0, 0};
```

```
     // svaka SPI transakcija pocinje sa CS -> 0
     bcm2835_gpio_write(PIN, LOW);
     delayMicroseconds(12);
     bcm2835_spi_transfern(buf, sizeof(buf));

     //svaka SPI transakcija zavrsava sa CS -> 1
     bcm2835_gpio_write(PIN, HIGH);
     delayMicroseconds(12);
     printf("Manufacturer ID: 0x%02X\n", buf[1]);
               // 0x1F kod od Atmel-a
     printf("Device ID: 0x%02X\n", buf[2]);
               // 0x24 za DataFlash od 4Mb

     printf("Pocetak   upisa   255   bajtova   u   fles
memoriju: molim sacekajte\n");

     for(num=0;num<255;num++)
     {
          ee_write((unsigned int)num,num);
          delayMicroseconds(75);
     }

     printf("Pocetak citanja 255 bajtova iz fles
               memorije:\n");
     if(ee_read((unsigned int)1)!=1)
     {
          printf("problem sa komunikacijom sa
                    at45db fles memorijom\n");
     }
     else
          for(num=0;num<255;num++)
          {
     printf("%02X  ", ee_read((unsigned int)num));
          bcm2835_delay(5);
          }

    printf("\nKraj testa!\n");
    bcm2835_spi_end();
    bcm2835_close();
    return 0;
}
```