Zigbee för längre avstånd

Zigbee for longer distanses

Jerker Arnberg

EXAMENSARBETE
Elektroteknik
Nr: E3339E

2006

HÖGSKOLAN
Dalarna

EXAMENSARBETE, C-nivå elektroteknik

Program	Reg nr	Omfattning
Elektroteknik, 120 p	E3339E	10 p
Namn Jerker Arnberg	Datum 2006-01-09	
Handledare Mats Isberg	Examinator Karl-Erik Noréll	
Företag/Institution Dalatron	Kontaktperson vid företaget/institutionen Roland Andersson	
Titel Zigbee för länge avstånd		
Nyckelord Zigbee, nätverk		

Sammanfattning
I dag går utvecklingen av trådlösa nätverk snabbt framåt. Zigbee är en helt ny teknik som bygger på IEEE 802-15-4 standarden. Zigbee utvecklades av the Zigbee Alliance som består av en rad stora elektronikföretag. Zigbee är en teknik som inriktar sig på låg energiförbrukning och låg kostnad. Tekniken är tänkt att användas för att ställa in och läsa av sensorer av olika slag.
Denna rapport är ett resultat av ett examensarbete som går ut på att utreda om Zigbee tekniken kan användas för lite längre avstånd.
Arbetet resulterade i två demoapplikationer för ett enkel zigbee system, och färdigskriven kod för en möjlighet att använda Zigbee för länge avstånd.

Högskolan Dalarna
Röda vägen 3
781 88 BORLÄNGE

Telefon: 023-77 80 00
Telefax: 023-77 80 50
URL: http://www.du.se/

DALARNA
University College

DEGREE PROJECT
Electrical Engineering

Programme	Reg number	Extent
Electrical Engineering	E3339E	15 ECTS
Name of student	Year-Month-Day	
Jerker Arnberg	2006-01-09	
Supervisor	Examiner	
Mats Isberg	Karl-Erik Noréll	
Company/Department	Supervisor at the Company/Department	
Dalatron	Roland Andersson	
Title		
Zigbee for longer distanses		
Keywords		
Zigbee, network		

Summary
To day you can witness a rapid development of wireless network technology. Zigbee is the latest technology that is based on the IEEE 802.15.4 standard. Zigbee was developed by the Zigbee Alliance is a group of some big electronics companies. The Zigbee technology focus on low energy consumption and cost. The technology's main use is to control and read different kinds of sensors.
This report is the result of a thesis work, which purpose was to investigate the possibility for usage of Zigbee for longer distances.
This thesis work resulted in two simple Zigbee demo applications and code written for a possible Zigbee system for longer distances.

Förord

Till dig som läser denna rapport vill jag säga att jag försökt så långt det går, att skriva rapporten på så sätt att vem som helst kan förstå den. Det kan vara en fördel, om man ska förstå vissa mer tekniska kapitel att man behärskar något programmeringsspråk och har läst grundläggande analog och digital ellära.
På grund utav sekretess finns ingen bilaga med källkod, och avsnitten om framtagandet av applikationen går inte in på detaljer.
Jag hoppas hur som helst att rapporten kan gynna dig på något sätt och att du finner den intressant och lättläst.
Jag vill passa på att tacka Dalatron som gav mig möjligheten att utforska detta spännande ämne. Tack även till Dr Robert Reese (reese@ece.msstate.edu) vid Mississippi State University, för exempelkod som varit till stor hjälp för mig under detta arbete.

Innehåll

1 Inledning

1.1 Bakgrund

Dalatron i Orsa utvecklar idag olika system för att läsa av sensorer och skicka dess data till en användare. Under ett projekt fick Roland Andersson på Dalatron idén om att man kanske skulle kunna använda en ny teknik Zigbee för deras ändamål. Då ett krav från deras sida var energisnålhet och låg kostnad så verkade tekniken passa perfekt. Det enda som inte passade var att de applikationer som finns på marknaden idag bara lämpar sig för relativt korta avstånd ca 10 – 50 m.

1.2 Syfte

Syftet med detta examensarbete är att undersöka vilka möjligheter Zigbee har för långa avstånd, och försöka ta fram en labbprototyp för en zigbee applikation som passar Dalatron.

1.3 Mål

Målet är att ta fram ett enkelt demosystem för Zigbee och undersöka möjligheterna för Zigbee för längre avstånd ca 100 – 1000 m.

1.4 Begränsningar

Den största begränsningen för detta projekt är tiden. Den slutliga produkten får heller inte vara allt för dyr eller använda för mycket energi, ett batteri skall räcka månader och kanske till och med år.

1.5 Rapporten

Rapporten är utformad så att först tas lite teori upp för att läsaren enkelt kan följa med i texten om arbetet och vilka resultat som uppnåddes. De olika sakerna som tas upp under teoridelen är de saker som slutligen användes till produkten.

2 Teori

2.1 Zigbee

2.1.1 Vad är Zigbee?

Zigbee är en teknik för trådlösa nätverk inriktat på låg kostnad och energisnålhet. Det lämpar sig bäst för styrning och kontroll av olika sensorer i hemmet eller industrin. Zigbee utvecklades för att vara ett alternativ till Bluetooth. Bluetooth lämpar sig inte för några längre avstånd än 10m, dessutom är bluetooth inte särskilt energisnålt. Zigbee bygger på en IEEE standard för trådlöskommunikation IEEE 802.15.4. Ett nätverk består av ett antal noder. Noderna är uppbyggda som en så kallad mesh eller star.

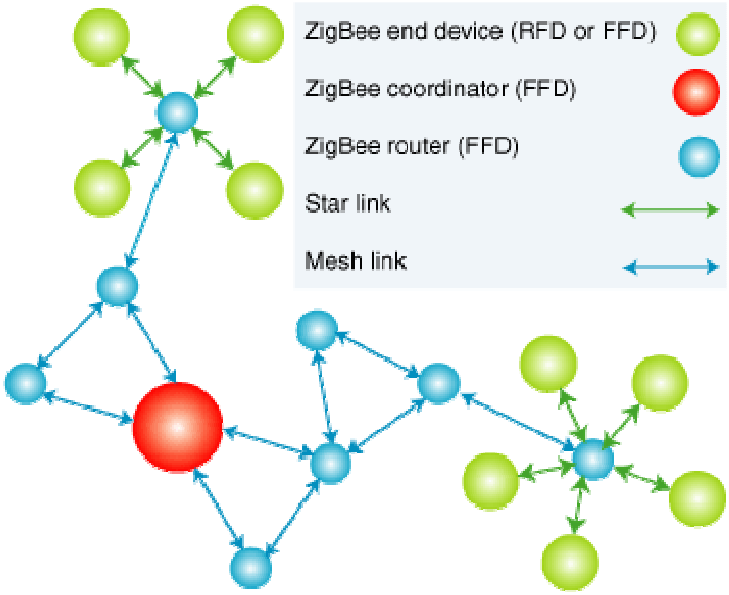

Bild 1 Olika noder i nätverkstoppologier

En nod kan vara en FFD (full-function device). Denna nod kan då antingen vara nätverkets coordinator d.v.s. den nod som startar och håller koll på nätverket, eller en router som vidarebefordrar data.
En FFD använder ganska mycket processorkraft och behöver mer minne för att hålla ordning på nätverket. En nod kan även vara en så kallad RFD (reduced-function device).
En RFD kan inte routa data, och är därför inte ett alternativ i en mesh-topologi. Den passar i en star-topologi och kan enbart kommunicera med en FFD. En RFD är mindre komplex än en FFD och kräver därför mycket mindre minne och processorkraft.

2.1.2 Zigbee Alliance

Zigbee Alliance startades av ett par företag som ville skapa en enkel, billig och energisnål standard för trådlöskommunikation. Idag är många olika företag medlemmar i Zigbee Alliance, allt från chiptillverkare till mjukvaruföretag. På listan över medlemmar finns namn som Philips, Samsung och ABB.

2.1.3 Zigbee översikt

Zigbee standarden bygger alltså på standarden IEEE 802.15.4 som togs fram av IEEE för att vara en standard som byggde på låg komplexitet, låg energiförbrukning och låg kostnad. Den specificerar de olika lagren och hur ett nätverk ska byggas med FFD:s och RFD:s. Zigbee bygger på den kända OSI-modellen och dess lager där de lägsta lagren är specificerade av IEEE 802.15.4 och de övre av Zigbee Alliance.

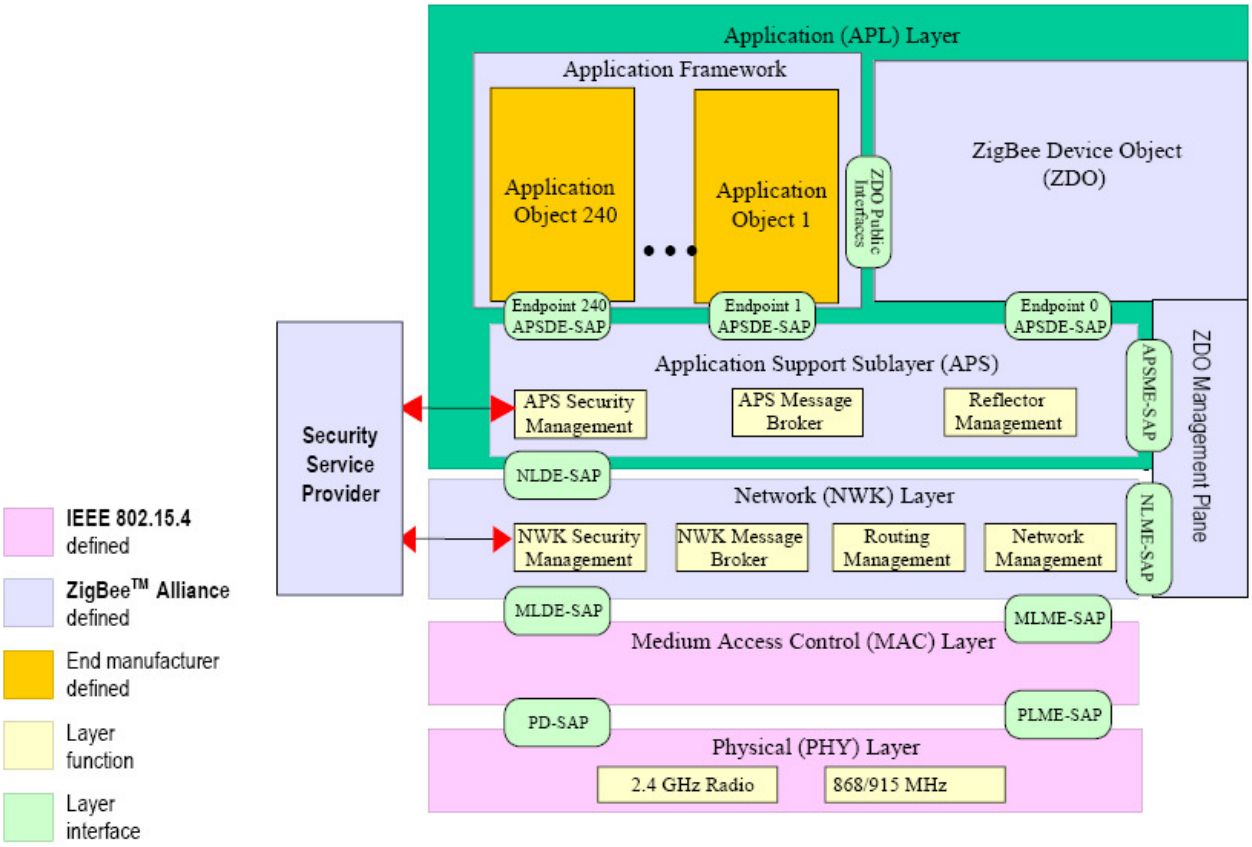

Bild 2 Zigbee lagrer översikt

2.1.4 Fysiska lagret

Det fysiska lagret är definierat av IEEE 802.15.4 och beskriver vilka olika frekvenser som kan användas med olika överförings hastighet. För 2,4GHz banden är överförningshastigheten 250Kb/s. För 915MHz banden 40Kb/s och för 868MHz 20Kb/s. Här specificeras även hur datan skall överföras.

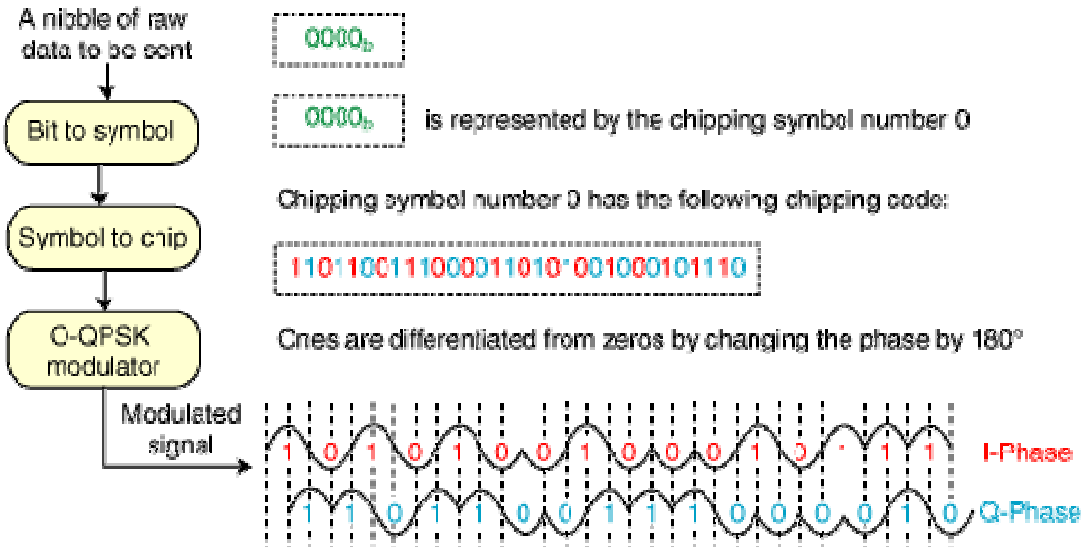

Bild 3 Signal modulering

Det första som händer när man ska skicka data är att datan översätts till en symbol 0 – 15, detta eftersom det finns 16 olika kombinations möjligheter med 4 bitar. Symbolen översätts sedan till en så kallad chipping code där varje symbol har en unik kod. En modulator skickar därefter ut en modulerad signal.

2.1.5 MAC lagret

MAC lagret är definierat av IEEE 802.15.4 och har hand om att lyssna på nätverket, söka efter noder som finns i nätverket och har hand om timingen i nätverket. Det översta av detta lager är implementerat i mjukvaran. Detta lager schemalägger och routar dataframes.

2.1.6 Network lagret

Network lagret är definierat av Zigbee Alliance och har hand om funktionen för att nätverket ska vara självbyggande. Detta lager har hand om att spara och ha koll på vilka noder som lämnar och kommer in i nätverket.

2.1.7 Application lagret

Application lagret består av ett underlager som är definierat av Zigbee Alliance och har hand om bindningen med andra noder. Den övre delen av Applications lagret är användarens applikation, här ska definieras om det ska vara en RFD eller en FFD, en coordinator eller router och vad just applikationen skall göra.

2.2 Hårdvara

2.2.1 Picdem Z

Picdem Z är en produkt från Microchip för att demonstrera en lättanvänd Zigbeeapplikation. Den innehåller två moderkort med en PIC18F4620 PIC-processor och två dotterkort med en CC2420 RF-tranciver.

Med demonstrationskorten följer en färdigskriven stack från microchip och en exempelapplikation för att tända och släcka en diod på respektive kort.

Morderkortet innehåller en 40-pinnars DIP-socket för en microprossesor, med följer PIC18F4620 processorn förprogrammerad med microchips stack och demoapplikationen. Moderkortet har två stycken lysdioder, tre tryckknappar, en för reset och två som kan användas till en applikation. Ett RJ-11 interface, som kan användas av ett utvecklingsprogram för programmering och debugging. Ett RS232 interface för att kommunicera med en PC eller ett annat kort. En kontakt för ett RF-kort med seriell överföring av data (SPI) och några I/O pinnar. Ett tomt utrymme dit alla oanvända pinnar är utdragna. Samt energiförsörjning och en kontakt för ett 9V batteri.

Demokorten kommer förprogrammerade som en coordinator (FFD) och en enddevice (RFD).

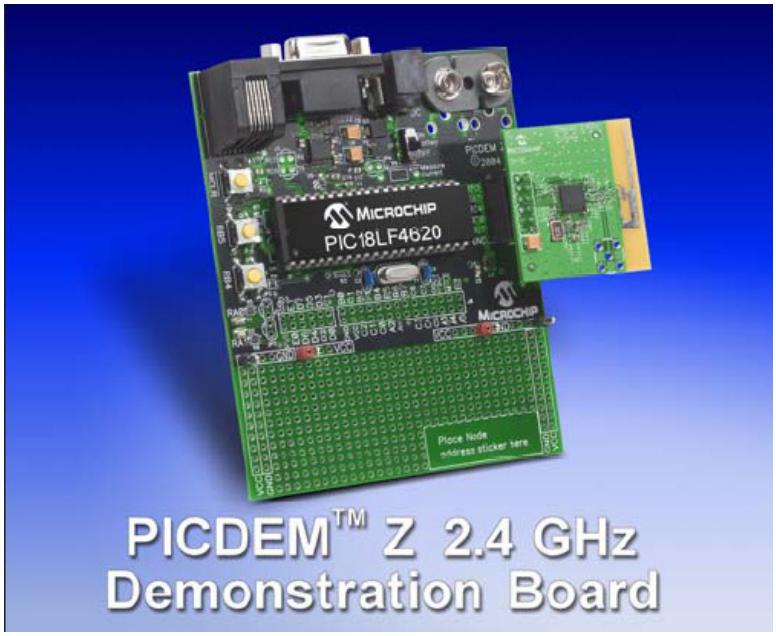

Bild 4 Picdem Z 2.4 GHz Demonstration Board

2.2.2 ZMD44101

ZMD44101 är en fullt IEEE 802.11.4 kompatibel RF-tranciver, som använder DSSS (Direct Sequence Spread Spectrum). Överförnings hastigheterna är 40kbit/s för 902 till 928 MHz eller 20kbit/s för 868.3MHz. ZMD 44101 är en Zigbee ready tranciver d.v.s. den har fysiska lagret och mac lagret implementerat. ZMD 44101 kan sända över ett avstånd på ca 100m.

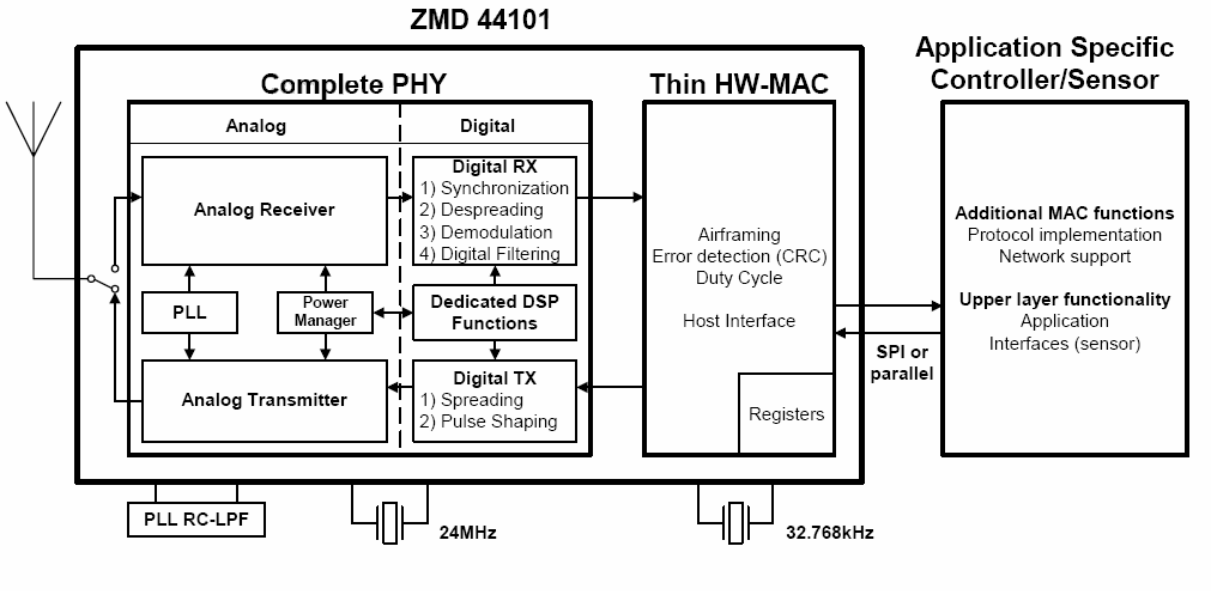

Bild 5 ZMD44101 överblick

ZMD44101 erbjuder två olika kommunikationsmöjligheter, antingen via ett SPI-interface (seriellt) eller ett parallellt interface.
Det seriella interfacet konfigureras via ett register. Standard är att SPI är satt som slave och behöver då en processor som agerar som master. Det av interfacen som inte används ska stängas av, detta görs genom att man sätter ett antal ingångar till 0.

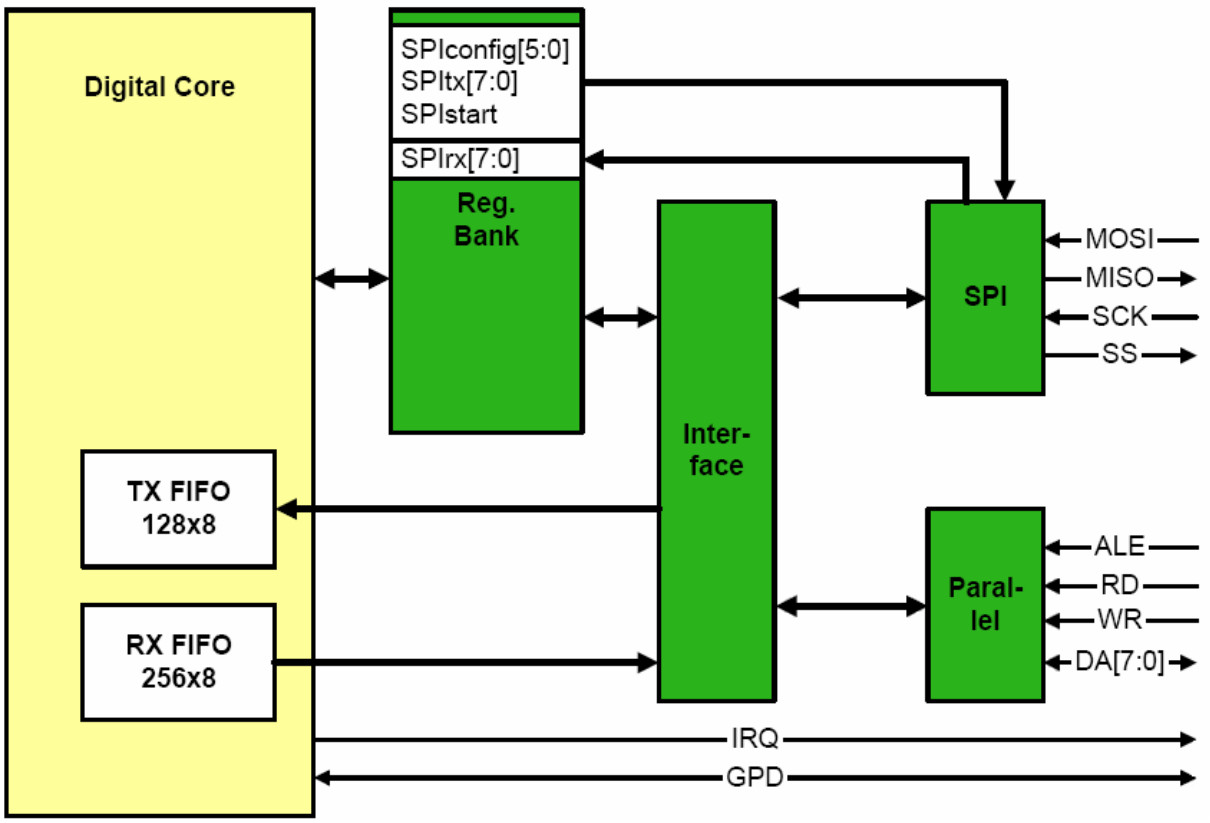

Bild 6 ZMD44101 kommunikatins möjligheter

I detta projekt används det seriella interfacet, och det beskrivs därför mer ingående..

Det seriella interfacet erbjuder en seriell länk direkt till ZMD 44101:s TX FIFO (register för data att skicka, first in first out) och RX FIFO (register för mottagen data, first in first out). SPI interfacet erbjuder de vanliga MOSI, MISO, SCK och SS signalerna som är standard för ett seriellt interface. Se bild 6 för en överblick över systemet.

För att skriva data till ZMD 44101 ska den första biten i den första byten skriven till MOSI vara satt till ’0’ och satt till ’1’ för en skrivoperation. All data sänds som MSB (mest signifikanta biten) först och LSB (minst signifikanta biten) sist. Protokollet för att överföra data börjar med en byte där första biten är en Read/Write indikator. Därefter en längd indikator som anger hur mycket data som ska skickas. Den andra byten innehåller adressen till vilket register som man vill skriva eller läsa ifrån. De följande byten är data som ska skickas.

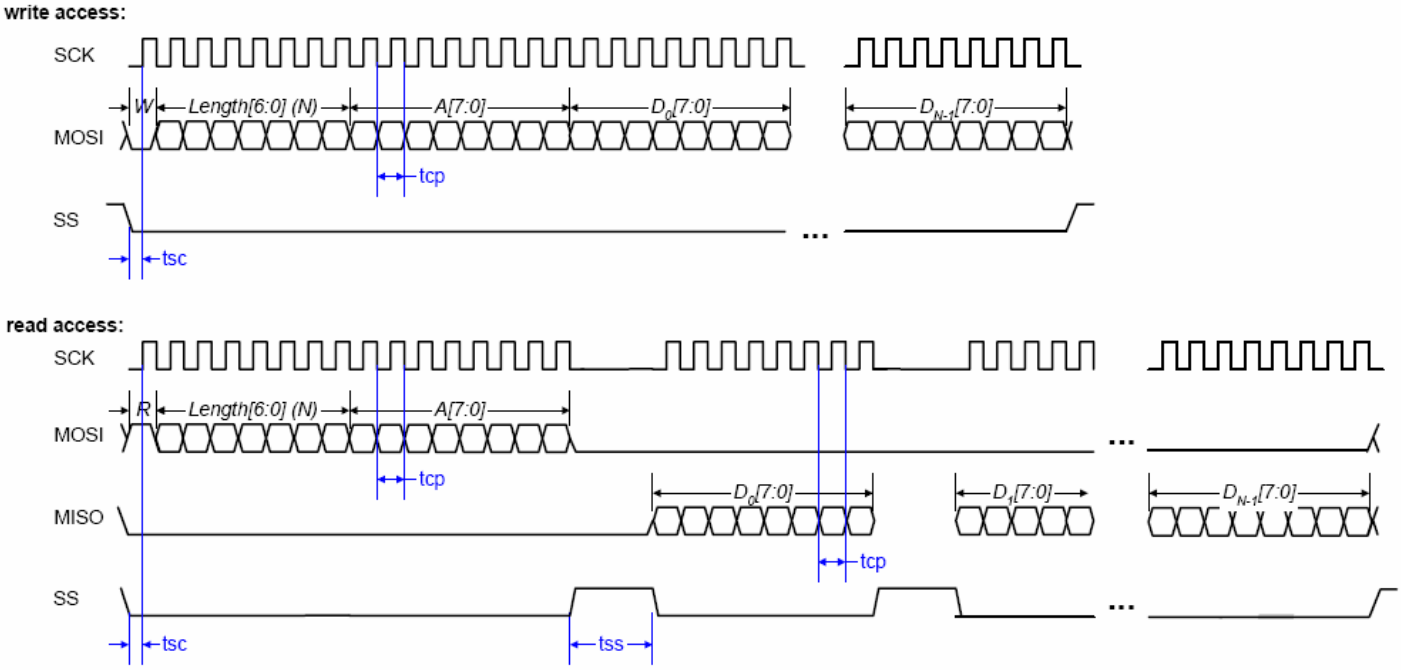

Bild 7 Exempel på en läsning och skrivning med SPI

Bild 7 visar hur en write resp. read access kan se ut.

ZMD 44101 har många register för att konfigurera MAC lagret resp. det fysiska lagret. Genom att skriva till registren så kan man konfigurera trancivern för olika ändamål t.ex. kan frekvensen ställas in och olika statusar läsas.

De olika registren är:

- **MAC control + status register**
  Innehåller register för att konfigurera MAC lagret och olika status register.

- **MAC timing registers**
  Innehåller register för att konfigurera timingparametrar.

- **Other MAC registers**
  Innehåller avancerade nätverks inställningar.

- **MAC header registers**
  I dessa register konfigureras t.ex. den egna och mottagarens MAC-adresser.

- **PHY registers**
  Innehåller register för den fysiska delen t.ex. vilken kanal som man sänder på.

ZMD 44101 behöver väldigt få externa komponenter. På schemat nedan visas ett exempel på hur en typisk applikation kan byggas. Utöver det som visas behövs en microprocessor som har mjukvaran för applikationen installerad.

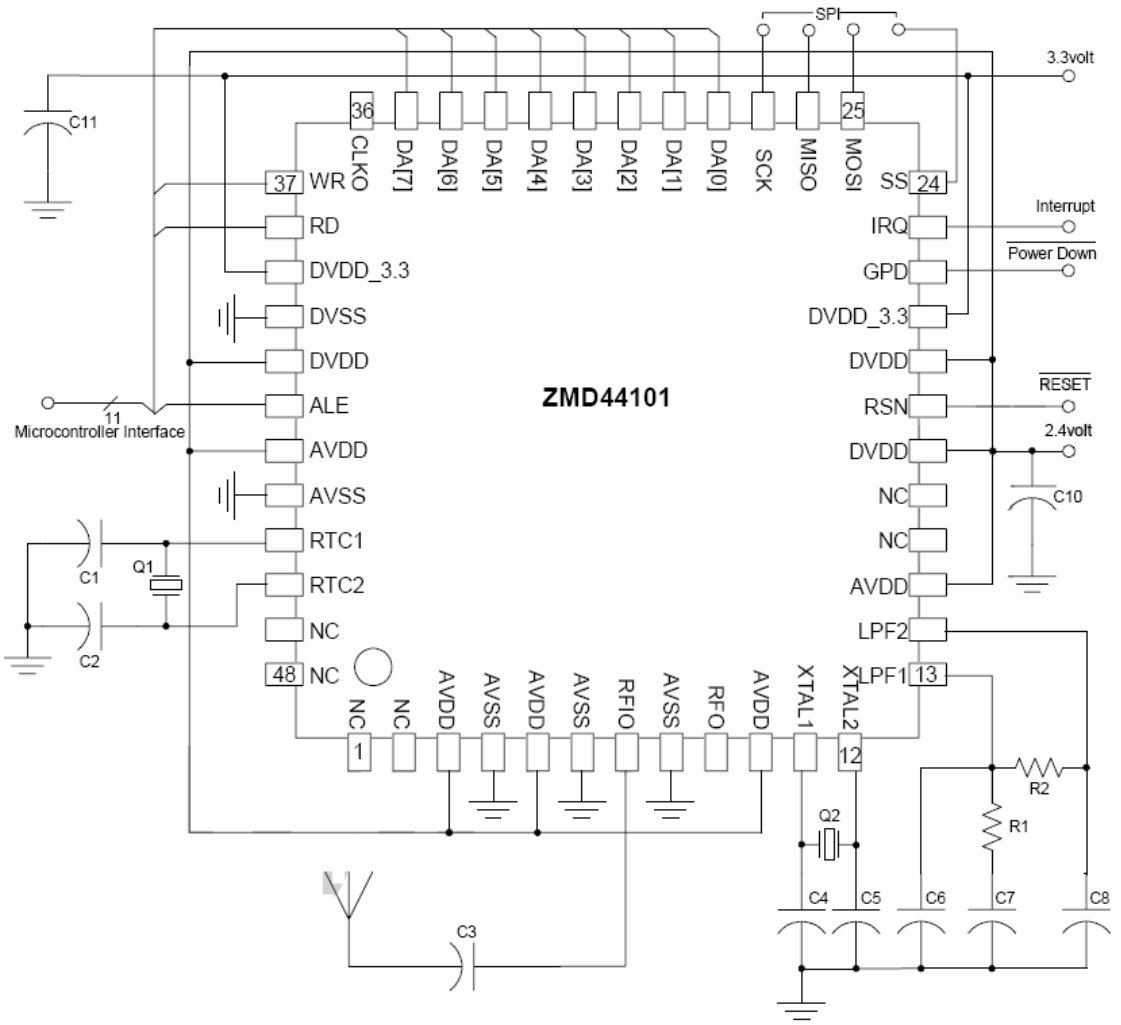

Bild 8 ZMD44101 Pinlayout

2.2.3 PIC18F4620

PIC18F4620 är en microcontroller (uC) och skiljer sig lite från en microprocessor (uP). En microcontroller behöver inget externt minne och något interface, då detta är inkluderat i själva kretsen. En uC är tänkt som en helhetslösning i ett enda chip. Den har oftast fler funktioner vad gäller olika seriella kommunikationsmöjligheter och fler A/D, D/A interface. En uC har en intern data och adressbus. Man kan säga att en uC är en liten komplett dator.

PIC18F4620 är framtagen för att vara en energisnål krets. Den kan antingen klockas från den interna oscillatorn eller med en timer vilket medför att under exekvering kan man reducera energiförbrukningen med upptill 90%. Man kan även försätta PIC18F4620 i ett vänteläge, och där igenom spara ytterligare energi, ned till 4% av den normala energiförbrukningen. Dessa olika lägen kan användaren själv styra, och de skrivs in i koden där det passar för den specifika applikationen.
Med hjälp av en bootloaderfunktion kan mjukvaran skriva till programminnet själv. Det är därför möjligt att bygga en applikation som kan uppdatera sig själv.

PIC18F4620 har en adresserbar USART-port som stödjer LIN-bus protokollet. Denna port kan enkelt användas som ett vanligt RS-232 interface för kommunikation med andra uC eller datorer.

PIC18F4620 styrs via ett antal register, genom att skriva till de olika registren kan man ändra processorns egenskaper. Ett av registren är t.ex. oscillator registret, där kan man ställa in olika frekvenser så det passar just den aktuella applikationen.

PIC18F4620 har totalt 7 olika operativa körmöjligheter. Dessa kategoriseras enligt följande:

- Run mode
- Idle mode
- Sleep mode

Run mode är default efter en reset, i detta mode drar kretsen maximalt med ström. Både CPU:n och kringliggande enheter klockas.
I Idle mode klockas bara kringliggande enheter och inte processorn. Detta bidrar till minskad strömförbrukning.
I Sleep mode klockas varken CPU:n eller kringliggande enheter. För att vakna upp ur denna mode krävs att ett avbrott sker.
För att en reset ska inträffa på PIC18F4620 kan följande inträffa:

- (POR) Power On Reset
- MCLR (Master CLeaR) aktiv låg
- WDT (WatchDog Timer)
- (BOR) programmable Brown-Out Reset
- RESET (Instruktion)
- Stack Full Reset
- Stack Underflow Reset

En POR inträffar när man startar kretsen genom att slå på ström. MCLR är en extern reset d.v.s. genom att fysiskt lägga en nolla på MCLR pinnen en stund så resetas kretsen och inget internt kan påverka MCLR. WDT inträffar när watchdog timern har räknat över, detta är ett mycket bra sätt att se till att en applikation aldrig låser sig någonstans i koden. Genom att mjukvaran alltid före den gör något startar om WDT:n så håller den koll på om det som man tänkt utföra utförs, eller om man fastnar i ett tillstånd. Om Vdd (matningsspänningen) sjunker under ett visst inställt värde inträffar en BOR. PIC18F4620 har även en instruktion för att utföra en reset när som helst i mjukvaran. Om man har konfigurerat ett register som har hand om stacken på ett givet sätt kan en Stack full eller Stack underflow reset inträffa.

Efter att någon av de olika typer av reset har inträffat går det att läsa av vilken som inträffade i ett flaggregister och vidta åtgärd efter vad man avläst.
PIC18F4620 har tre stycken olika minnen:

- Program minne
- Data RAM
- Data EEPROM

Programminnet är 64Kb stort, men arkitekturen tillåter upp till 2Mb programminne. I programminnet kan man lagra upp till 32768st instruktioner.

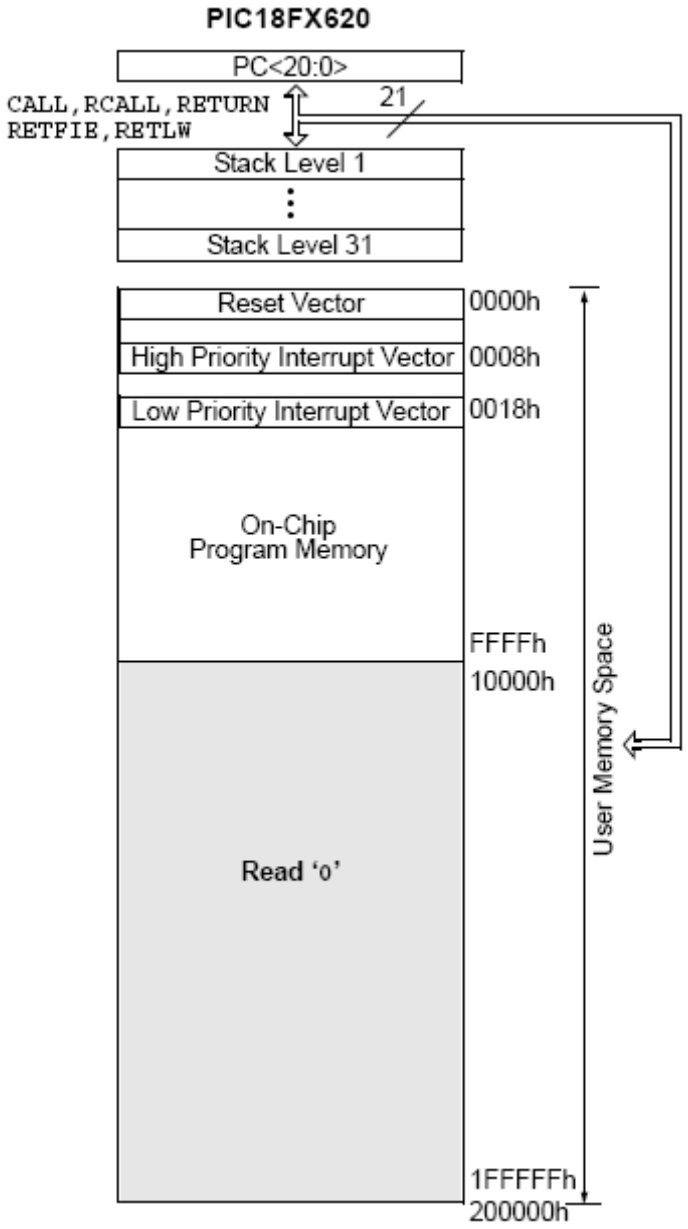

Bild 9 Program memory map and stack for PIC18F4620

Dataminnet i en PIC18F4620 är implementerat som statiskt RAM. Data-minnet innehåller alla SRF:s (Special Funktion Registers) och GPS:s (General Purpose Registers). SFR registren används för kontroll och status för PIC-kretsen, och GPR registren för data lagring. SFR:s används av CPU:n och kringliggande enheter för att kontrollera den angivna inställningen av kretsen.

Address	Name	Address	Name	Address	Name	Address	Name
FFFh	TOSU	FDFh	INDF2(1)	FBFh	CCPR1H	F9Fh	IPR1
FFEh	TOSH	FDEh	POSTINC2(1)	FBEh	CCPR1L	F9Eh	PIR1
FFDh	TOSL	FDDh	POSTDEC2(1)	FBDh	CCP1CON	F9Dh	PIE1
FFCh	STKPTR	FDCh	PREINC2(1)	FBCh	CCPR2H	F9Ch	—(2)
FFBh	PCLATU	FDBh	PLUSW2(1)	FBBh	CCPR2L	F9Bh	OSCTUNE
FFAh	PCLATH	FDAh	FSR2H	FBAh	CCP2CON	F9Ah	—(2)
FF9h	PCL	FD9h	FSR2L	FB9h	—(2)	F99h	—(2)
FF8h	TBLPTRU	FD8h	STATUS	FB8h	BAUDCON	F98h	—(2)
FF7h	TBLPTRH	FD7h	TMR0H	FB7h	PWM1CON(3)	F97h	—(2)
FF6h	TBLPTRL	FD6h	TMR0L	FB6h	ECCP1AS(3)	F96h	TRISE(3)
FF5h	TABLAT	FD5h	T0CON	FB5h	CVRCON	F95h	TRISD(3)
FF4h	PRODH	FD4h	—(2)	FB4h	CMCON	F94h	TRISC
FF3h	PRODL	FD3h	OSCCON	FB3h	TMR3H	F93h	TRISB
FF2h	INTCON	FD2h	HLVDCON	FB2h	TMR3L	F92h	TRISA
FF1h	INTCON2	FD1h	WDTCON	FB1h	T3CON	F91h	—(2)
FF0h	INTCON3	FD0h	RCON	FB0h	SPBRGH	F90h	—(2)
FEFh	INDF0(1)	FCFh	TMR1H	FAFh	SPBRG	F8Fh	—(2)
FEEh	POSTINC0(1)	FCEh	TMR1L	FAEh	RCREG	F8Eh	—(2)
FEDh	POSTDEC0(1)	FCDh	T1CON	FADh	TXREG	F8Dh	LATE(3)
FECh	PREINC0(1)	FCCh	TMR2	FACh	TXSTA	F8Ch	LATD(3)
FEBh	PLUSW0(1)	FCBh	PR2	FABh	RCSTA	F8Bh	LATC
FEAh	FSR0H	FCAh	T2CON	FAAh	EEADRH	F8Ah	LATB
FE9h	FSR0L	FC9h	SSPBUF	FA9h	EEADR	F89h	LATA
FE8h	WREG	FC8h	SSPADD	FA8h	EEDATA	F88h	—(2)
FE7h	INDF1(1)	FC7h	SSPSTAT	FA7h	EECON2(1)	F87h	—(2)
FE6h	POSTINC1(1)	FC6h	SSPCON1	FA6h	EECON1	F86h	—(2)
FE5h	POSTDEC1(1)	FC5h	SSPCON2	FA5h	—(2)	F85h	—(2)
FE4h	PREINC1(1)	FC4h	ADRESH	FA4h	—(2)	F84h	PORTE(3)
FE3h	PLUSW1(1)	FC3h	ADRESL	FA3h	—(2)	F83h	PORTD(3)
FE2h	FSR1H	FC2h	ADCON0	FA2h	IPR2	F82h	PORTC
FE1h	FSR1L	FC1h	ADCON1	FA1h	PIR2	F81h	PORTB
FE0h	BSR	FC0h	ADCON2	FA0h	PIE2	F80h	PORTA

Bild 10 Special Function Register map for PIC18F4620

Program flashminnet är läs- skriv- och tömbart under normal operation. När man läser från minnet läser man 1 byte i taget, och när man skriver så skriver man block om 64 byte styck.

DataEEPROM minnet är läs och skrivbart under normal operation. Man kommer åt minnet genom att använda SFR:s. För att skriva eller läsa EEPROM använder man SFR EEDATA för datan och SFR EEADRH, EEADR för att ange adressen.
PIC18F4620 har många avbrottskällor, för att konfigurera dessa används registren:

- RCON
- INTCON
- INTCON2
- INTCON3
- PIR1, PIR2
- PIE1, PIE2
- IPR1, IPR2

Dessa register används för att enabla ett avbrott och sätta prioriterings ordning. Den högsta prioriterade avbrottsvektorn ligger på 0008h och den lägsta ligger på 0018h.

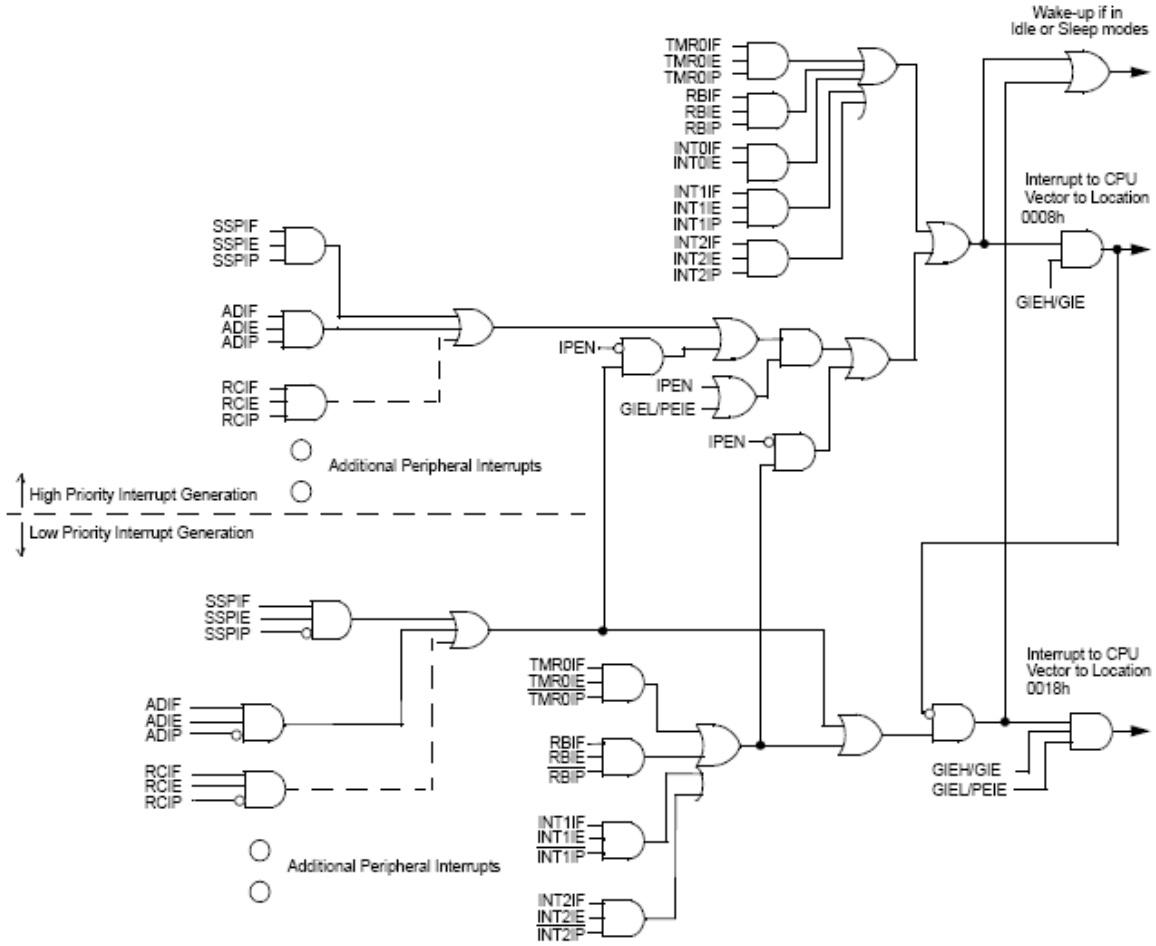

Bild 11 PIC18F4620 INTERRUPT LOGIC

PIC18F4620 har upp till 5st I/O portar. Man konfigurerar en port genom följande tre register:

- TRIS (data direction register)
- PORT (värdet på den aktuella pinnen)
- LAT (output latch)

För att t.ex. använda porten A som input skriver man en ”etta” till register TRISA och sedan kan man läsa av register PORTA.

PIC18F4620 har flera olika moduler inbyggt, en utav dessa är den som används för att kommunicera med dotterkortet på PICDEM Z. Denna modul kallas MSSP (Master Synchronous Serial Port). MSSP är ett seriellt interface som är utmärkt för kommunikation med andra enheter. MSSP modulen kan konfigureras antingen eller som:

- SPI™ (Seriall Peripheral Interface)
- I2C™ (Inter-Integrated Circuit)

Båda varianterna kan vara antingen master eller slave. Det som används till detta projekt är SPI. SPI sänder 1byte data synkront seriellt till ett annat SPI

interface. Man kan sända och ta emot data samtidigt. SPI använder fyra stycken pinnar för att kommunicera:

- SDO (Serial Data Out)
- SDI (Serial Data In)
- SCK (Serial Clock)
- SS (Slave Select aktiv låg)

I master mode så är SCK och SS utgångar och i slave mode är dessa ingångar.

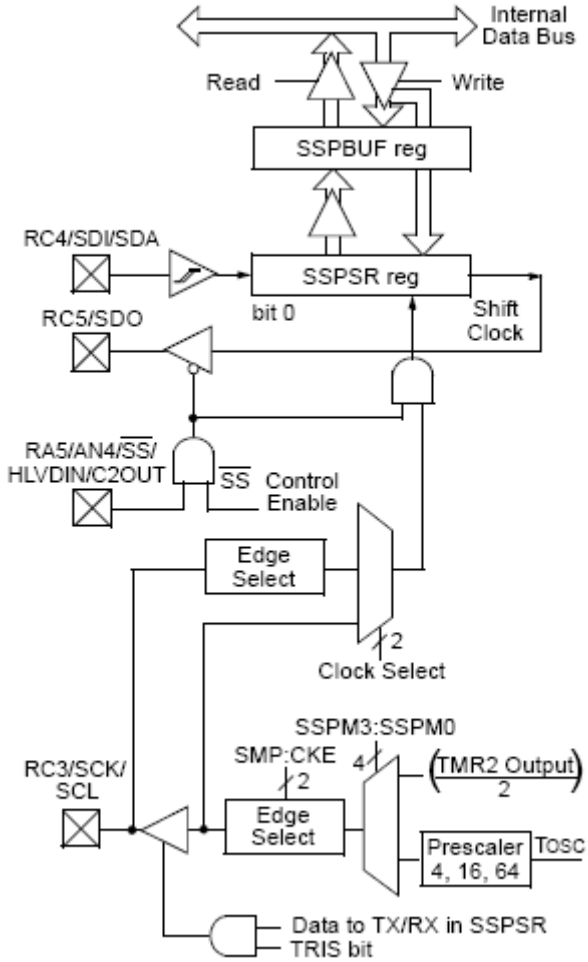

Bild 12 MSSP Block Diagram (SPI™ MODE)

Det är fyra register som används för att styra SPI:n:

- SSPCON1 (MSSP Control Register 1)
- SSPSTAT (MMSP Status Register)
- SSPBUF (Serial Recive/Transmit Buffer Register)
- SSPSR (MSSP Shift Register)

SSPCON1 och SSPSTAT används för att konfigurera resp. kontrollera SPI:n. SSPSR skiftar data in eller ut ur buffern, och är inte direkt läs- eller skrivbar. SSPBUF används för att läsa den mottagna datan eller för att skicka data.

För att börja kommunikationen måste man bestämma vem som ska vara en master d.v.s. styr SCK. Sedan måste man bestämma om man ska klocka datan på stigande eller fallande tillslag av klockan. All data skickas MSB (mest signifikanta biten) först. En typisk koppling mellan två processorer visas på bild 13.

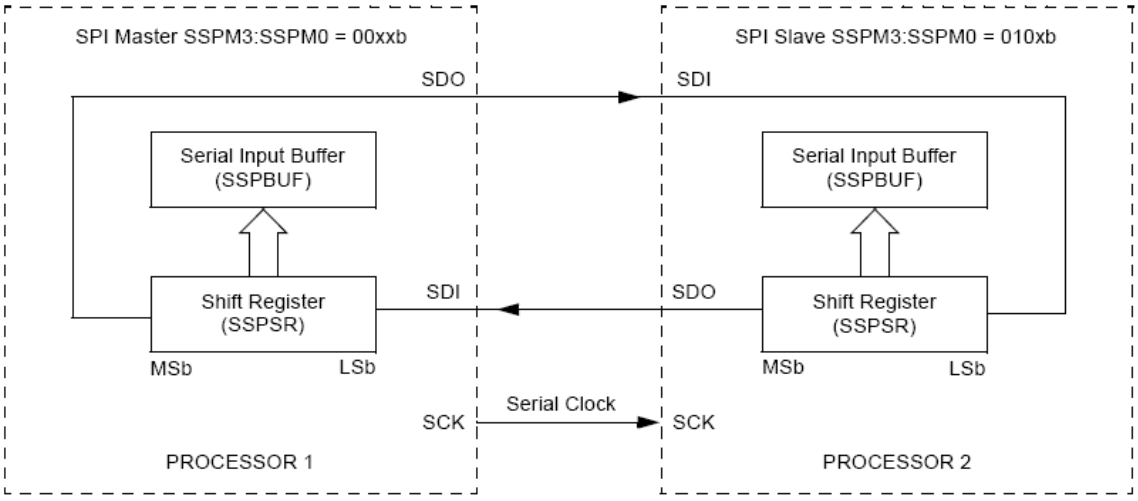

Bild 13 SPI™ Master/Slave Connection

En master kan börja sända data när som helst genom att den kontrollerar SCK. Så fort man skriver till SSPBUF så börjar mastern sända data. Nedan visas hur en kommunikation kan se ut.

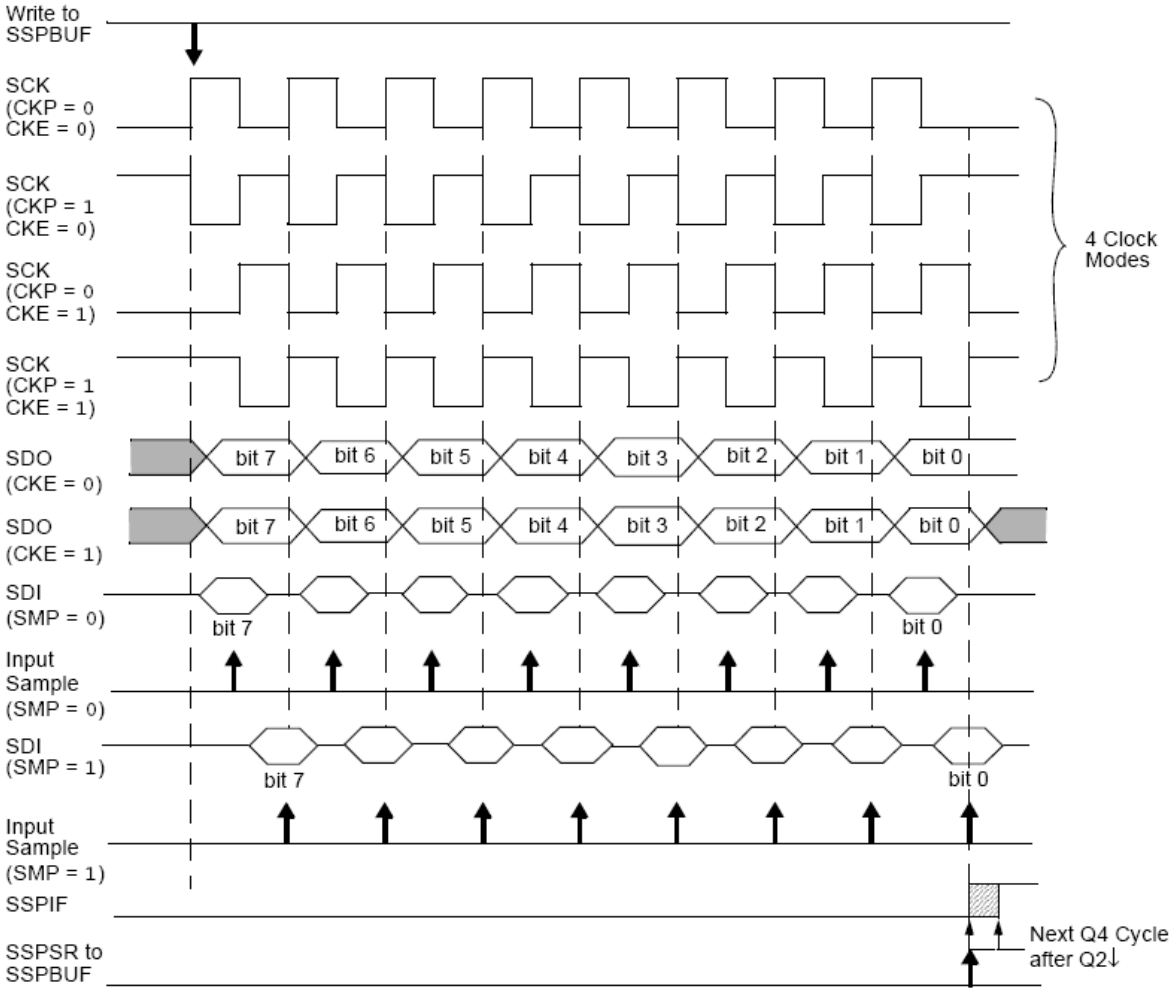

Bild 14 SPI™ Mode Waveform (Master Mode)

För att en slave ska kunna sända data så måste man ge en Slave Select (SS) signal till mastern som kan starta klockan. Detta kallas för en synkroniseringsfas, därefter kan slaven börja sända data. Nedan visas hur en typisk kommunikation kan se ut.

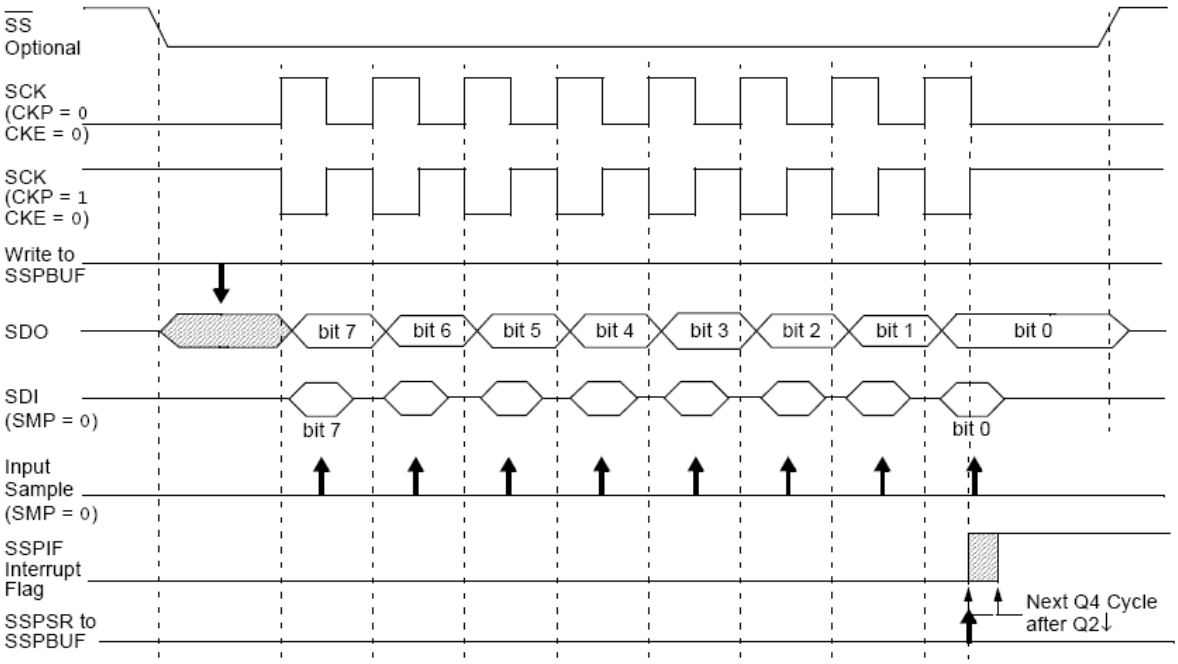

Bild 15 SPI™ Mode Waveform (Slave Mode)

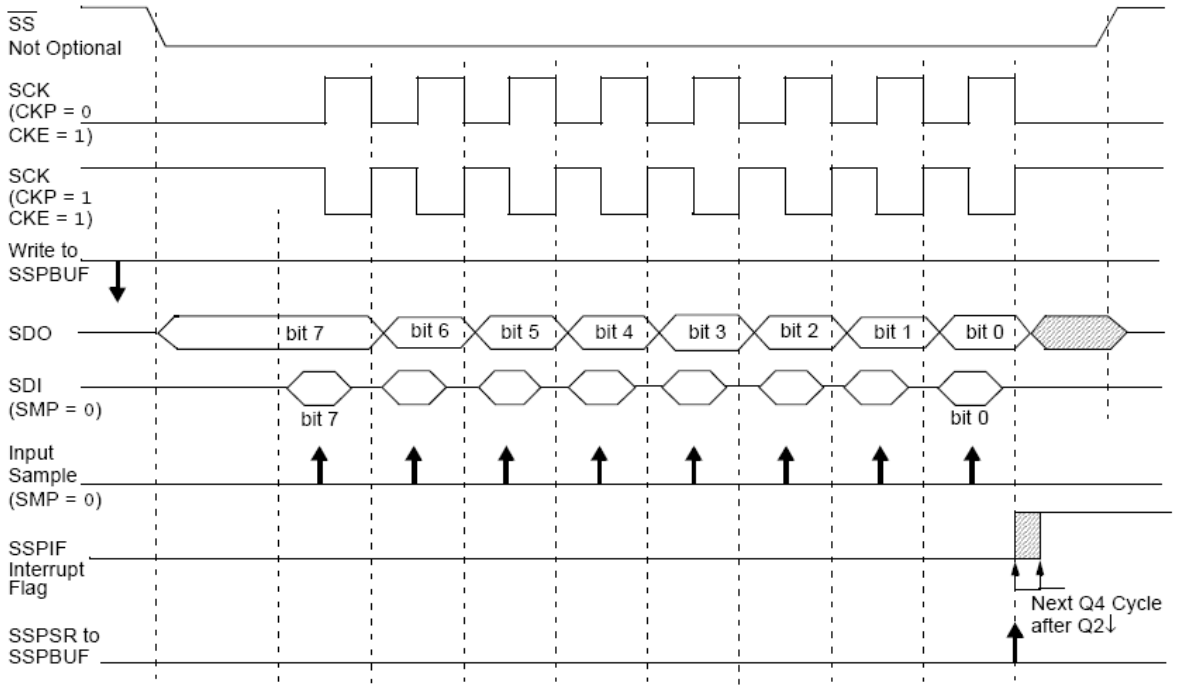

Bild 16 SPI™ Mode Waveform (Slave Mode)

2.3 Mjukvara

2.3.1 Microchip stack for Zigbee

Microchip stack for Zigbee är baserad på version 1.0 av Zigbee specifikationen från Zigbee Alliance. Det är en öppen källkod som finns att ladda ner gratis från internet från microchips hemsida. Stacken är skriven i programmeringsspråket C. Stacken stödjer 2.4GHz bandet med Chipcons CC2420 RF-tranciver eller Ubec 2400 RF-tranciver. För att använda stacken till en applikation behövs dessa tre saker:

- En PIC18F microcontroller med ett SPI™ interface.
- En Zigbee-ready RF-tranciver som stacken stödjer.
- En antenn.

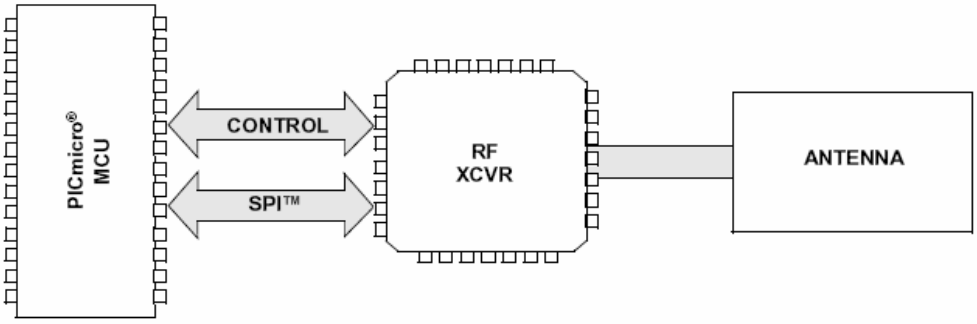

Bild 17 TYPICAL ZigBee™ NODE HARDWARE

Bild 17 visar hur en typisk nod kan vara uppbyggd.

Microchip Stack for Zigbee innehåller en rad olika källkodsfiler som man kan bygga sin applikation med. All kod kompileras ej, utan man bygger sin applikation utgående från en definitionsfil. Hela stacken är skriven med preprocessordirektiv, d.v.s. beroende på vad som är definierat i definitionsfilen kompileras bara den kod som behövs för den specifika applikationen.

Med stacken följer två demonstrations-applikationer passande för microchips Picdem Z. Det första demot är en coordinator som startar upp ett nätverk. Det andra är en RFD. Genom att trycka på olika switchar kan man tända och släcka dioder på motstående nod:s moderkort.
Man behöver en del filer för att bygga en applikation Tabell 1 visar vilka filer man behöver för att bygga en typisk RFD nod, och Tabell 2 visar filerna för en FFD.

Source Files	Purpose
Your App Files	Must include `main()` entry point
`zigbee.def`	Microchip Stack options specific to your application
`Console.c`	RS-232 terminal routines – needed if `ENABLE_DEBUG` is defined or your application uses console routines
`MSPI.c`	Master SPI™ interface routines to access RF transceiver
`Tick.c`	Tick manager, used to keep track of time-out and retry conditions
`zAPL.c`	ZigBee™ application layer
`zAPS.c`	ZigBee application support sublayer
`ZDO.c`	ZigBee device object – required if ZigBee remote management is needed
`zMAC.c`	IEEE 802.15.4 MAC layer
`zNVM.c`	Nonvolatile memory storage routines – may be replaced with your own nonvolatile storage specific file
`zNWK.c`	ZigBee network layer
`ZPHY???.c`	RF transceiver specific routines – `zPHYCC2420.c` for Chipcon CC2420 and `zPHYZMD44101.c` for ZMD 44101 transceiver
`zProfile.c`	ZigBee profile routines – required if standard profile support is needed (not fully implemented in version 1.00.00)
`18f????.lkr`	Linker script file specific your selection of device – required if using C18

Tabell 1 Filer som behövs för en typisk RFD

Source Files	Purpose
Your App Files	Must include `main()` entry point
`zigbee.def`	Microchip Stack options specific to your application
`Console.c`	RS-232 terminal routines – needed if `ENABLE_DEBUG` is defined or your application uses console routines
`NeighborTable.c`	Implements neighbor and binding table
`SRAlloc.c`	Dynamic memory manager to implement indirect transmit buffer
`Tick.c`	Tick manager
`zAPL.c`	ZigBee™ application layer
`zAPS.c`	ZigBee application support sublayer
`ZDO.c`	ZigBee device object – required if ZigBee remote management is needed
`zMAC.c`	IEEE 802.15.4 MAC layer
`zNVM.c`	Nonvolatile memory storage routines – may be replaced with your own nonvolatile storage specific file
`zNWK.c`	ZigBee network layer
`ZPHY???.c`	RF transceiver specific routines – `zPHYCC2420.c` for Chipcon CC2420 and `zPHYZMD44101.c` for ZMD 44101 transceiver
`zProfile.c`	ZigBee profile routines – Required if standard profile support is needed (not fully implemented in version 1.00.00)
`18f????.lkr`	Linker script file specific your selection of device – required if using C18

Tabell 2 Filer som behövs för en typisk FFD

Microchip Stack for Zigbee använder sig av en rad callbackfunktioner, som man som kodare måste implementera. En callbackfunktion anropas av stacken när något inträffar. Det kan t.ex. vara om en nod joinar vårt nätverk, då anropar stacken funktionen:

```
BOOL AppOkayToAcceptThisNode(LONG_ADDR *longAddr)
```

Denna funktion få man alltså implementera själv och avgöra i detta fall om den aktuella noden får komma in i nätverket eller inte.

3 Arbetet

3.1 Studie

Arbetet började med en litteraturstudie i ämnet för att få lite på fötterna innan möjligheterna undersöktes.

Det bestämdes ganska snabbt att ett utvecklingskort från microchip PicdemZ skulle vara utgångspunkten för projektet.

Det första förslaget som undersöktes var möjligheten att använda en tranciver från chipcon CC1020 som sänder på frekvensen 400MHz. Problemet med denna var att den inte var Zigbee-ready d.v.s. inget MAC lager och fysiskt lager var implementerat. Detta förslag hade sina nackdelar då Zigbee standarden inte kunde följas, eftersom IEEE 802.15.4 inte definierar en frekvens på 400MHz.

Idén var då att mellan PIC-processorn på Picdem Z och CC1020 skulle det placeras antingen en FPGA eller en processor där de saknade lagren skulle implementeras. Men efter att ha studerat detta närmre, läst delar av IEEE 802.15.4 standarden och haft kontakt med en utvecklare från Chipcon drogs slutsatsen att ett sådant projekt skulle vara alldeles för omfattande. Då fanns det två alternativ kvar, antingen att gå ifrån Zigbee helt och designa ett enkelt system för att routa data, eller att använda några av de frekvenser som IEEE 802.15.4 definierar.

Det sista förslaget valdes eftersom det skulle bli det bästa för både eamensarbetet och Dalatron.

Den tranciver som skulle användas blev ZMD:s 44101.
Dotterkortet som trancivern skall sitta på och som skall passa till PicdemZ utvecklar Dalatron. Uppgiften blev att implementera stacken och applikationen.

3.2 Implementering

3.2.1 Stack

Stacken från microchip var det första som implementerades. Med hjälp av det medföljande demot programmerades och testkördes de två noderna. ”Demot” går ut på att när man trycker på en knapp på den ena noden, så ska en diod tändas på den andra noden.
Allt fungerande tillfredställande.

3.2.2 Protocoll

Roland Andersson på Dalatron ville ganska tidigt att det skulle specificeras upp ett internt protokoll som skulle användas för kommunikationen mellan noderna.

Utgående från två tidigare protokoll som Dalatron använt, ett lite enklare protokoll och ett betydligt mer omfattande. Bestämdes att ett lite mindre enklare protokoll skulle specificeras, då applikationen inte krävder så många olika kommandon.

För att man lätt skulle kunna kommunicera med ett tänkt användargränssnitt på en dator, så bestämdes att det skulle användas ASCII-tecken för att kommunicera.

Det första som behövdes var en avgränsare (delimiter). Delimitern inleder protokollet, och anger om det är en förfrågan (request) eller ett svar (replay).

ASCII-tecknet # inleder protokollet om det handlar om en request och tecknet ? om det handlar om ett svar.

Det som var nödvändigt i för projektet var att veta vilken av alla noder som har skickat datan och vilken som ska få datan. Här begränsades tekniken lite. Då man i princip skulle kunna använda hela den 64-bitar långa IEEE adressen, som skulle möjliggöra miljontals kombinationer. Användes istället tre ASCII-tecken för att avgöra detta. Dessa tre är tänkt att vara de sista tre byten i den fysiska adressen. Tre tecken för att ange vem som skickar, och tre för att ange till vem.

Efter det kommer ett ASCII-tecken (command) för att ange vad man vill göra, eller vad man svarar på för förfrågan.

Den sista delen av protokollet är själva datan. Ingen längdangivelse används utan datan avslutas alltid med tecknet &. En del kommandon behöver ingen data, då avslutas protokollet bara med &-tecknet.

# or %	0 - FFF	0 - FFF	ASCII-Char		
Delimiter	DestID	SrcID	Command	DATA	&

Bild 18 Protokoll design, Dalatron Zigbee demo

3.2.3 Applikationen

Det fanns ingen tid att skriva någon PC-mjukvara för att kommunicera med noderna. Därför valdes programmet Hyperterterminal för att kommunicera med noderna. Detta var ett bra alternativ då det fanns färdigskrivna rutiner för en sådan kommunikation att ladda ner från internet. Koden är skriven samma ”anda” som stacken är skriven d.v.s. med mycket preprocessor direktiv. Den slutgiltiga koden för en cordinator och en RFD har samma main funktion. Det är beroende på vad som är definierat som avgör om koden för en cordinator, eller en RFD kompileras.

Koden är skriven så att man utför ett visst antal uppgifter. Under en uppgift har en watchdog koll på att man inte fastnar i något tillstånd.

Den första prioriteringen var att få en kommunikation mellan noderna. Det som man skrev i Hyperteminal till den ena noden, skulle sändas vidare till den andra och presenteras på dennes hyperterminal. Många av dessa funktioner hittades på nätet skrivna av DR Reese. Utgående ifrån DR Reese kod och den medföljande demokoden från Microchip implementerades applikationen.

När detta var klart och fungerade tillfredsällande implementerades protokollet.

När något skickas till en nod kollas först så att längden på vad som skickas är mins så långt som det minsta tillåtna för protokollet. Det minsta man kan skicka är 1 + 3 + 3 + 1 + 1 alltså 9 byte (se 3.2.2 Protokoll). Efter det sparas alla delar av protokollet i varsin global variabel. Beroende vilket värde delimitern har så anropas dess funktion av huvudloopen. Väl inne i en delimiter funktion så kollas slutligen vilket command man har erhålligt. Commandet avgör slutligen vilken funktion som ska anropas, och där utförs det som efterfrågades.

I nuläget är ett command för vardera delimiter impementerat. Detta är enkelt utbyggbart efter vad som efterfrågas.

3.2.4 ZMD44101

För att kunna använda ZMD44101 istället för CC2420 behövdes en del stackfiler utvecklas.

Det första som gjordes var att skriva kod för att mappa ihop ZMD44101 och processorn. Alla pinnar måste vara på rätt plats om något ska fungera.

Därefter modifierades stacken uppåt i funktions hierarkin för att få den kompatibel med ZMD44101.

Många funktioner fick skrivas om helt på nytt för att ZMD44101 skulle utföra det som stacken ville.

3.3 Testning

3.3.1 Stacken

Först kompilerades det medföljande demot under utvecklingsprogrammet IDELab från Microchip med HI-Tech pic 18 kompilatorn. Det fungerade inte. Efter kontakt med Microchips support. Svarade de att det var bara deras första version av stacken som hade stöd för HI-tech kompilatorn.

Efter det så hade Microchip lämnat supporten för den kompilatorn och har nu bara stöd för deras egna PIC18 C kompilator.

Jag kände då att det började gå åt alldeles för lång tid till detta problem. Därför bestämde jag mig för att ladda ner en gratis version i 70 dagar av Microchips kompilator, och försöka fixa kompilator problemet parallellt med det fortsatta arbetet.

Med den nya kompilatorn gick det bra att kompilera koden. De medföljande demot fungerade utan problem.

3.3.2 Applikationen

Applikationen testades allt eftersom den byggdes upp. De nya funktionera testades allt eftersom de blev skrivna.

Stacken från Microchip är under pågående utveckling, vilket medför att den kan vara ganska instabil. Det släpps hela tiden nya versioner och buggfix-filer att inkludera i stacken.

Applikationen kan tendera att låsa sig ibland när man ska skicka data och systemet måste då startas om. Detta beror troligen på ett känt fel på chipcons CC2420 som slutar att fungera. Chipcon är medvetna om detta fel, och det kommer att vara ordnat till uppföljaren till CC2420. Den slutliga applikationen fungerar tillräckligt väl för att den skall kunna användas för att demonstrera ett Zigbee system.

3.3.3 ZMD44101

Testningen av applikationen med ZMD44101 gick ej att genomföra. Det var meningen att Dalatron skulle ta fram dotterkortet med ZMD-kretsen. Men oförutsedda händelser förhindrade tyvärr Dalatron att hinna med att ta fram kortet.

Det enda som kan tillägga är att koden går att kompilera, men som alla ingenjörer vet så betyder det inte att det fungerar i praktiken. Det finns

fortfarande en del funktioner i stacken som inte riktigt går att få ZMD44101 att utföra på ett korrekt sätt.

Slutligen kan tilläggas att nu i vår (2006) så kommer ZMD att släppa uppföljaren till 44101. Det kan vara en god idé att hålla ögonen på den kretsen då ZMD har sagt att den skall vara bättre och stabilare än 44101.

4 Resultat

Detta projekt resulterade i två fungerande demokort för en Zigbee-applikation. Kod färdigskriven för ett Zigbee system för längre avstånd. En utredning om möjligheterna kring Zigbbe för längre avstånd.

5 Slutsats

Finns det någon framtid för Zigbee och långa avstånd?

Zigbee är ingen standard för några längre avstånd. Jag tror realistisk att man kan få upp räckvidden högst till ca 300 – 500 m, om man har tänkt följa standarden. Dessutom begränsar lagarna på vilka frekvenser man får sända och med hur mycket effekt.

Går man dock ifrån standarden så finns det ingen direkt gräns. Men det medför mycket mer arbete, eftersom de stora tillverkarna av Zigbee-enheter följer standarden. Man måste göra allt på egenhand och det skulle bli väldigt tidskrävande.

Alternativet är att gå ifrån Zigbee helt. Detta kan vara ett alternativ, men det kommer också här att krävas en hel del jobb om resultatet skall bli bra. Det som är själva fördelen med Zigbee, ett energisnålt nätverk som bygger och håller reda på sig själv, tar väldigt lång tid att ta fram.

Det arbete mot en slutlig produkt som genom detta examensarbete påbörjades, är det bästa som man i dagens läge kan göra, om man inte vill lägga ner väldigt mycket tid och pengar

Jag tycker att jag uppnådde det mål som låg till grund för detta projekt. Jag är väldigt tacksam över att ha fått chansen att lära mig om Zigbee, och tror att det kommer att vara väldigt bra att kunna i framtiden. Arbetet har varit väldigt inspirerande men även svårt eftersom det är en helt ny teknik. Det har känts som om att man jobbat precis bakom den internationella teknikfronten inom detta ämne.

6 Källor

6.1 Kapitel

2.1 Zigbee
(2004) *ZigBee Specification v1.0 l*, Zigbee alliance
< http://www.zigbee.org/en/spec_download/download_request.asp> 2005-12-12

Cross, Pete (2005) *Zeroing in on ZigBeel*, Circuit cellar
< http://www.circuitcellar.com/library/print/0205/Cross175/index.htm >
2005-12-12

2.2.1 Picdem Z
(2005) *PICDEM™ Z Demonstration Kit User's Guide*, Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/51524a.pdf >
2005-12-12

2.2.2 ZMD44101
(2005) *ZMD44101 Data Sheet,ZMD*
< www.zmd.de/pdf/ZMD44101_DataSheet_Mar2005.pdf>
2005-12-12

2.2.3 PIC184620
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

2.3.1 Microchip Stack for Zigbee
(2004) Microchip Stack for the ZigBee™ Protocol, Microchip
2006-01-06

6.2 Bilder

Bild 1 Olika noder i nätverkstoppologier
(2005) Circuit cellar
< http://www.circuitcellar.com/library/print/0205/Cross175/index.htm >
2006-01-06

Bild 2 Zigbee lagrer översikt
(2005) Circuit cellar
< http://www.circuitcellar.com/library/print/0205/Cross175/index.htm >
2006-01-06

Bild 3 Signal modulering
(2005) Circuit cellar
< http://www.circuitcellar.com/library/print/0205/Cross175/index.htm >
2006-01-06

Bild 4 Picdem Z 2.4 GHz Demonstration Bord

(2005) *PICDEM™ Z Demonstration Kit User's Guide*, Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/51524a.pdf >
2006-01-06

Bild 5 ZMD44101 överblick
(2005) *ZMD44101 Data Sheet,ZMD*
< www.zmd.de/pdf/ZMD44101_DataSheet_Mar2005.pdf>
2005-12-12

Bild 6 ZMD44101 kommunikatins möjligheter
(2005) *ZMD44101 Data Sheet,ZMD*
< www.zmd.de/pdf/ZMD44101_DataSheet_Mar2005.pdf>
2005-12-12

Bild 7 Exempel på en läsning och skrivning med SPI
(2005) *ZMD44101 Data Sheet,ZMD*
< www.zmd.de/pdf/ZMD44101_DataSheet_Mar2005.pdf>
2005-12-12

Bild 8 ZMD44101 Pinlayout
(2005) *ZMD44101 Data Sheet,ZMD*
< www.zmd.de/pdf/ZMD44101_DataSheet_Mar2005.pdf>
2005-12-12

Bild 9 Program memory map and stack for PIC18F4620
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 10 Special Function Register map for PIC18F4620
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 11 PIC18F4620 INTERRUPT LOGIC
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 12 MSSP Block Diagram (SPI™ MODE)
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 13 SPI™ Master/Slave Connection
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 14 SPI™ Mode Waveform (Master Mode)
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 15 SPI™ Mode Waveform (Slave Mode)
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 16 SPI™ Mode Waveform (Slave Mode)
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 17 TYPICAL ZigBee™ NODE HARDWARE
(2004) *PIC18F2525/2620/4525/4620 Data Sheet,* Microchip
< http://ww1.microchip.com/downloads/en/DeviceDoc/39626b.pdf >
2006-01-06

Bild 18 Protokoll design, Dalatron Zigbee demo
(2005) Dalatron Elecronics AB, Jerker Arnberg

6.3 Tabeller

Tabell 1 Typicall RFD application files
(2004) Microchip Stack for the ZigBee™ Protocol, Microchip
2006-01-06

Tabell 2 Typicall FFD application files
(2004) Microchip Stack for the ZigBee™ Protocol, Microchip
2006-01-06

6.4 Personer

Dr Robert Reese
Mississippi State Univerity
reese@ece.msstate.edu

Roland Anderson
VD Dalatron electronics AB
r.anderson@dalatron.se