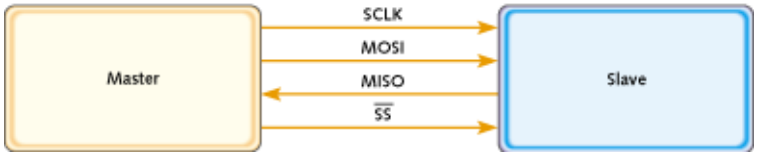
Laboratorul 6
Serial Peripheral Interface

1. Introducere

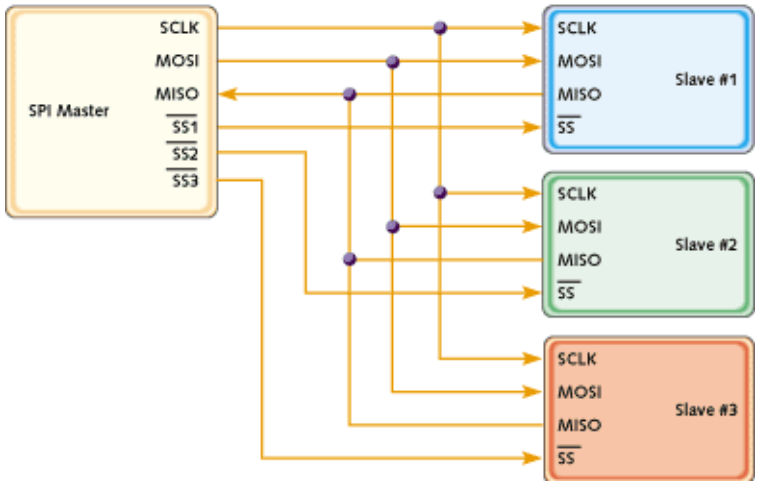
In laboratoarele precedente am vazut la lucru doua tipuri de comunicatie seriala: RS-232 si I2C. In acest laborator ne vom familiariza cu un al treilea astfel de standard: **Serial Peripheral Interface Bus (SPI)**. SPI este un standard sincron (ca si I2C) dezvoltat de Motorola ce opereaza in mod full-duplex (transferul de date are loc in ambele directii simultan). Device-urile comunica folosind o relatie de tipul master/multi slave (nu sunt suportati mai multi masteri) master-ul fiind cel care initiaza frame-urile de date.

SPI se mai numeste si "four wire" serial bus pentru al deosebi de celelalte standarde ce folosesc 1, 2 sau 3 fire. Sa vedem care sunt cele 4 fire (vezi prima figura):

SCLK — Serial Clock (output from master)
MOSI/SIMO — Master Output, Slave Input (output from master)
MISO/SOMI — Master Input, Slave Output (output from slave)
SS — Slave Select (active low; output from master)



Observam ca de data aceasta avem doua fire pe care se transmit date (MOSI si MISO). De asemenea, slave-ul este selectat direct prin intermediul semnalului SS, pentru fiecare slave fiind nevoie de inca un fir de selectie (vezi cea de-a doua figura).



2. SPI vs I2C

Ambele standarde sunt folosite cu succes pentru comunicatia cu periferice incete ce sunt accesate intermitent (EEPROM-urile si ceasurile de timp real sunt exemple de astfel de device-uri). Totusi, SPI se muleaza mai bine ca I2C pe aplicatiile care folosesc stream-uri de date (spre deosebire de aplicatiile

unde se citesc/scriu diverse locatii din slave device). Un exemplu de aplicatie ce foloseste stream-uri este comunicatia dintre micropocesoare sau DSP-uri (digital signal processors).

SPI poate atinge rate de transfer semnificativ mai mari decat I2C - interfetele SPI putand functiona la zeci de MHz. SPI este eficient mai ales in aplicatii ce ii folosesc capacitatea de a realiza conexiuni full duplex, ca de exemplu comunicarea dintre un "codec" (coder-decoder) si un DSP, ce presupune trimiterea de sample-uri in ambele directii.

Datorita faptului ca nu exista suport built-in pentru adresarea device-urilor, SPI necesita mai mult efort si mai multe resurse hardware decat I2C cand avem mai multi slave. Pe de alta parte SPI este mai simplu si mai eficient in aplicatii point-to-point (single master, single slave) din acelasi motive: lipsa adresarii device-urilor presupune mai putin overhead.

Sumarizand, putem enumera urmatoarele avantaje ale SPI:

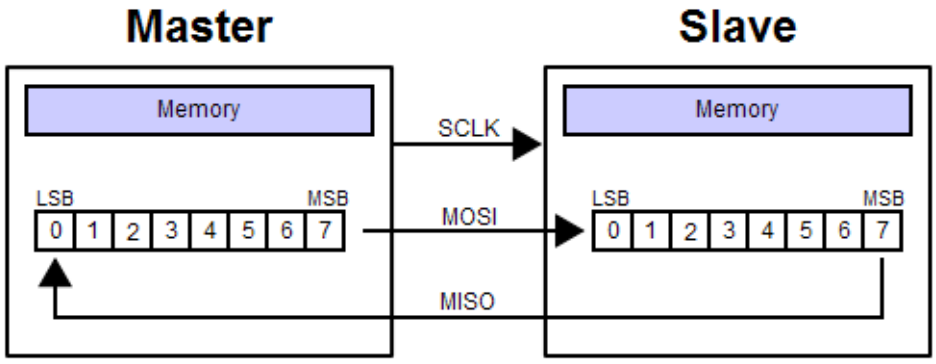
Comunicatie full-duplex
Throughput semnificativ mai mare
Flexibilitate pentru bitii transferati:
    nu exista limitarea la cuvinte de 8-bit
    se pot trimite mesaje de orice lungime
Se interfateaza usor, usurinta in folosire
Mai eficient in comunicarea point-to-point

Dintre dezavantaje amintim:

Foloseste mai multe fire
Nu are suport pentru adresarea devicerurilor; este necesar cate un semnal de Slave Select pentru fiecare slave
Nu exista flow control si slave acknowledgement (master poate "vorbi in gol" fara sa stie).
Suporta numai un singur master
Functioneaza pe distante mici spre deosebire de RS-232 sau RS-485

3. Mod de functionare

Pentru a porni comunicatia masterul trebuie sa seteze ceasul la o frecventa cel mult egala cu frecventa suportata de slave.
Masterul selecteaza apoi cipul slave dorit, punand 0 pe linia SS spre acesta.
In timpul unui ciclu SPI, transmisia este full-duplex:
    masterul trimite un bit pe linia MOSI, iar slave-ul il citeste de pe aceeasi linie;
    slave-ul trimite un bit pe linia MISO, iar master-ul il citeste de pe aceeasi linie;
Comunicatia pe SPI implica de obicei existenta a doi registri de shiftare (Shift Registers), unul in master si unul in slave, conectati in ciclu:



De obicei primul bit shiftat pe liniile de MISO/MOSI este bitul cel mai semnificativ, in timp ce un nou bit este adaugat pe pozitia cea mai putin semnificativa din registru.
Dupa ce intregul cuvant a fost trimis, prin shiftare, masterul si slave-ul au interschimbat valorile

din cei doi registri de shift.
Daca mai exista date de transmis, procesul este reluat.
Cand nu mai exista date de transmis, masterul intrerupe generarea ceasului si, in general, pune 1 pe linia de SS asociata slave-ului.
Cipurile slave care nu au fost selectate vor ignora semnalele de pe SCK si MOSI si nu vor genera nimic pe MISO.
Masterul poate selecta doar un singur slave la un moment dat.

4. SPI - ATMega16

Microcontroller ATMega16 pune la dispozitia utilizatorilor sai si interfata SPI putand functiona atat ca master cat si ca slave. Dupa cum deja ne-am obisnuit, pentru configurarea si utilizarea unei interfete lucram cu registri: de control, de stare si de date. Pentru interfata SPI implementata in ATMega16 avem urmatorii registri:

SPCR - Control Register - in acest registru dam enable la SPI, activam intreruperile asociate, setam modul de functionare a microcontrolerului (master sau slave) si ceasul.
SPSR - Status Register - cel mai important bit din acest registru este SPIF, care semnalizeaza cand s-a terminat o transmisie.
SPDR - Data Register - acesta este registrul din care citim, sau in care scriem, datele pe care le-am primit, sau pe care vrem sa le trimitem.

5. Specificatii ENC28J60

ENC28J60 este un controller Ethernet cu interfata SPI. Este special proiectat pentru a fi conectat prin SPI la un controller, oferind acestuia o interfata Ehernet. ENC28J60 indeplineste specificatiile IEEE 802.3 si ofera urmatoarele servicii:

packet filtering
throughput mare pentru date (datorita modulului DMA incorporat)
calculul sumei de control asistat de hardware.

ENC28J60 incorporeaza urmatoarele module:

Interfata SPI - canal de comunicatie cu controllerul gazda.
Registri de control - controlul si monitorizarea ENC28J60.
Memorie Ram dual port - buffer pentru pachetele receptionate si trimise.
Unitate de arbitru - regleaza cererile de access ale DMA-ului.
O interfata pentru magistrala - interpreteaza comenzi si date venite prin SPI.
MAC - conform IEEE 802.3.
PHY - modul pentru nivelul fizic.

5.1. Organizarea memoriei

Memoria lui ENC28J60 este implementata ca RAM static si se imparte in 3 tipuri:

Registri de control
Ethernet buffer
Registri PH

5.1.1. Registri de control

Acestia asigura interfata principala intre controllerul gazda si logica controllerului Ethernet. Scriere si citirea acestor registri se poate face in mod direct de controllerul gazda utilizand SPI, acesta avand astfel controlul asupra operatiilor executate de ENC28J60.

Registrii de control sunt in general grupati in registrii de tip ETH, MAC si MII (Media Independent Interface). Ne putem da seama din ce categorie face parte un registru uitandu-ne la prefixul numelui sau.

Memoria unde sunt mapati registrii de control este partitionata in 4 bank-uri (vezi tabelul 3-1, pagina 12 din Data Sheet-ul ENC28J60), selectabile prin intermediul bitilor BSEL1:BSEL0 din registrul ECON1. Fiecare bank are 32 bytes, deci putem adresa registrii in interioul unui bank folosind o adresa de 5 biti. Deoarece registrii EIE, EIR, ESTAT, ECON2 si ECON1 sunt foarte des utilizati, acestia au fost mapati in fiecare bank, pentru a facilita accesul la ei.
Cel mai important registru dintre cei mentionati in tabelul de mai sus este ECON1. Biti importanti din acest registru sunt:

TXRTS: Transmit Request to Send bit

1 = logica de transmisie transmite un pachet
0 = logica de transmisie este idle

RXEN: Receive Enable bit

1 = pachetele care nu au fost filtrate vor fi scrise in bufferul de receive
0 = toate pachetele vor fi ignorate

BSEL1:BSEL0: Bank Select bits

11 = SPI va vea acces la registrii din bank-ul 3
10 = SPI va vea acces la registrii din bank-ul 2
01 = SPI va vea acces la registrii din bank-ul 1
00 = SPI va vea acces la registrii din bank-ul 0

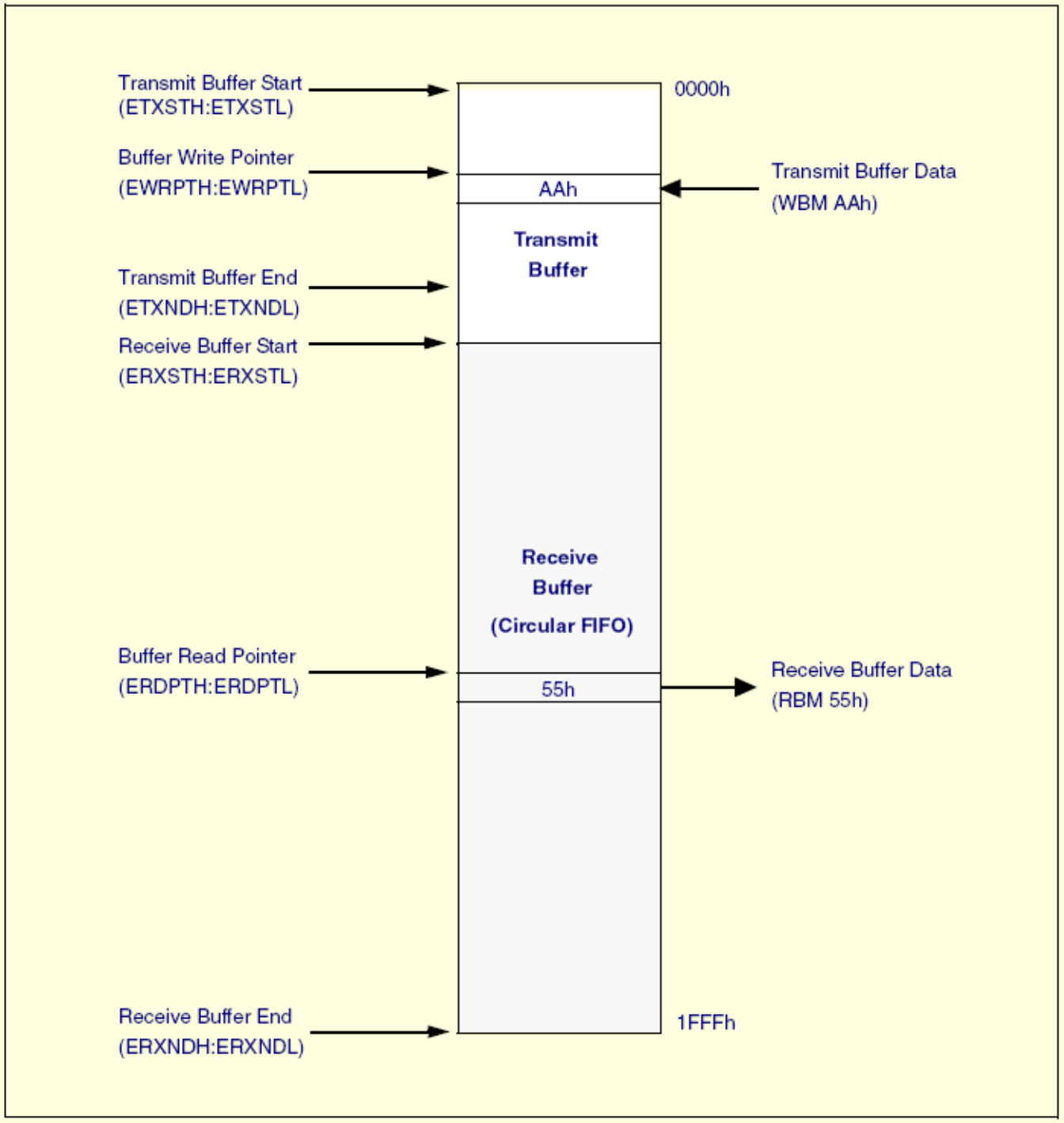
5.1.2. Ethernet buffer

Bufferul Ethernet reprezinta acea zona de memorie in/din care ENC28J60 va scrie/citi packetele receptionate sau transmise. Capacitatea bufferului este de 8Kbytes si este impartit in doua zone distincte pentru pachete receptionate si transmise. Dimensiunea si locatia celor doua zone de memorie sunt configurabile de catre controllerul gazda prin intermediul SPI.

Receiver Buffer - este organizat ca un buffer circular FIFO, controlat de hardware. Perechile de registri ERXSTH:ERXSTL si ERXNDH:ERXNDL sunt folositi ca pointeri spre inceputul si sfarsitul bufferului. Hardware-ul are grija ca pachetele sa fie dispuse in mod circular in spatiul dintre cei doi pointeri.

Transmit Buffer - orice spatiu care nu este folosit pentru receive din cei 8Kbytes, este folosit ca buffer de transmit. Cand controllerul gazda vrea sa trimita un pachet, pune in registrii ETXST si ETXND adresele de inceput si de sfarsit ale respectivului pachet. Hardware-ul NU verifica daca acesti pointeri sunt in spatiul de transmit. Este responsabilitatea controllerului gazda ca adresele furnizate sa fie corecte.

FIGURE 3-2: ETHERNET BUFFER ORGANIZATION

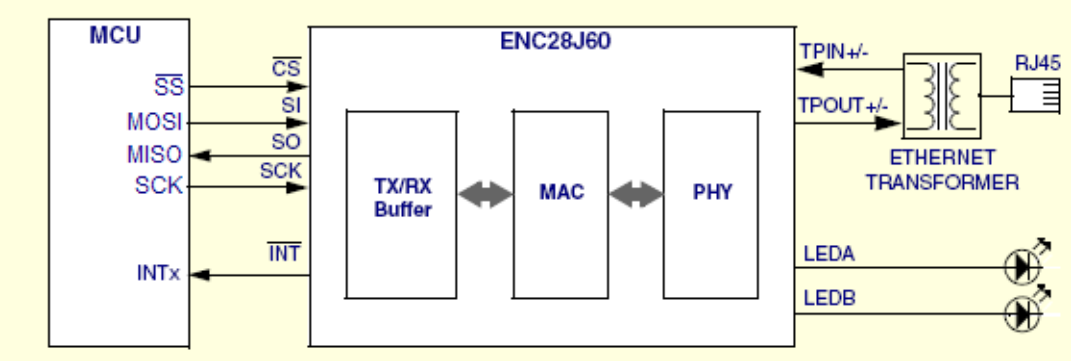


5.1.3. Registri PHY

Permit configurarea si controlul modulului Nivel Fizic. Spre deosebire de registrii de control si de bufferul Ethrenet, registrii PHY nu pot fi modificati in mod direct de controllerul gazda prin SPI. Pentru a avea acces la acestia, controllerul ii poate accesa indirect prin intremediul registrilor de control de tip MII.

5.2. Implementarea SPI la ENC28J60

Controllerul Ethernet ENC28J60 este proiectat pentru a se conecta cu usurinta la portul de SPI al celor mai multe microcontrollere.



Functionarea ENC28J60 depinde in totalitate de comenzile primite de la controllerul gazda, de aceea in continuare ne vom uita la forma pe care o au aceste comenzi SPI.

TABLE 4-1: SPI INSTRUCTION SET FOR THE ENC28J60

Instruction Name and Mnemonic	Byte 0								Byte 1 and Following							
	Opcode			Argument					Data							
Read Control Register (RCR)	0	0	0	a	a	a	a	a	N/A							
Read Buffer Memory (RBM)	0	0	1	1	1	0	1	0	N/A							
Write Control Register (WCR)	0	1	0	a	a	a	a	a	d	d	d	d	d	d	d	d
Write Buffer Memory (WBM)	0	1	1	1	1	0	1	0	d	d	d	d	d	d	d	d
Bit Field Set (BFS)	1	0	0	a	a	a	a	a	d	d	d	d	d	d	d	d
Bit Field Clear (BFC)	1	0	1	a	a	a	a	a	d	d	d	d	d	d	d	d
System Reset Command (Soft Reset) (SRC)	1	1	1	1	1	1	1	1	N/A							

Legend: a = control register address, d = data payload.

Generalitati legate de comenzile SPI:

comenzile sunt luate in considerare de ENC28J60 doar in momentul in care acesta este selectat
-> controllerul gazda pune linia CS (SS) pe 0
o comanda se termina, punandu-se linia CS pe 1
primul byte trimis de controllerul gazda spre cip este format din:
3 biti reprezentand codul operatiei
5 biti reprezentand:
o constanta in cazul scrierii / citirii din bufferul Ethernet
adresa unuia dintre cei 32 registri de control ai bank-ului curent in cazul accesului la un registru de control

Sa analizam urmatoarele comenzi:

Read Control Register (RCR):

controllerul trimite doar un byte
daca se doreste citirea dintr-un registru ETH, imediat dupa primirea comezii, ENC28J60 va incepe transmisia continutului registrului respectiv pe SO.
deci primul byte primit de controller pe linia de MISO va fi chiar informatia dorita (continutul registrului)
daca se doreste citirea dintr-un registru MAC sau MII, prima data ENC28J60 va trimite un

byte dummy, si abia apoi continutul registrului respectiv.

Write Control Register (WCR):

controllerul va trimite 2 octeti, despre primul am vorbit, iar al doilea este byte-ul care va fi scris la adresa registrului specificat.

6. Tema laboratorului

In acest laborator vom folosi ATMega16 pentru a implementa o interfata web. Pagina web afisata contine informatii despre ip-ul microcontroller-ului si al clientului, ora curenta, up-time si se pot controla 8 led-uri. Pentru a ne conecta la reteaua laboratorului vom folosi chip-ul ENC28J60 pe post de slave, in timp ce ATMega16 va fi master iar conexiunea dintre ei va fi de tip SPI. Pentru a afisa pagina web avem nevoie de o implementare a protocolului TCP. Pentru a minimiza dimensiunea codului implementarea va fi simplista (stateless TCP) si nu va permite fragmentarea pachetelor.
Chiar si asa codul devine relativ complex. Cele mai importante parti sunt:

main.c: implementeaza loop-ul principal in care sunt primite cereri.

enc28j60.c: implementeaza functiile de comunicatie intre ATMega16 si ENC28J60.

ip_arp_udp_tcp.c: implementeaza protocolul simplificat TCP

TODO-uri:

Creati un proiect cu sursele de pe site. Dupa ce ati creat proiectul setati nivelul de optimizare la "-Os" din menu-ul Project->Configuration Options->General.
enc28j60.c: implementara functiilor de citire si scriere in/din ENC28J60.
main.c: configurarea ceasului (vezi laboratorul 2)