BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
MÉRÉSTECHNIKA ÉS INFORMÁCIÓS RENDSZEREK TANSZÉK

Rádiós IC illesztése Blackfin DSP-hez

SZAKDOLGOZAT

Készítette	*Belső konzulens*	*Külső konzulens*
Horváth Ákos	Molnár Károly	Balogh László

2010

HALLGATÓI NYILATKOZAT

Alulírott Horváth Ákos, a Budapesti Muszaki és Gazdaságtudományi Egyetem hallgatója kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, és a diplomatervben csak a megadott forrásokat használtam fel. Minden olyan részt, amelyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Tudomásul veszem, hogy az elkészült diplomatervben található eredményeket a Budapesti Műszaki és Gazdaságtudományi Egyetem, a feladatot kiíró egyetemi intézmény saját céljaira felhasználhatja.

Budapest, 2010. május 14.

Horváth Ákos
hallgató

Kivonat

A vezeték nélküli rendszerek mindennapjaink részévé váltak, népszerűségüket az általuk nyújtott kényelemmel és mobilitással szerezték. A legtöbb okos telefon, laptop illetve egyéb mobil eszköz esetében ma már alapkövetelmény legalább egy fajta vezeték nélküli kommunikációs képesség megléte. A szakdolgozat egy rádiós kapcsolatot magában foglaló rendszerrel foglalkozik. A BME-MIT tanszéken rendelkezésre áll egy Chipcon gyártmányú CC2420 rádiós adóvevő és a hozzá tartozó fejlesztő csomag. A fejlesztő kártya és az eszközhöz ingyenesen járó szoftver segítségével egyszerű adatátvitelt lehet megvalósítani. Valós funkciót implementáló alkalmazásban a rádiós IC-t valamilyen társprocesszorral kell használnunk. A szakdolgozat során a feladat egy alacsony szintű rutin létrehozása a CC2420 és egy Blackfin 537-es DSP között, valamint ezek felhasználásával a rádiós kommunikáció szemléltetése. A dolgozat bemutatja a processzorok SPI portjait, és az ehhez tartozó vezérlő regisztereket. A szükséges hálózati protokollokat, és a CC2420 chiphez kapcsolódóan az IEEE 802.15.4 szabványt is feldolgozza. Végül a megvalósított rendszer kerül bemutatásra, amelyben nyugtázott adatátvitel demonstrálja a rádiós működést.

Abstract

Wireless systems have became part of our everydays. Their popularity has been gained by the mobility and comfort they provide. It is a basic requirement from today's smart phones, notebooks and other mobile devices to support at least one type of wireless communication. In my thesis I treat with a communication system including a wireless radio line. At the BME-MIT department a Chipcon made CC2420 radio transceiver is available in a development kit. With the development board and the communication software simple data transmission can be accomplished. We need to use a co-processor with the radio frequency transceiver to implement real-life functions. The task was to create a communication line between the CC2420 chip and a Blackfin 537 digital signal processor and to demonstrate radio communication functions using this link. This document describes the SPI port of the processors and their controller registers. It also presents the required network protocolls and construes the IEEE 802.15.4 standard in connection with the CC2420 Chipcon transceiver. Finally the realized system is demonstrated, in which an acknowledging function illustrates radio communication.

Tartalomjegyzék

1.Bevezetés

A beágyazott eszközökben manapság megtalálható a hálózati hozzáférés lehetősége is. Jellemző a különböző vezeték nélküli hálózatok támogatása, mint például a Bluetooth és a WLAN, melyek a felhasználóknak kényelmet és rugalmasságot biztosítanak. A szakdolgozat során az IEEE (Institute of Electronical and Electronics Engineers) 802.15.4-es vezeték nélküli hálózati szabvánnyal foglalkozok. A szabványhoz igazodik a CC2420 rádiós IC, amelyet alacsony fogyasztású vezeték nélküli alkalmazásokhoz terveztek, és amely hardveresen támogatja a 802.15.4 szabvány szerinti kommunikációt.

A szakdolgozat első négy fejezete általános áttekintést ad, majd az ezt követő három fejezetben a megvalósítást mutatom be. A hálózati alapok áttekintését követően, a harmadik fejezet már részletesen tárgyalja a 802.15.4 szabvány és a ZigBee protokoll együttesét, melyek az alacsony fogyasztású vezeték nélküli hálózatok alsóbb rétegeinek leírását tartalmazzák. A negyedik fejezet áttekintést ad a csatorna hozzáférések szabályozásáról. A szakdolgozat során feldolgozott szabvány az adatcsomagok ütközését elkerülő vivő érzékeléses protokollt alkalmazza. Az angol elnevezést rövidítő CSMA/CA (Carrier Sense Multiple Acces with Collision Avoidance) megjelölés használata elterjedt. A rádiós hálózatok gyakran használják a Master-Slave modellt, ahol a master szerepét egy nagyobb teljesítményű feldolgozó egység tölti be, például egy DSP. A két vezérlő kommunikációjának élesztéséhez az adóvevőt a DSP-hez kell illeszteni. Egy ilyen rendszer került összeállításra a BME-MIT DSP laborjában, melynek komponenseit írja le az ötödik fejezet. Az adóvevő és a DSP kommunikációjának megvalósítása az SPI porton történik. Miután a kommunikációs csatorna felépült a két eszköz között, lehetséges a DSP-n a rendelkezésre álló adatokkal bonyolultabb feladatok elvégzése is, például egy helymeghatározó algoritmus futtatása vagy csatornahozzáférések szabályozása. A felépítő egységek ismeretében a hatodik fejezet leírja az SPI kommunikáció megvalósítását a DSP és a CC2420 SPI portjainak áttekintésétől az alacsony szintű rutint alkotó függvényekig. Majd a rádiós kommunikáció demonstrálására összeállított mérési elrendezést és a mérések eredményeinek ismertetését foglalja magába a hetedik fejezet, továbbá a működés bemutatására a nyugtakeretek küldésének és fogadásának folyamatát. A dolgozat végén a munka összegzése, illetve továbbfejlesztési lehetőségek szerepelnek.

2. Hálózatok

2.1 Számítógépes hálózatok

Az olyan számítógépek, melyek egymástól függetlenül működőnek, viszont összeköttetésben vannak egymással számítógépes hálózatokat [1] alkotnak. Összeköttetés alatt azt értjük, hogy a két fél információ cserére képes egymás között. A kapcsolat létrejöhet rézhuzallal, lézersugárral, mikrohullámmal illetve távközlési műholddal is. A hálózatok nagy előnye az erőforrások megoszthatósága. Ilyen módon fizikai helytől független, nagy teljesítményű hozzáférés biztosítható. Továbbá költségmegtakarítást, azonnali adatfeldolgozást érhetünk el velük. Számítógép-hálózatok segítségével egy hatékony kommunikációs eszközhöz is jutunk, különösképpen, ha a hálózatunkat a világhálóhoz is csatlakoztatjuk. Egy hálózat kiépítése adott esetben költségigényes feladatot is jelenthet. Vezetékes hálózatoknál az összeköttetés megteremtéséhez elengedhetetlenül szükséges eszközök (hálózati kártya, hub, speciális nyolc eres ún. UTP kábel) beszerzése, beépítése, illesztése, fenntartása képvisel költséget. Vezeték nélküli hálózatoknál ezek a költségek értelemszerűen nem jelentkeznek, ez azonban nem feltétlenül jelenti azt, hogy kevésbé költségesek. A hálózat működtetési feladataihoz tartozik a több felhasználós környezet és a megosztott adatbázisok kezelése, ezt egy speciális hálózati operációs rendszer végzi. Ez a rendszerkörnyezet, nehezebben adminisztrálható, drágán működtető szoftvert igényelhet.

2.1.1 Hálózati struktúrák

A világszerte alkalmazott hálózati megoldások hatótávolságuk, adatátviteli sebességük, biztonságosságuk, topológiájuk, kiterjedésük és az adatátvitelre használt média szerint is megkülönböztethetőek. A következő csoportokat említhetjük:

- LAN (Local Area Network): lokális vagy helyi hálózatok, melyeknél a számítógépek egymáshoz viszonylag közel helyezkednek el. Két pont közötti, illetve adatszóró típusaikat említhetjük. Többféle topológia szerint is összeállíthatók, vezeték nélküli helyi megfelelője WLAN (Wireless Local Area Network).

- MAN (Metropolitan Area Network): LAN-ok közti kommunikációt teszik lehetővé. Legtöbbször nagyvárosokban fordulnak elő, erre utal az angol elnevezése is. Általában

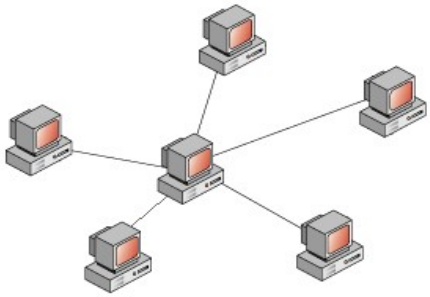
ugyanazok a jellemzők vonatkoznak rájuk, mint a LAN-okra, kivéve a kiterjedésüket. Az IEEE 802.6 –os szabványt dolgozták ki rájuk vonatkozóan.

- WAN: országot, illetve földrészt lefedő hálózat. MAN-ok és/vagy LAN-ok közti kommunikációval teszik lehetővé a nagyméretű lefedettséget. A globális vagy világméretű hálózati rendszerek, nagyszámú elemet foglalnak magukba. Ezek számítógépek illetve helyi hálózatok is lehetnek. Legjobb példája az Internet.

Egyre nagyobb teret hódítanak a PAN (Personal Area Network) hálózatok, melyek a LAN-nál kisebb kiterjedésű, személyi hálózatok. Legismertebb példája a Bluetooth.

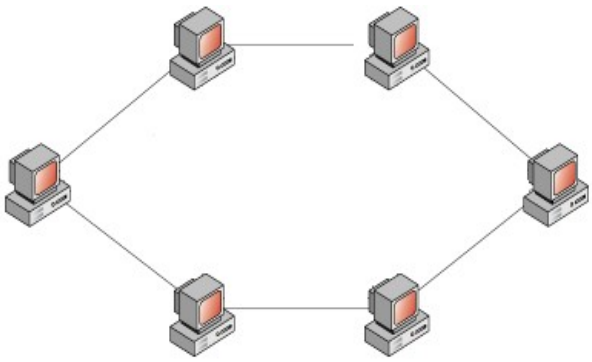
A hálózat alakját a fizikai összeköttetések és a csomópontok elhelyezkedése határozza meg. Ezt nevezzük hálózati topológiának, melynek főbb típusai:

- Csillag: centralizált kialakítás, a csomópontok egy közös elosztóba, csillagpontba (pl. egy hub) vannak bekötve, a rendszer tovább bővíthető. Nem üzenetszórásos, hanem ponttól-pontig típusú elrendezés. Szakadás esetén megbízhatóbb, viszont hátránya, hogy sok kábelt igénylő megoldás vezetékes hálózatoknál. Vezeték nélküli hálózatoknál is használják ezt a logikát.



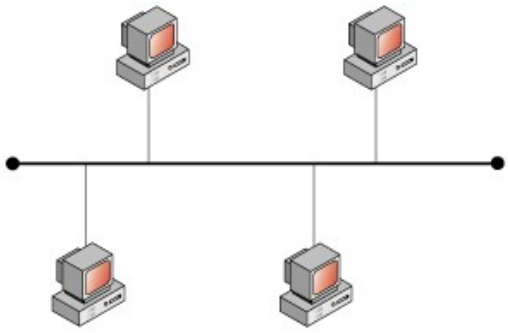
1.ábra: Csillag topológia

- Gyűrűs: soros elrendezés, melyben a csomópontok közvetlenül csatlakoznak, így egy zárt hurkot alkotnak. Gráfja egy kör, minden állomásnak két szomszédja van. Huzalozása olyan tekintetben nehézkes, hogy új csomópont hozzáadásakor illetve elvételekor meg kell bontani a hálózatot. Az adatáramlás egy kitüntetett irányba történik. Amíg az adatot nem mentik le, addig a gyűrűben kering, tárolódik, tehát üzenetszórásos rendszert alkot. Az adatok károsodását megelőzendő a címzettnek mielőbb le kell menteni és nyugtázni a vételt.



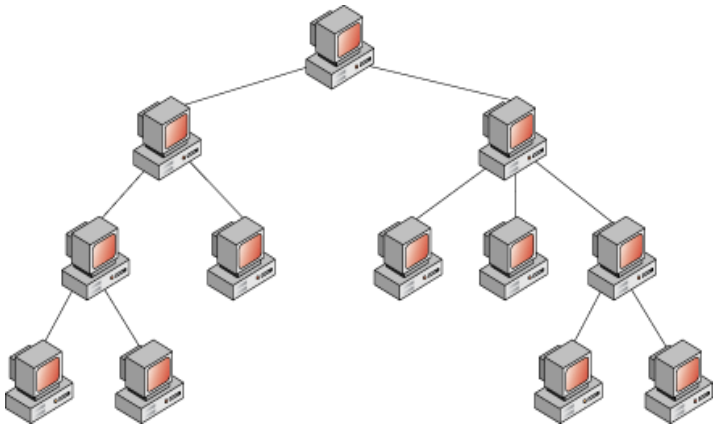
2.ábra: Gyűrű topológia

- Busz: sín- illetve lineáris elrendezésként is ismert, sorba fűzött gépek alkotják. Gyakori a helyi hálózatokban, mivel olcsó a kialakítása. Ha egy adó üzenetet küld a buszra, azt mindenki hallja, üzenetszórásos rendszer. A busz végeinek hullámimpedanciával történő lezárása következtében nem lép fel reflexió, ez megfelel a végtelen hosszú vezeték esetének. Az elrendezésből adódó hátránya, hogy a kábel megbontása, a hálózat működésképtelenségét okozza.



3.ábra: Busz topológia

- Fa: egy körmentes gráfot alkotó elrendezés, azaz két csomópont között csak egy útvonal létezik. Előnye, hogy meghibásodás esetén csak a csomópont és a hozzátartozó gyökerek esnek ki, pont-pont összeköttetést valósít meg.



4.ábra: Fa topológia

Szabálytalan elrendezésnek nevezzük, ami e fentiekbe nem besorolható.

Ha a végrehajtandó feladatok, illetve az erőforrások megosztása szerint vizsgáljuk a hálózatokat, akkor két modellt különböztethetünk meg:

- Szerver-kliens architektúra: az elterjedtebb esetet képviselik. Van egy kitüntetett szereppel bíró állomás, amely a hálózat vezérlését és felügyeletét látja el. A munkaállomások, azaz a kliensek a szerverhez kapcsolódva, annak irányításával használják a hálózatot. Ezek a szerver-kliens felépítésű hálózatok erősen centralizált jelleget mutatnak. Előnye, hogy a kommunikáció ellenőrzött, illetve kliens lehet kis teljesítményű számítógép is. Hátránya, hogy ha a szerver meghibásodik, akkor megszűnik a hálózati szolgáltatás.

- Peer-to-peer hálózatok: más néven egyenrangú hálózatok. Hálózati szempontból minden gép kliens és szerver is lehet. Előnye, hogy nem kell fenntartani egy költséges szervert. Hátránya, hogy a kliensek több erőforrással rendelkeznek, akkor is, ha ez nem indokolt, mivel szerverként is működhetnek. További hátrány, hogy a peer-to-peer hálózatok sebezhetőbbek.

Használhatunk mindkét megoldást támogató hálózati rendszereket egyazon számítógépről is, köszönhetően a hálózati protokolloknak és megoldásoknak.

Attól függően, hogy a hálózati állomások közötti összeköttetések segítségével egyszerre egy-egy állomás vagy esetleg egyszerre több állomás között folyhat adatcsere, két típusú

összeköttetést különböztetünk meg:

- adatszórásos: egyszerre több állomás is képes fogadni ugyanazt az információt
- pont-pont közötti rendszereket: adatcsere egyidőben csak két fél között valósítható meg.

Az adatszóró hálózatokat tovább csoportosíthatjuk a csatorna hozzárendelés módja szerint. Ilyen tekintetben megkülönböztetünk statikus és dinamikus hozzárendelésű hálózatokat, melyek közül a kérés alapján történő, dinamikus csatornakiosztás az elterjedtebb.

Egy konkrét példát tekintve vegyük az Ethernet sín illetve csillag topológia szerint kiépített helyi hálózatot. Leggyakrabban ezt a megoldást használják LAN-ok kialakítására. Ez esetben a hálózatnak van egy gerince (BackBone - közös adatátviteli vonal), amihez az összes csomópont csatlakozik. A gerinc mindkét vége ellenállással van lezárva, a rendszer elemei sorba vannak fűzve egy kábelre. Minden csomópontnak egyedi címe van. Olcsó, kevés kábel kell hozzá. Hiba esetén az egész hálózat működésképtelen lesz. Mindezt az IEEE 802.3-as szabványban rögzítették. Az ilyen hálózatban bármely gép bármikor adhat, tehát létrejöhet adatütközés. Ilyenkor az adatokat egy megfelelő várakozási idő után újra kell küldeni. Ezen probléma kezelésére használjuk a hálózati hozzáférési eljárásokat, mint például az ütközés elkerülő CSMA/CA illetve az Ethernet által használt ütközés detektáló CSMA/CD (Carrier Sense Multiple Acces with Collision Detection) eljárás. Az elméleti maximális adatátviteli sebessége 10 MBit/s vagy 100 MBit/s, ez az összeköttetés módjától függ. A valóságban ez kisebb érték az egyéb hálózati adatforgalom miatt. Minden olyan számítógép, amely Ethernet hálózatra van kapcsolva fizikailag egyetlen kábelezésen forgalmaz adatot. Ennek következménye, hogy a hálózati kommunikációban résztvevő számítógépek számának növekedésével az átvihető adatok mennyisége egységnyi idő alatt csökken. Drágább technológia alkalmazásával ez kiküszöbölhető, azonban átlagos hálózati forgalom mellett az Ethernet jól használható. Az Ethernet hálózatokat 50 Ohm ellenállású koax kábelekkel építik ki, illetve árnyékolatlan (ritkábban árnyékolt) csavart érpárú kábelezéssel is. A topológiát a routerek és switchek elhelyezésével alakíthatjuk ki.

2.1.2 Hálózati protokollok és a keretek

Egy számítógépes hálózatról legtöbbször nem mondható el, hogy ugyanazon típusú gépek vannak benne. Ez kompatibilitási gondokat vethet fel, amit a hálózat tervezésekor figyelembe kell venni a

megfelelő szabványok és protokollok alkalmazásával. A probléma egyszerűsítése érdekében a hálózati kommunikáció során elvégzendő feladatok rétegekbe (layer-ekbe), vagy szintekbe szerveződnek, melyek egymásra támaszkodó rendszert alkotnak. A felsőbb rétegek számára az alsóbb rétegek működésének részletei rejtettek, fekete doboznak tekinthetők. Minden réteg értéknövelő szolgáltatást nyújt a felette lévőnek. A kommunikáció során az azonos szinten lévő aktív résztvevők (entitások) különböző szabályokat és konvenciókat alkalmaznak, melyek összességét hálózati protokollnak nevezzük. Az egymással szomszédos rétegek közötti interfész a két réteg közötti elemi műveleteket és szolgáltatásokat tartalmazza. Az adatok nem közvetlenül jutnak át az adótól a vevőhöz. A rétegek egymásnak adják át az információt addig, amíg az a legalsó réteghez nem ér. A valós kommunikáció a fizikai rétegen zajlik, majd a vevő oldalán pedig a rétegeken felfelé haladva jut el az információ a célba. Ez a folyamat egyfajta beágyazásnak majd ezt követően kibontásnak tekinthető (packing – unpacking). A rétegek és protokollok alkotják a hálózat architektúráját. Az architektúra kialakítása során figyelembe kell venni az adatátvitel szabályait, ami lehet egyirányú (szimplex), váltakozóan kétirányú (fél-duplex) vagy egyszerre kétirányú (duplex). Szimplex átvitelnél az információ egy irányba terjed, fél-duplex megoldásnál szintén, viszont itt változtatható az irány. Duplex átvitelnél egy időben kétirányú kommunikáció valósítható meg.

Protokollok alkalmazásával többféle hálózati kommunikációt igénybe vevő programot használhatunk. Az ARPANET nevű kísérleti hálózat részére fejlesztették ki a TCP/IP-t (Transmission Control Protocol/Internet Protocol), a cél a megbízhatóság illetve több hálózat zökkenőmentes összekapcsolása volt. Az átvitelvezérlő protokoll, azaz a TCP oldja meg a hibamentes átvitelt két gép között az interneten. Az IP meghatározza a küldött csomagok formátumát.

A fenti szabályok alkalmazásával átvitt adatok feldarabolva, több részletben kerülnek továbbításra. Az információ úgynevezett keretekbe (frame-ekbe) foglalva áramlik. Alapvető kerettípusok az adat- , menedzsment- , kontrollframe-ek. Az adatkeretekben található a hasznos adat, a kontrollkeretek az adatcsere felügyeletére szolgálnak (pl.: ACK), a menedzsmentkeretek a hálózati kapcsolatok karbantratására szolgálnak (pl.:autentikációs keretek). A keretek tartalmazzák többek között a címzettre vonatkozó információt is, legtöbbször a fejlécükben. Egyéb, a feldolgozásra vonatkozó információ is megtalálható a keretben, mindezeket a megfelelő sorrendben kell küldeni. A teljes adathalmazt csomagonként kapja meg a címzett, majd a darabokból összeállítható a teljes állomány. Ezzel lehetőség nyílik arra, hogy egy címzetthez tartozó két keret között egy újabb címzettnek szánt keretet gond nélkül átvigyünk. Így tulajdonképpen ”egy időben”

történik a kommunikáció két vonalon is. Tegyük fel, hogy egy adó több száz MB-nyi információt küld el, miközben egy másik adó mindössze néhány KB átvitelére lenne szüksége a csatornára. A fenti leírás értelmében nem kell megvárnia, míg az élő kapcsolat véget ér és felszabadul a csatorna, közbeékelve is küldhet információt. További előnyünk származik a keretek használatából a kommunikáció minősége szempontjából. A rézkábelen átvitt adatok megsérülhetnek, ilyenkor viszont nem kell a teljes állományt újraküldeni, elég mindössze a hibát tartalmazó keretet újraküldeni. Ezzel időkiesést kerülünk el. A keretek általános felépítését, valamint az Ethernet keretformátumát az 5. ábra szemlélteti.

Az átlag felhasználó ezekről a háttérben zajló kommunikációs megoldásokról nem tud, számára elég annyi információ, hogy az adatok átvitele bitre pontosan lezajlott.

5.ábra: Keretformátumok

2.1.3 Az OSI referenciamodell

A Nemzetközi Szabványügyi Szervezet (International Standards Organization) dolgozta ki az OSI (Open System Interconnection) hivatkozási modellt, ami a protokollok szabványosítása felé tett első lépés volt. Ennek köszönhetően a nyílt rendszerek összekapcsolására irányuló szabványnak eleget tevő rendszerek képesek egymással együtt működni eltérő protokoll szabályok esetében is. Az ajánlás hét rétegből áll, ezek független, de egymásra épülő layer-ek (6.ábra). Ezek további jellemzője, hogy különböző absztrakciós szinteket képviselnek, jól definiált feladatot hajtanak végre. A modell tehát nem hálózati architektúrát határoz meg, hiszen protokollokat nem ír elő, csak azt mondja meg, hogy az egyes rétegeknek milyen feladatokat kell végrehajtaniuk. Alulról felfelé haladva a rétegek a következők:

- Fizikai réteg (physical layer): az információ megjelenési formája változó, feszültség érték,

vagy a feszültség változásának iránya is lehet. A hordozó közeg is többféle lehet, példaként elektromos kábelt, fénykábelt, rádióhullámot is említhetjük. A fizikai réteg feladata, hogy a biteket továbbítsa a kommunikációs közegen, ami az egyik oldalon logikai egyes, az a másik oldalon is egyes legyen. Itt zajlik a tényleges, fizikai kommunikáció. Meghatározandó, hogy a logikai értékeknek milyen a feszültségszintek felelnek meg, mennyi ideig fog tartani egy bit átvitele, hogyan épül fel és bomlik le a kapcsolat. A fizikai réteg kérdéskörébe tartozik a csatlakozók kialakítása illetve a kódolás (pl.:RTZ, NRZ, Manchester 2, differenciális Manchester) megválasztása is.

- Adatkapcsolati réteg (data link layer): a fizikai szint szolgáltatásainak igénybevételével a hálózati réteg számára hibáktól mentes átvitelt kell biztosítania. Ezt az adatok adatkeretté tördelésével (legtöbbször néhány száz bájt) oldja meg. Az információn kívül kiegészítő információ szállítására is szükség van (például melyik állomás küldi melyiknek, mit kell tennie a kerettel). Ezeket megfelelő sorrendben kell elküldeni, majd a visszaigazolásra, nyugtásra várni, azt feldolgozni. Tehát az adatkapcsolati réteg meghatározza a kommunikáció adategységének, a keretnek a felépítését, határait. Ennek a rétegnek kell megoldania az elveszett vagy sérült keretekkel kapcsolatos problémákat is.

- Hálózati réteg (network layer): az adatkapcsolati réteg szomszédos állomások közötti átviteli képességére alapozva megoldja a csomagok eljuttatását az adótól a vevőig. Azaz megtervezi az információ útvonalát a hálózaton. Ez történhet statikus módon, előre rögzített útvonalak alapján, illetve dinamikusan. A torlódásvédelem megoldása, illetve eltérő protokollokkal rendelkező hálózatok összekapcsolása is ennek a rétegnek a feladata. Üzenetszórásos rendszerekben az útvonal kiválasztás igen egyszerű, ilyenkor a hálózati réteg gyakran nem is létezik.

- Szállítási réteg (transport layer): feladata a viszonyrétegtől kapott információ eljuttatása a hálózati rétegnek illetve az esetleges hibák javítása. Az adatok feldarabolva kerülnek továbbításra. Így biztosítva végponttól végpontig terjedő megbízható átvitelt. Mindezt a viszonyréteg számára átlátszóan kell végrehajtania.

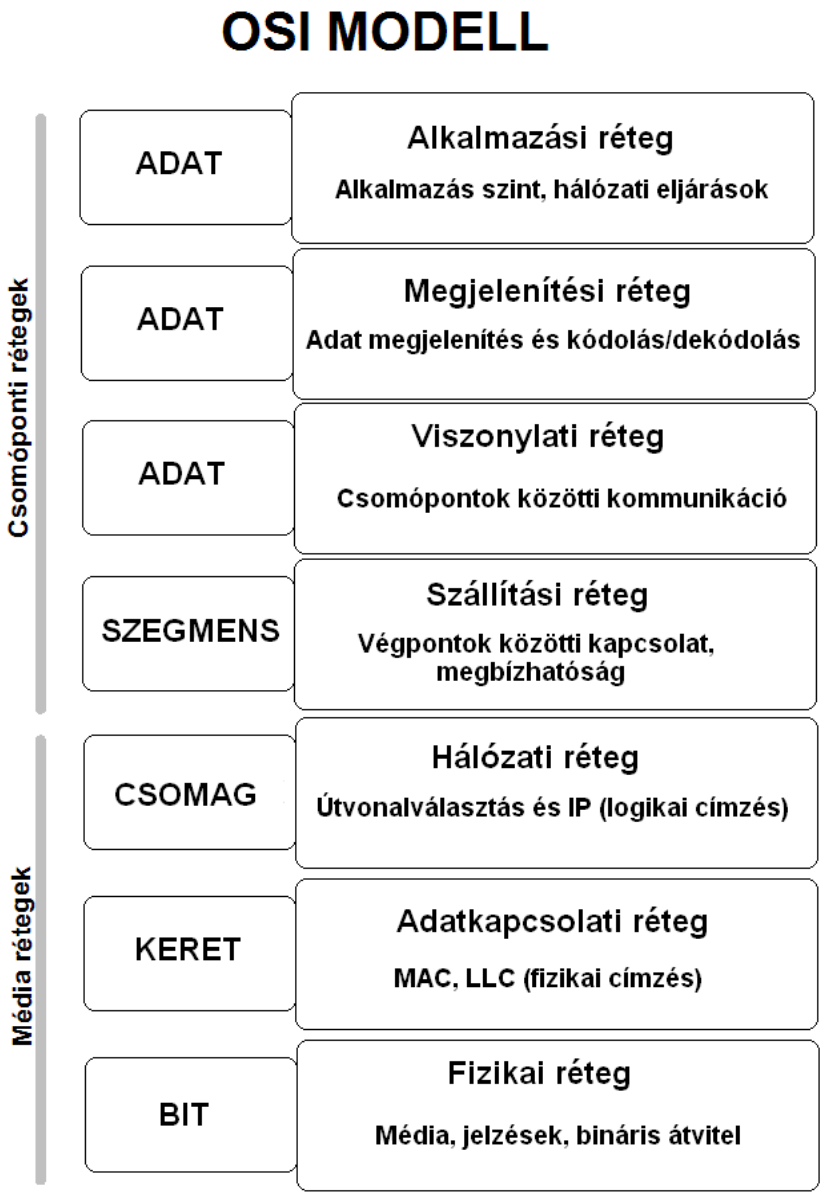
- Viszony réteg (session layer): lehetővé teszi, hogy felhasználók viszonyt létesítsenek

egymással. Adatátvitelt tesz lehetővé, néhány kiegészítő szolgáltatás mellett. Ilyen például fájlmozgatás két gép között, távoli rendszerbe való belépés, párbeszéd szervezés. További szolgáltatása a szinkronizáció, illetve a vezérjelkezelés (token management).

- Megjelenítési réteg (presentation layer): tipikus feladata a bitek kódolása szabványok szerint (pl.: ASCII, Unicode). Az alsóbb rétegektől eltérően ez a réteg a bitek szintaktikájával és szemantikájával foglalkozik. Magában foglalja az adattömörítést, és a kriptográfiát is.

- Alkalmazási réteg (application layer): különböző hálózati szolgáltatások protokollját tartalmazhatja. Ez a réteg már szorosan kapcsolódik a felhasználóhoz. Ide tartozik az állománytovábbítás mellett például a levelezés és internetes szolgáltatások.

Egyes rétegek több feladatot látnak el, például az adatkapcsolati rétegbe nagyon sok funkció került az OSI modellben. Ezért két alrétegre bontották, az alsó a MAC (Media Access Control) réteg, a felső az LLC (Logical Link Control) réteg. Az LLC alréteg feladata a forgalom szabályozás (flow control), a hibajavító kódolás, a nyugtázás, és az ismétlés kérése adott esetben. A MAC alréteg feladata meghatározni, hogy adott pillanatban az állomások közül melyik adhat a csatornán.



6.ábra: Az OSI hét rétegű modell

A gyakorlatban az OSI modell nem terjedt el, inkább a TCP/IP modell kapta a nagyobb szerepet, de mint hivatkozási modell jól használható.

2.2 Vezeték nélküli hálózatok

A hétköznapi életben fokozottan jelenlévő PDA-k, laptopok és egyéb hordozható gépek hatásai közé tartozik a vezeték nélküli hálózatok elterjedése [2]. A WLAN elnevezéssel is egyre többször

találkozunk.

A rádiós hálózatok különböző frekvenciájú hullámokat használnak adatátvitelre. A rádióhullámok több előnnyel is rendelkeznek, például könnyen előállíthatók, és képesek nagy távolságokba elérni. Áthatolnak épületek falain is, tehát a felhasználás helyének megválasztása szabadabb, mivel a rádióhullámok minden irányban terjednek az adó- és vevőkészülékek elhelyezése is szabadabban történhet. Alacsony frekvencián jobban áthatolnak a fizikai akadályokon, viszont a jel teljesítmény lecsökken a forrástól távolodva, méghozzá $1/r^3$ szerint. Ez az antenna úgynevezett távoltere. A nagyfrekvenciás rádióhullámok terjedési tulajdonságai ettől eltérnek, egyenes vonal mentén terjednek, és a tárgyakról visszaverődnek. Hátrány, hogy a rádióhullámokat egyéb elektromos berendezések zavarják.

A jelet az antenna sugározza a vevőnek a közegen keresztül, illetve az antenna feladata a küldött jel befogása is. Többféle alakú és méretű kivitele létezik, elektromos szempontból fontos az antenna lefedése, iránykarakterisztikája (propagation pattern), továbbá az antenna teljesítmény (radiation power).

A jelenleg létező vezeték nélküli hálózati megoldások közül optikai úton létrejött kapcsolatot használnak például a Bluetooth egyes változatai, illetve az Infra és a Laser technológia. Rádiócsatornát használ a Bluetooth, a HiperLAN/2, egyes PAN (Personal Area Network) megoldások, a GSM, GPRS, EDGE, HSDPA az IEEE 802.11 és változatai illetve az IEEE 802.16-os szabvány.

2.3 Vezeték nélküli hálózati megoldások, szabányok és jellemzőik

A ma használt vezeték nélküli megoldások közül a fontosabbak közé sorolhatóak a következők [3]:

- Bluetooth : a Bluetooth Special Interest Group fejlesztette, legfőképp alacsony fogyasztású eszközökhöz. Maximum 8 eszköz alkothatja az így felépített hálózatot. A hatótávolsága rövidnek mondható, 10 méter alatti érték. Számítógép és mobil eszközök között a legjellemzőbb. Sebessége 1 MBit/s, a lassabb hálózatok közé sorolhatjuk. A Bluetooth egy PAN megoldás, ami az ISM sávban működik.
- HiperLAN/2 (High Performance Radio LAN) : az European Telecommunications Standards Institute (ETSI) fejlesztése. Hatótávolsága már nagyobb, 100 méter maximálisan. Átviteli sebessége már gyorsabb, 54 MBit/s. Az ISM és az UNII frekvenciasávban is működik.
- Az IEEE 802.11-es szabványcsalád tagjai és jellemző adataik:
  Ez a szabvány a 2,4 illetve 3,6 valamint az 5 GHz-es frekvenciákon üzemelő hálózatok

specifikációját írja le [3]. A szabványért az IEEE 802 LAN/MAN Standards Comittee csoport a felelős. Az első vezeték nélküli szabvány 1997-ben jött létre, mára már elavult, megjelölése a 801.11. Levegőbeli átviteli sebessége 1 és 2 MBit/s között volt hibajavító kóddal együtt a 2,4 GHz-es frekvenciasávon. Ezt követte a 802.11a szabvány 1999-ben. Ez már 5 Ghz-en biztosított 54 MBit/s maximális átvitelt, ami a gyakorlatban jellemzően 20 MBit/s-et jelentett. Ez a változat már világszerte elterjedt, legfőképp irodai illetve otthoni alkalmazásban. Hatótávolsága 15 és 30 méter között mozgott. Hamar követte a 802.11b szabvány, amelynél a frekvencia 2,4 GHz-re csökkent, ami a hatótávolság növekedését eredményezte, ami 30-90 méter közötti érték volt. Az elméleti adatsebességek 1 és 11 MBit/s között változtak függően a környezettől. Megemlítendő azonban, hogy a 2,4 GHz-es sávot más protokollok is használják, így igen lehet nagy az interferencia. A 802.11g változat 2003 nyarán jelent meg, amely 2,4 GHz frekvencián biztosít 54 MBit/s-os adatátvitelt. A gyakorlatban 20-22 MBit/s a jellemző érték. Nemrég megjelent változat a 802.11n, melyet megnövekedett hatótávolság illetve sebesség jellemez (540 MBit/s elméleti határ). Ilyen hálózatoknál több antennán történik a vétel és az adás (MIMO= Multiple Input Multiple Output). A 802.11 szabványcsaládra épülő hálózati eszközök összekapcsolásából épülő rendszer a WLAN. A meglévő vezetékes hálózatok lecserélésére illetve kiegészítésére is alkalmazhatók. Ezek a hálózatok tartalmaznak hozzáférési pontokat (Access Point), és kliens készülékeket (Station). Ez a két építőelem hoz létre egy BSS-t (Basic Service Set), amelyek a WLAN hálózatok alapelemei. Egy hozzáférési pont által lefedett területet a BSSID (Basic Service Set Identifier) segítségével lehet azonosítani. Ezzel a címmel kommunikáló kliensek egy BSS alá tartoznak.

A 802.11-es szabvány két fizikai hozzáférési módot határoz meg, a DSSS (Direct Sequence Spread Spectrum)-t és az FHSS (Frequency Hopping Spread Spectrum)-t, ezek a modulációk szétkenik a spektrumot a frekvenciatartományban. Alap hozzáférési eljárása a DCF (Distributed Coordination Function) kombinálva a CSMA-CA (Carrier Sense Multiple Access / Collision Avoidance) eljárással.

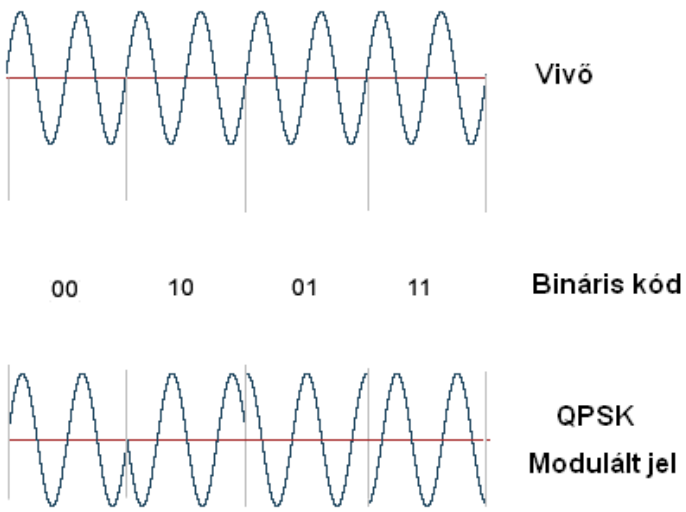
Szintén a CSMA/CA algoritmust használja a 802.15.4 -es szabvány, amely alacsony fogyasztású, kis hatótávolságú rádiós kommunkiációt ír le.

Korábban megoldhatatlan problémák megoldására képesek a vezeték nélküli szenzorhálózatok. Mind az egészségügyben, mind az ipari irányítástechnikában, főként olyan

helyeken, ahol vezetékezést nem lehet alkalmazni. Ilyen rendszereknél többnyire követelmény az alacsony fogyasztás, viszont általában kielégítő egy alacsonyabb átviteli sebesség. Erre a célra is léteznek szabványok, például az IEEE 802.15.4, illetve a ZigBee Alliance fejlesztése, a ZigBee. A vezeték nélküli szenzorhálózatok mote-oknak nevezett rádiós vezérlőkártyákból épülnek fel(pl.: Berkeley mote, MITMÓT).

3. Az IEEE 802.15.4 szabvány és a ZigBee

Néhány alkalmazás számára elegendő néhányszor 10 KBit/s átviteli sebesség alacsony fogyasztás és kis erőforrásigény mellett. Ilyen megoldások számára nyújt lehetőséget az IEEE802.15.4 [3,5] szabvány fizikai és adatátviteli rétegeket definiálva. Az adatátviteli sebesség jellemzően 20-250 KBit/s között mozog az ilyen rendszereknél, a hatótávolság 30-50 méter körüli. Az átviteli közegért versengés folyik, a szabvány CSMA/CA alapú csatorna hozzáférést alkalmaz, csillag topológiát és peer-to-peer hálózatok létrejöttét támogatja. A szabvány két fizikai réteget specifikál: egy a 868 MHz/915 MHz-es sávban működő BPSK (Binary Shift Keying) modulációt alkalmazó és egy a 2.4 GHz-es sávban működő O-QPSK(Offset Quadrature Phase Shift Keying) modulációs eljárást használó fizikai réteget.



7.ábra: QPSK moduláció

Az utóbbi 250 KBit/s névleges sávszélességgel rendelkezik, míg az előbbi 20-40 KBit/s névleges sávszélességre képes.

A spektrum szétterítésével, méghozzá a DSSS (Direct Sequence Spread Spectrum) kódolás előírásával javít a szabvány a szimbólumközti áthalláson. A DSSS egy pszeudó-zajjal, azaz zajszerű spektrumú előre ismert bináris számsorozattal XOR-ozza a jelet. A csatornán az így kapott közel fehér spektrumú jelet küldjük át. A vevő oldalon szintén ismert a kód, így a vevő egy újabb XOR művelettel visszakapja az eredeti jelet, kevés zajjal terhelve. Ehhez a vevői oldalon még a vivőt és

az órajelet vissza kell állítanunk. A szabvány a csomaghibákra (Packet Error Rate) 1 %-ot, viszonylag nagy értéket enged meg. A használt fizikai réteg a különböző országok korláltozásainak függvényében változhat (8. ábra).

	FREKVENCIA SÁV	LEFEDETTSÉG	ADATSEBESSÉG	CSATORNÁK SORSZÁMA
2.4 GHz	ISM	Világszerte	250 kbps	11-26
868 MHz		Európa	20 kbps	0
915 MHz	ISM	Amerika	40 kbps	1-10

8.ábra: Az IEEE 802.15.4 fizikai definíciói

A ZigBee protokoll erre a szabványra épül [4]. Az elnevezés a ternészetből ered. A méhek (Bee) mozgásuk során egy cikk-cakk (Zig-Zag) mintát írnak le. A teljes kolóniában képesek egymással információt megosztani, mint például egy újonnan felfedezett táplálék forrás iránya, távolsága illetve helye. A Bluetooth remekül megállja a helyét a nagyobb átviteli sebsségű rendszerekben, például beszédátvitel esetében. A ZigBee technológia ott lép életbe ahol nem szükséges nagy átviteli sebesség ugyanakkor fontos a hatékony telepkihasználás, változtatható topológia, valamint a kismértékű felhasználói beavatkozás igénye. Az 9. ábra mutatja a különböző vezeték nélküli technológiák legfontosabb jellemzőit:

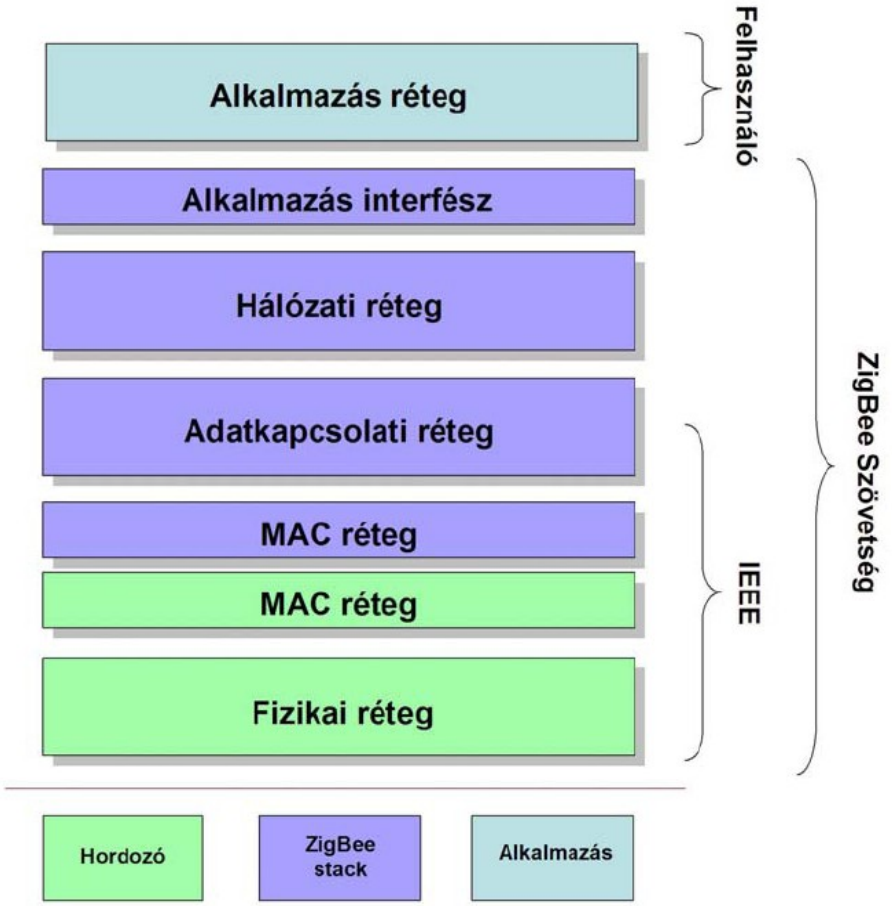
Market Name	ZigBee™	---	Wi-Fi™	Bluetooth™
Standard	802.15.4	GSM/GPRS CDMA/1xRTT	802.11b	802.15.1
Application Focus	Monitoring & Control	Wide Area Voice & Data	Web, Email, Video	Cable Replacement
System Resources	4KB - 32KB	16MB+	1MB+	250KB+
Battery Life (days)	100 - 1,000+	1-7	.5 - 5	1 - 7
Network Size	Unlimited (2^{64})	1	32	7
Bandwidth (KB/s)	20 - 250	64 - 128+	11,000+	720
Transmission Range (meters)	1 - 100+	1,000+	1 - 100	1 - 10+
Success Metrics	Reliability, Power, Cost	Reach, Quality	Speed, Flexibility	Cost, Convenience

9.ábra: Vezeték nélküli technológiák

A ZigBee alkalmazásoknál a hatékony időzítésnek köszönhetően elmondhatjuk, hogy a ZigBee eszközök nagyon hamar tudnak csatlakozni a médiához, adatokat továbbítani azon, lekapcsolódni a csatornáról, illetve gyorsan tudnak alvó állapotba kerülni. Ezáltal biztosítják a telepek kapacitásának kímélését. A ZigBee állomások az optimalizált telepfelvételnek köszönhatően akár egyetlen átlagos ceruzaelemmel is több évig képesek működni.

A ZigBee protokoll nyílt szabvány az alacsony teljesítményű vezetéknélküli hálózatokhoz. Az IEEE802.15.4 szabvány az alacsony átvitelű WPAN-okra vonatkozik, az alacsony szintű protokoll rétegek közül definiálja a fizikai réteget és a közeghozzáférési réteget, a különböző szintek között meglehetősen nagy biztonságú adatátvitelt biztosítva. A ZigBee a protokoll stack efeletti rétegeire vonatkozóan tesz előírásokat a hálózati rétegtől egészen az applikációs rétegig. Az ilyen hálózatokra jellemző kis fogyasztás miatt a hatótávolság szintén alacsony. Az LR WPAN (Low Range Wireless Persoal Area Network) elnevezés innen ered.

Fontos tényező a megfelelő biztonság kialakítása, mivel ellenkező esetben más eszközök lehallgathatják küldött adatainkat. Az IEEE 802.15.4 szabványban definiálva vannak az authentikációs folyamatok illetve a titkosítási szintek. A fejlesztők kiválaszthatják, hogy milyen fokú titkosítást szeretnének alkalmazni. A titkos kulcsok hálózaton történő továbbítása a ZigBee feladata, melynek segítségével biztonságosan lehet távolról is felügyelni a rendszerek működését. Egyes alkalmazásokban az adatbiztonság kevésbé fontos (pl. hőmérő). Ilyenkor a rendszer biztonságossága megválasztható. Kevesebb biztonsági képesség implementálásával javítani lehet a telepek élettartamán. Ellenben ipari vagy katonai alkalmazások esetén, ahol nagyon fontos az adatok sértetlensége, ott a biztonsági paraméterek garanciája prioritást élvez.



10.ábra: A protokoll stack felépítése

A ZigBee és a felette elhelyezkedő IEEE802.15.4 szabvány a rendszertervezők számára két eltérő képességű eszközök használatát engedélyezi:

- FFD (Full Functionality Device): teljes működési képességekkel rendelkező eszköz, amely

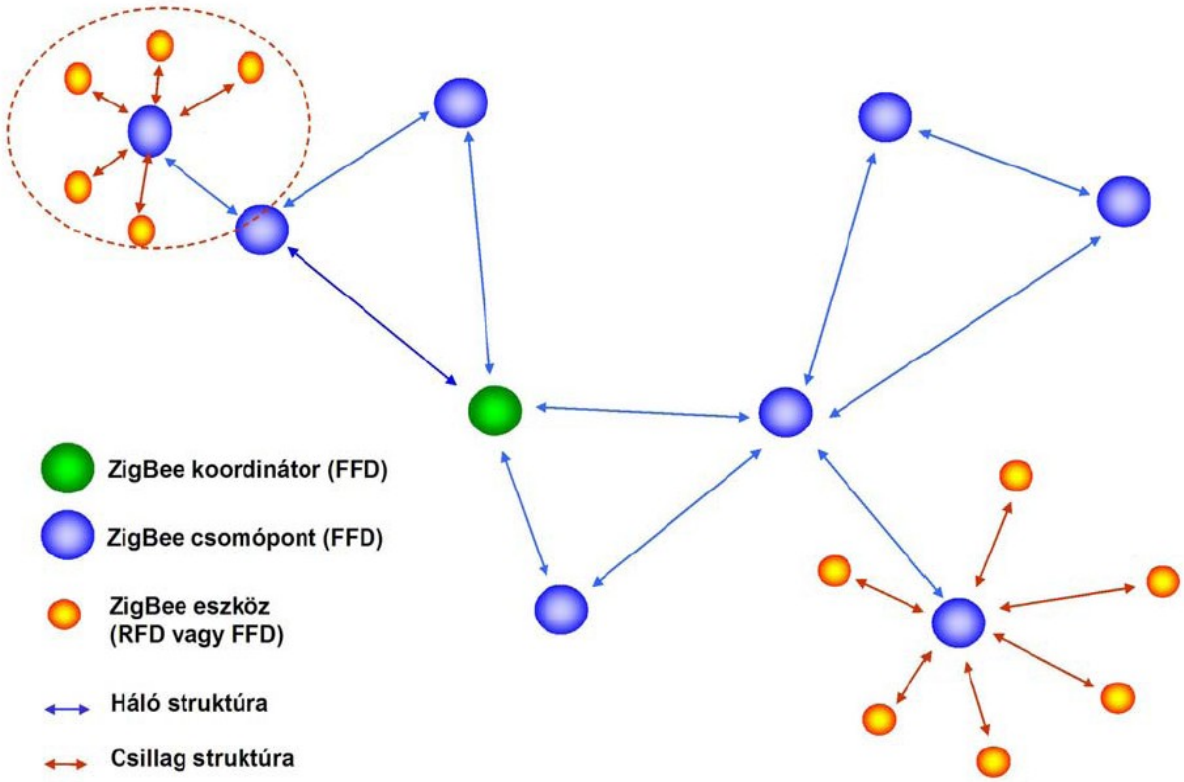
koordinátorként is képes üzemelni.

- RFD (Reduced Functionality Device): csökkentett működési képességekkel rendelkező eszköz, amely nem képes koordinátorként üzemelni.
- Network Coordinator: hálózat felügyeleti eszköz, feladata a hálózat folyamatos szervezése, beacon üzenetek továbbítása, hálózati csomópontok irányítása, illetve a párosított eszközök közötti üzenetek irányítása.

Minden ZigBee hálózat rendelkezik egy RFD-vel vagy FFD-vel, illetve egy Network Coordinator-ral. A hálózati csomópontok teleppel működnek és az energiatakarékosságra törekednek. Figyelik az elérhető hálózatokat, a hozzájuk kapcsolódó eszköztől továbbítják a hálózatba az üzeneteket. ZigBee hálózatokban FFD működési osztályba tartoznak a hálózati kordinátorok és a hálózati csomópontok is.

Az IEEE 802.15.4 hálózatokban topológiai szempontból az egyik megoldás a csillag elrendezés, és az ilyen hálózatokat összefogni képes fa-topológia, továbbá a klaszter-fa (cluster tree) elrendezés. A csillag forma a Master-Slave rendszerekhez ideális, a hálós modell peer-to peer alkalmazásokhoz, míg a klaszter-fa rendezett, de Master nélküli hálózati elrendezésekhez.

A ZigBee hálózatok többnyire háló-struktúra (mesh) szerint épülnek fel, ami rugalmasabb útvonalválasztást tesz lehetővé. További előnye, hogy egyes csomópontok kiesése vagy meghibásodása könnyebben kiküszöbölhető.

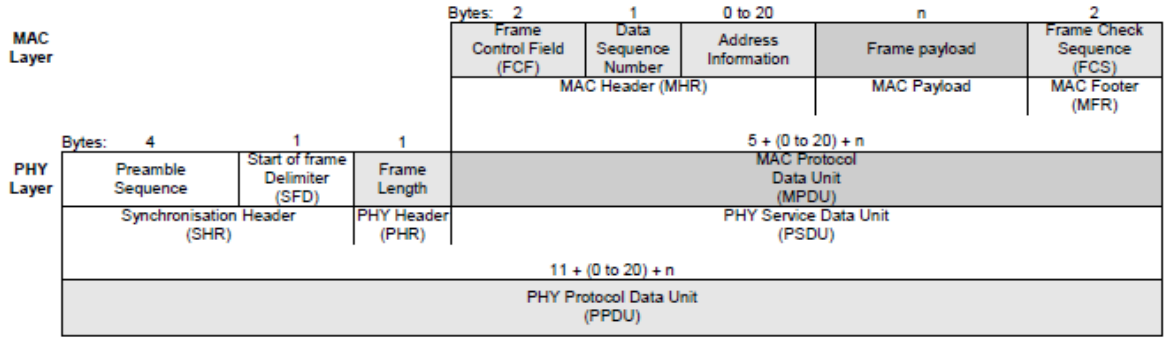


11.ábra: ZigBee hálózati struktúra

A keretszerekezetet vizsgálva két fő kialakítási szempontot fedezhetünk fel. A bonyolultság minimális szinten tartása mellet fontos, hogy zajos csatornában is tudjunk adatokat továbbítani. Az IEEE802.15.4 négy különböző kerettípust definiál. Ezek a következők: Beacon, adat, nyugta , MAC felügyelet. Adatcsomagok vétele után a vevő egy CRC ellenőrzést hajt végre. Amennyiben mindent rendben talál, nyugtát küld, amennyiben hibát észlel a csomag dobása mellett elmulasztja a nyugtaküldést. Lehetőség van úgynevezett szuperkeret használatára is, amelyet a hálózati koordinátor küld ki a hálózatba és a többszörös hozzáférés okozta problémák elkerülésére használjuk. Egy ilyen keret 16 egyenlő időrést tartalmaz. A beacon üzenetet mindig egy ilyen szuperkeret elején küldi ki. Az üzenni kívánó felek a versengéses időszakban (Contention Access Period) a CSMA/CA (Carrier Sense Multiple Acces with Collosion Avoidance) szabályai szerint férnek hozzá a csatornához. Ezt az eljárást a következő fejezetben részletesen megismerhető. A szuperkeretek végén bizonyos eszközök számára lehetséges fix időrés (Guaranteed Time Slot) dedikálása, ez a versengés mentes időszak (Contention Free Period). A versengéses időszakban a kommunikáló feleknek azt is figyelembe kell venniük, hogy a versengésmentes időszak előtt be kell fejezniük az adási szándékukat. Ugyanígy a versengés mentes periódusban adó eszközöknek

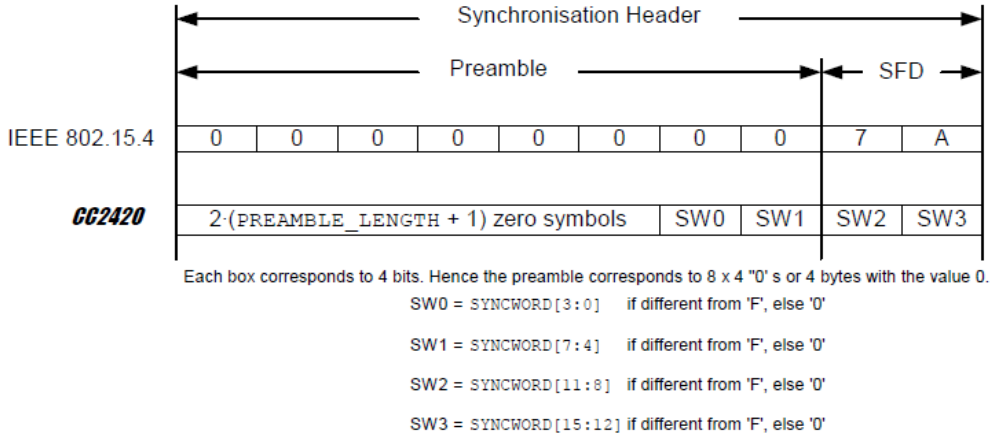
biztosítaniuk kell, hogy a nekik kirendelt időablakokban vagy legkésőbb a szuperkeret versengésmentes időszakának végéig befejezik az adásukat. A fentiek alapján megkülönböztetjük a szuperkereteket és ablakozott CSMA/CA-t használó Beacon és az ablakozás nélküli CSMA/CA-t használó Non-Beacon üzemmódokat. A Beacon nélküli üzemmód előnye, hogy a slave alvó állapotban tartható külső interrupt érkezéséig abban az esetben, ha a master hatótávolságon belül van és folyamatosan üzemel.

A CSMA/CA növeli a sikeres átvitel valószínűségét, de garantálni nem tudja az ütközés elkerülést. Ezért minden egyes sikeresen átvitt csomag megérkezését jelzi a vevő egy ACK csomaggal.



12.ábra: Az IEEE 802.15.4 keretformátuma

A fenti ábrán látható az IEEE 802.15.4 szabványnak megfelelő keretformátum [5,7]. A keret elején találjuk a szinkronizációs fejlécet (Synchronisation Header), amely 5 byte hosszú. Ebből 4 bájt a szinkronizációt segítő információt nem hordozó bitsorozat (Preamble Sequence), és további egy bájt a keret elválasztó (Start of Frame Delimiter=SFD). A szabvány szerint az SFD értéke 0xA7 és az ezt megelőző 4 bájt 0x00. Ezen keretalkotók hossza változtatható, de akkor kiesünk a szabvány hatásköréből.



13.ábra: Szinkronizációs fejléc

Az SFD után következő 1 bájt a keret további hosszát határozza meg (Frame Length). Ezen bájt legmagasabb helyiértékén 0-nak kell szerepelnie, a fennmaradó 7 biten határozhatjuk meg a hosszt. Így ennek maximuma 127. Ez a mező küldés és vétel során is releváns. A keret további része a MAC protokoll adat egység(MAC Protocol Data Unit=MPDU) elnevezést kapta. A hálózati réteg tovább bontja az MPDU-t 5 alkotóra. Ezek közül sorrenben az első a keret kontrol mező (Frame Control Field=FCF), amely 2 bájt hosszú. Tartalmának jelentése az alábbi ábrán látható:

Bits: 0-2	3	4	5	6	7-9	10-11	12-13	14-15
Frame Type	Security Enabled	Frame Pending	Acknowledge request	Intra PAN	Reserved	Destination addressing mode	Reserved	Source addressing mode

14.ábra: Keret kontrol mező

A következő egy bájt az úgynevezett Data Sequence Number (DSN) mezőnek van fenntartva, majd ezt követi a cím, amely 0 és 20 bájt hosszú lehet. A cím után foglal helyet a keretben a hasznos adat (MAC payload), végül a keretet egy ellenőrző mező (Frame Check Sequence=FCS) zárja.

4. MAC protokollok

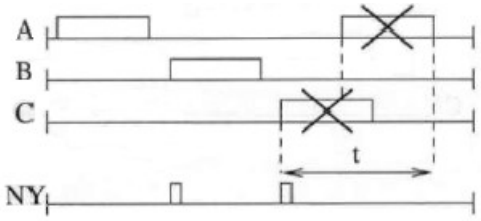
A hálózati összeköttetések létrejöhetnek két pont közötti, vagy üzenetszórásos csatornán. Utóbbi esetben az állomások között verseny folyik a csatorna használati jogáért. Lényeges kérdés, mi alapján osztjuk ki a jogot, ezt a feladatot látja el a MAC [1,6] alréteg az adatkapcsolati szinten.

Alapvetően három féle közeg hozzáférési mód létezik. Az átvitelvezérlés lehet véletlen, osztott illetve központosított. Véletlen vezérlés esetében bármely állomás elkezdhet adni, ha szabad a csatorna (pl.: CSMA/CD). Osztott átvitelvezérlés alkalmazásakor egy állomás adhat (pl.: token busz), központosított esetben egy szerver osztja ki a közeget. Más nézőpontból vizsgálva statikus és dinamikus csatornakiosztást különböztetünk meg. A statikus megoldásra példa, amikor diszkrét időintervallumokat foglalunk forgó prioritással. Minden kommunikációs résztvevőnek megvan a saját időszelete, amit kivárva hozzáfér a csatornához. Ha a megfelelő időszelet alatt nem küld az adott eszköz, akkor a csatorna kihasználatlanul marad. A dinamikus megoldás kérés alapján működik. A statikus kiosztás hagyományos módszere az FDM (Frequency Division Multiplexing), azaz frekvenciaosztásos multiplexelés. Ha N darab állomás van a hálózaton, akkor a sávszélességet N részre osztjuk fel, és minden felhasználó kap egy frekvenciasávot, elkerülve így az interferenciát. Jó hatásfokú mechanizmusnak bizonyul ez a módszer az olyan hálózatokban, ahol viszonylag kevés és rögzített számú felhasználó van, illetve ezek nagy terhelést eredményező forgalmat bonyolítanak. Olyan hálózatban, ahol több állomás van, esetleg ezek száma változik vagy a forgalom változó nem az FDM a legjobb megoldás. A problémát az jelenti, hogy a nem aktív állomások sávszélessége elvész, mivel mások nem férnek hozzá. Dinamikus csatornakiosztás eseténél bizonyos előfeltételezésekkel élünk. Feltételezzük, hogy az állomások függetlenek, a kommunikációt egyetlen csatornán bonyolíthatjuk le. Továbba ha egyszerre kerül adásra két keret, akkor azok ütköznek, feltételezzük, hogy az állomások képesek megállapítani, hogy a csatorna szabad-e. Az ilyen protokollokat csatornafigyelő (Carrier Sense) protokolloknak nevezzük, gyakorlati példa az 1-perzisztens CSMA, a nemperzisztens CSMA és a p perzisztens CSMA, melyek a következő alfejezetben részletesebben megismerhetők.

4.1 ALOHA

A master-slave hierarchia szerinti állomások közötti kommunikációt segíti. Eredetileg rádiós rendszerre szánták ezt a protokollt. A master gépet a többi géppel két csatorna köti össze, az egyik vonalon a nyugtaküldés történik, a másik az üzemi csatorna. A helyes átvitelt a master egy

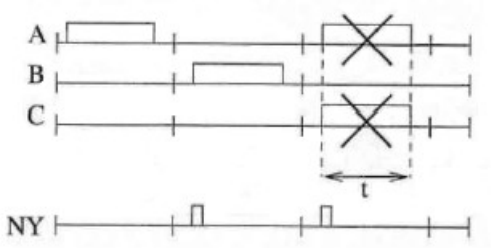
nyugtával jelzi a slave-nek. Egy slave bármikor adhat az üzemi vonalon, azonban előfordulhat, hogy egyidőben egy másik slave is ad. Ekkor ütközés történik, mindkét keret elveszik (15. ábra). A nyugtázás küldésekor nem léphet fel ütközés, mivel ezt a csatornát csak a master használhatja. Amennyiben nem érkezik nyugtázás (acknowledge) a mastertől, akkor a slave ütközést feltételez, és újraadja a keretet.



15.ábra: Csomagok ütközése Aloha használatával

A keretek hossza előírtan azonos.

Az ALOHA protokollját kiegészíthető időrések használatával, ezesetben réselt ALOHA-nak nevezzük a protokollt. A versengő állomások az időrés elején kezdhetnek adni, amit egy a master által kibocsájtott szinkron jel jelez. Egy időrés valamivel hosszabb, mint az egy keret leadásához szükséges idő. Így ha ütközés történik, akkor azzal egy időrést veszítünk (16. ábra), szemben az ALOHA-val, ahol szerencsétlen esetben közel két keretnyi időt is veszíthetünk. Így a csatorna maximális kihasználtsága nőhet.

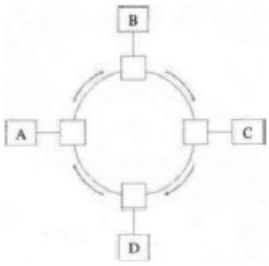


16.ábra: Csomagok ütközése réselt Aloha használatával

4.2 Token ring

Magyarul vezérjeles gyűrűként ismerhetjük. Az elnevezésben szereplő token szó jelentése vezérjel. Ez egy speciális keretet jelent, funkcióját tekintve feljogosít az adatközlésre. Működése egyszerű, az adhat, akinél a token van. A vezérjelet természetesen egy megszabott idő után tovább kell adni. Előnye, hogy így nincsenek ütközések, jó kihasználtság érhető el. Gyűrű topológiát használ (17.

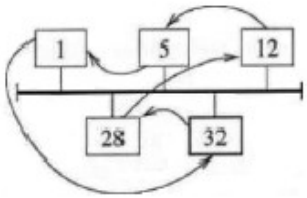
ábra), szabványát az IEEE 802.5 rögzíti. Mára már elavultnak számít ez a megoldás.



17.ábra: Token ring

4.3Token busz

A vezérjeles sín vagy busz is a múlté. Az állomások a vezérjel továbbítását egy logikai gyűrű szerint oldják meg. Ez azt jelenti, hogy mindegyik állomás előre tudja, hogy kitől kaphatja meg a vezérjelet, és kinek továbbíthatja azt. Adatot minden állomás küldhet bármely másik állomásnak. Az adatforgalom itt is ütközésmentes. Busz topológiát használ (18. ábra), az IEEE 802.4-es szabványban rögzítették a leírását.



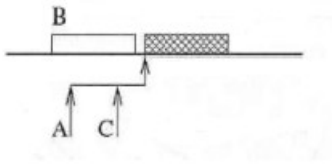
18.ábra: Vezérjeles busz

4.4 CSMA változatok

A CSMA(Carrier Sense Multiple Access) vivő érzékeléses többszörös hozzáférést jelent. Olyan rendszereknél alkalmazzák, ahol biztonsággal megoldható, hogy az állomások addig ne kezdjenek el adni, amíg a csatorna foglalt.Ezt úgy valósítják meg, hogy az állomások figyelik az átviteli közeget, képesek ”belehallgatni” a csatornába. Több változata is kialakult.

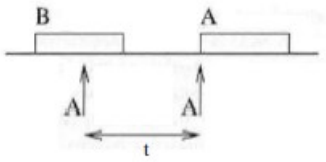
Egy változat az 1-perzisztens CSMA. Tegyük fel, hogy „B” állomás éppen használja a csatornát. Eközben az „A” állomás is használni szeretné az átviteli közeget. Ilyen esetben „A”addig figyeli a csatornát, amíg az fel nem szabadul. Amint ez megtörténik, rögtön adni kezdi a saját kereteit. Ennek az egyszerű eljárásnak a hátránya, hogy ha olyan helyzet áll elő amelyben a foglalt

csatornára egyszerre több állomás is vár, akkor amint szabad az adatátvitel, mindegyik állomás elkezdi adni a kereteit. Így ezek ütközni fognak (19. ábra), és az adatok elvesznek.



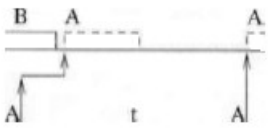
19.ábra: 1-perzisztens CSMA

A nemperzisztens CSMA eljárás során amikor „A" belehallgat a csatronába, és az foglalt, akkor vár egy előre definiált t ideig. Mikor ez eltelik, újra megvizsgálja, hogy felszabadult-e az erőforrás. Ha szabad, akkor használatba veszi, ha nem, újabb t ideig vár és újra próbálkozik ennek lejárta után. Ezzel a megoldással nagy valószínűséggel elkerülik egymást a várakozó állomások, azaz egy keret befejeződésekor valószínűleg nem fognak többen egyszerre adni.



20.ábra: Nemperzisztens CSMA

További változat a p-perzisztens CSMA. Ennél a megoldásnál egy állomás a csatorna foglaltsága esetén vár, majd amint az felszabadul p valószínűséggel adni kezd illetve 1-p valószínűséggel tovább vár t ideig, majd újrapróbálkozik, ahol p $0<p<1$.



21.ábra: P-perzisztens CSMA

4.5 CSMA/CD

Ütközésérzékeléssel kiegészített vivőérzékeléses többszörös hozzáférést jelent. A CSMA/CD (Carrier Sense Multiple Access with Collosion Detection) azt jelenti, hogy ha egy állomás nem érzékel vivőt és adni kezd, attól még tovább folytatja a csatorna figyelését. Ha egy másik állomás adását detektálja, akkor megszakítja a saját adását. Ennek a protokollnak egy továbbfejlesztett változatát használják az Ethernet hálózatok.

4.6 CSMA / CA

A CSMA/ CA protokollt az IEEE 802.11-es szabvány tette népszerűvé. A vezetékes hálózatoknál használt CSMA/CD eljárás az ütközéseket detektálja, szükség esetén újra küldésre kerül sor. Vezeték nélküli hálózatokban az ütközéseket el kell kerülni, mert nem mindig detektálhatóak az ütközések. Előfordulhat két csomag ütközése során, hogy az adók kioltják egymás jelét, a vevő nem érzékel semmit. Amennyiben az ütközések nem mindig detektálhatóak, akkor meg kell előzni, hogy ütközés jöhessen létre. Ezt foglalja magába a CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) eljárás, amely vivő érzékeléses többszörös hozzáférést jelent ütközés elkerüléssel.

A folyamat lépései a következők:

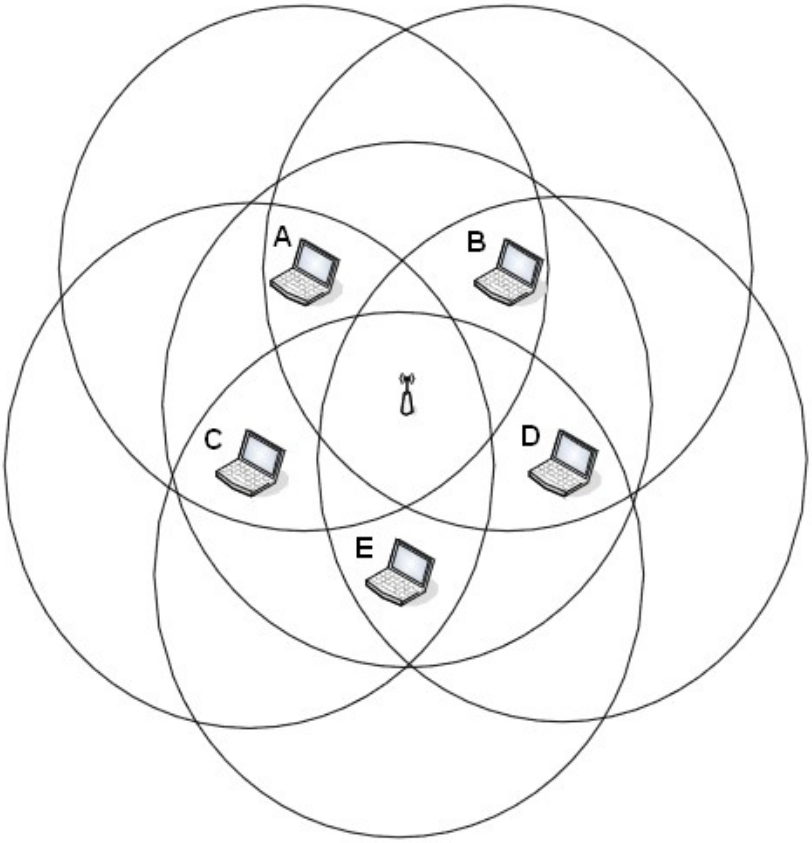
- A küldést kezdeményező állomás figyeli, hogy a csatorna szabad-e.
- Amikor szabad, nem kezd el rögtön adni. Véletlen hosszú, úgynevezett IFS (Inter Frame Space) ideig vár, és ezután újra belehallgat a csatornába. Így csökkenthető annak az esélye, hogy ütközés jöjjön létre.
- A csatorna újbóli vizsgálata után -amennyiben az még mindig szabad- elkezdi az adást.
- Az adatok elküldése után, nyugtázásra vár a vevőtől.
- Ha a nyugtázás nem érkezik meg, az adatokat újra küldi az egyes lépéstől.

Az ütközés elkerülés az IFS idő kivárása mellet egy RTS (Ready to Send) / CTS (Clear to Send) cserével is elérhető. Az ilyen eljárás során az adó az adatok átvitele előtt küld egy RTS jelet a vevőnek, jelezvén küldési szándékát. A vevő egy CTS csomaggal jelez vissza, ha készen áll az

adatok fogadására.

A CSMA/CA protokoll DCF (Distributed Coordination Function) és PCF (Point Coordination Function) üzemmódban működhet. Központi vezérlés oldja meg az ütközések elkerülését PCF üzemmódban, míg az állomások mellérendelt viszonyban vannak egymáshoz képest és maguk felelnek azért, hogy csomagot akkor küldjenek, amikor szabad a csatorna.

A DCF üzemmód esetén a folyamat részleteit tekintsük egy konkrét elrendezés esetén. Tételezzünk fel egy olyan elrendezést, ahol az A állomás hatósugarában van a B es és a C, a B állomás hatósugarában van az A és a D, a C állomás hatósugarában van A és E , a D állomás hatósugarában van B és E, az E állomás hatósugarában van a C és D. Az elrendezést szemlélteti az alábbi ábra:



22.ábra: Rádiós állomások lefedése

Elsőként az A állomás belehallgat a csatornába mielőtt küldeni szeretne. Ha a csatorna foglalt, akkor nem kezd küldésbe, hanem véletlen ideig vár, majd újra megvizsgálja, hogy szabad-e a csatorna. Amennyiben a csatorna szabad, az állomás egy RTS (Request to Send) keretet küld a B állomásnak, azaz jelzi, hogy küldeni szeretne. A B állomás, azaz a célállomás feldolgozza az RTS

keretet, és ha nem küld neki senki adatot akkor egy CTS (Clear To Send) kerettel válaszol, azaz jelzi a kezdeményezőnek, hogy készen áll adatok fogadására. Ezekkel a keretekkel az olyan helyzeteket védjük ki, amikor a D állomás ad a B-nek, de ezt az A nem veszi észre. Ez a probléma rejtett terminál néven is ismert. Ilyen esetben az A állomás üresnek érezné a csatornát, és elkezdene küldeni B-nek, ami ütközéshez vezetne, a D és az A állomás adatai elvesznének. Amikor az A jelű adó megkapja a CTS keretet, elkezdi küldeni az adatcsomagot a B jelű célállomásnak. A küldés megkezdésekor az adó elindít egy ACK időzítőt, a B állomás a küldés hibátlan befejezésekor visszaküld egy ACK keretet. Ezzel egy adatcsomag küldése történt meg. Ha az ACK időzítő lejárta előbb kapta a visszaküldött ACK keretet az A állomás, akkor a teljes állomány elküldéséhez az újabb RTS keretet küld a B-nek a következő adatcsomag elküldésére. Ha azonban az időzítő lejárt, akkor újraküldi az adott keretet. Tegyük fel, hogy az A állomás hatósugarában van még a C jelű állomás is. Ilyenkor a C is megkapja az RTS keretet, amit az A szánt B-nek. A C állomás képes előrelátni, hogy az A jelű adatot fog küldeni, kiszámítja, hogy az meddig fog tartani és egy belső NAV állapot foglaltra állításával jegyzi meg magának, hogy az A állomás foglalt.

Amikor a B állomás visszaküldi a CTS keretet A-nak a fentihez hasonló módon azt D is megkapja. A D állomás a keretből kiszámítja, hogy meddig fog tartani a kommunikáció, és erre az időre a B állomáshoz tartozó belső NAV állapotot foglaltra állítja. Amíg a NAV foglalt, addig nem kezdeményez küldést az adott állomásnál. A NAV jeleket az állomások tárolják, nem kerülnek elküldésre.

Az E állomás kívük esik az A és a B állomás hatósugarán, így nem kapja meg az RTS és CTS kereteket. AZ E állomás ugyan a C és D hatósugarában van, de mivel a NAV jelek nem kerülnek elküldésre, üresnek érzi a csatornát. Ha E üzenetet akar küldeni C-nek vagy D-nek, akkor a protokoll szabályai szerint teheti meg ezt.

5. A kommunikációs rendszer komponensei

A BME-MIT Tanszék DSP laborjában összeállított rendszert CC2420-as modulok, a hozzá tartozó Develepmont Kit, illetve egy ADSP-BF537 EZ-KIT LITE fejlesztő kártya alkotja. A Blackfin DSP programozása Visual DSP-ben történik, a CC2420 Development Kitben található Chipcon RF IC vezérlését pedig a Smart RF Studio-val végezzük.

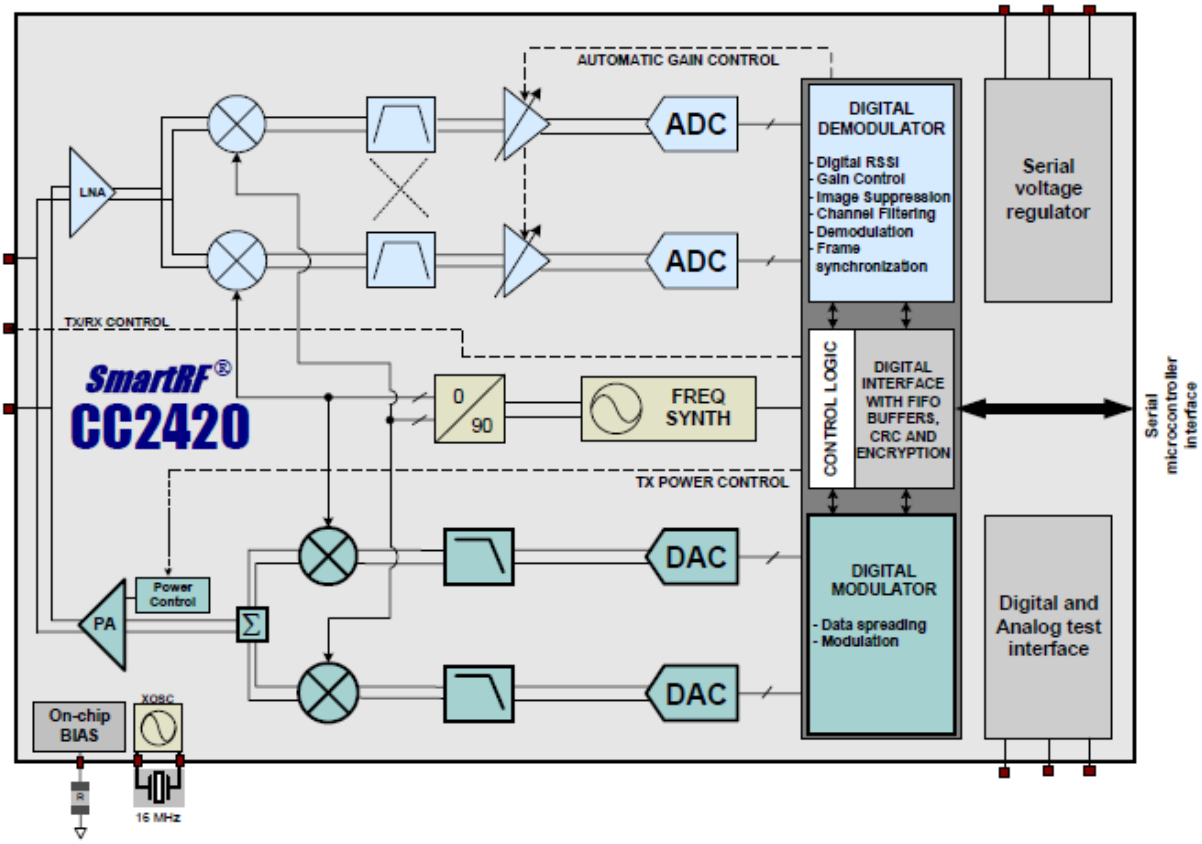
5.1 A CC2420-as IC

A Chipcon terméke egy 2.4 Ghz-es chip, amely az IEEE 802.15.4 szabványt támogatja. Alacsony fogyasztású és alacsony feszültségű alkalmazásokhoz fejleszették ki. Az ISM sávban működő CC2420-as adóevevő az említett szabvány alsó rétegeit, a fizika illetve a hálózati réteg egy részét valósítja meg [7].

Legfőbb jellemzői:

- 2400-2483.5 Mhz-es rádiófrekvenciás vevő,
- DSSS (Direct Sequence Spread Spectrum) modem 2 Mchip/s és 250 kbps effektív adat-sebesség,
- O-QPSK moduláció,
- alacsony fogyasztás (vétel: 18.8 mA, adás:17.4 mA),
- magas érzékenység (- 95 dBm),
- alacsony táp igény (2.1 – 3.6 V),
- beépített regulátor,
- programozható kimenő teljesítmény,
- elkülönített adó- és vevő FIFO-k,
- kevés külső komponens csatlakoztatása szükséges,
- SPI interfész,
- 802.15.4 MAC harveres támogatása,
- Hardveres szinkronizációs előtag (CRC),
- CCA, digitális RSSI (Received Signal Strength Indicator) érték,

- automatikus biztonsági funkciók a adó-és vevő FIFO-kban,
- kis méretű(7x7 mm), QLP tokozás.



23.ábra: A CC2420 blokkvázlata

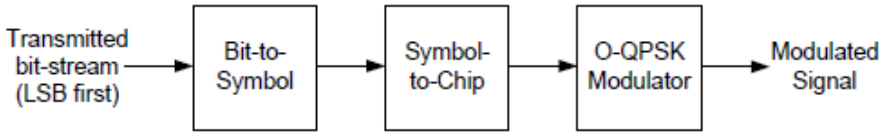
A CC2420-as a vett rádió frekevenciás jelet erősíti (LNA – Low Noise Amplifier), majd középfrekvenciára konvertálja át (IF – Intermediate Frequency). Konvertálás után a 2 Ghz-es jelet szűri és erősíti, majd az ADC az analóg jelet digitálissá konvertálja. A digitalizált jelen végzi el a speciális átalakításokat, mint például a byte szinkronizáció, vagy a végső csatorna szűrés. Az ADC dinamikus tartományban működik, az AGC (Automatic Gain Control) hurok biztosítja, hogy ne legyen túlvezérlés, illetve a kis jelek megfelelő erősítését.

Vétel során az SFD (Start of Frame Delimiter) láb magas értéke jelzi egy keret érkezését. Az SFD hosszát a CC2420-as IC esetében a SYNCWORD regiszterben állíthatjuk, míg az úgynevezett Preamble Sequence hosszát az MDMCTRL0.PREAMBLE_LENGTH regiszter segítségével állíthatjuk át. Ezutóbbi hossza csak küldéskor releváns. Az eszköz a vett adatot egy 128 byte-os vételi FIFOba buffereli. A felhasználó SPI interfészen keretsztül ki tudja olvasni az adatokat. Az ellenőrző szekvenciát (CRC) a CC2420 hardveresen állítja elő. Továbbá a CCA (Clear Channel

Assesment) jel is elérhető az IC egy lábán vételi üzemmódban.

Adás során az adatok szintén egy 128 byte-os FIFO-ból kerülnek elküldésre. A két FIFO nem azonos, a vételi FIFO-hoz hasonlóan ki tudjuk olvasni SPI-on keresztül a kimenő FIFO tartalmát is. A szabványnak megfelelően a keretek szinkronizációs előtaggal és SFD-vel kezdődnek, ezt is hardveresen állítja elő az eszköz. Szintén az IEEE 802.15.4 szabványnak megfelelően állítja elő a digitális modulátor a jelet, majd átadja a DAC-nek. Egy analóg aluláteresztő szűrő adja tovább a jelet felkonvetálásra. Innen már rádió frekvenciás a jel, melyet a PA (Power Amplifier) erősítő vesz át, majd az antennához érkezik.

A modulációs formátumot az IEEE 802.15.4-es szabvány állítja fel. DSSS (Direct Sequence Spread Spectrum) esetén chipek megfelelő sorozatával kódoljuk az egyest és a nullát. A kódolás során minden byte két szimbólumra válik szét, ezek 4 bitesek egyenként. Az átvitel az LSB-vel (Least Significant Byte) kezdődik. Minden szimbólumot egy chip-szekvencia képvisel, ezek egyenként 32 chip-esek (25.ábra).



24.ábra: DSSS moduláció

A 4 bites szimbólumok chip-szekvenciáit az alábbi táblázat szerint értelmezzük:

Symbol	Chip sequence (C_0, C_1, C_2, ... , C_{31})
0	11011001110000110101001000101110
1	11101101100111000011010100100010
2	00101110110110011100001101010010
3	00100010111011011001110000110101
4	01010010000101110110110011100011
5	00110101001000101110110110011100
6	11000011010100100010111011011001
7	10011100001101010010001011101101
8	10001100100101100000011101111011
9	10111000110010010110000001110111
10	01111011100011001001011000000111
11	01110111101110001100100101100000
12	00000111011110111000110010010110
13	01100000011101111011100011001001
14	10010110000000111011110111000100
15	11001001011000000111011110111000

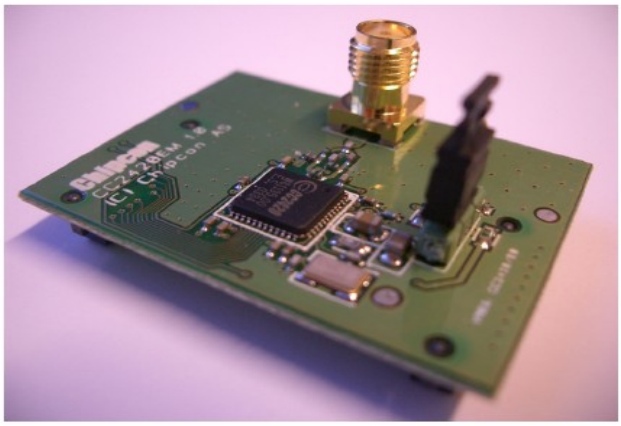
25.ábra: Chip-szekvenciák értelmezése

Az IEEE 802.15.4 szabványt hardveresen támogatja a CC2420 chip. A szabvány felállít egy címformátumot, melyet képes szűrni a CC2420, így olyan kereteket nem dolgoz fel, amelyek nem az IEEE 802.15.4 szabványnak megfelelőek (address recognition).

A BME-MIT DSP laborjában rendelkezésre áll egy CC2420DK (Development Kit) jelölésű fejlesztő csomag [8], melynek szolgáltatásai megkönnyítik és felgyorsítják a munka menetét. A csomag tartalmaz 2 darab CC2420EB (Evaluation Board) fejlesztőkártyát és 2 darab CC2420EM (Evaluation Modul) fejlesztőmodult. A modulon van beültetve maga az IC és a megfelelően illesztett külső komponensek.

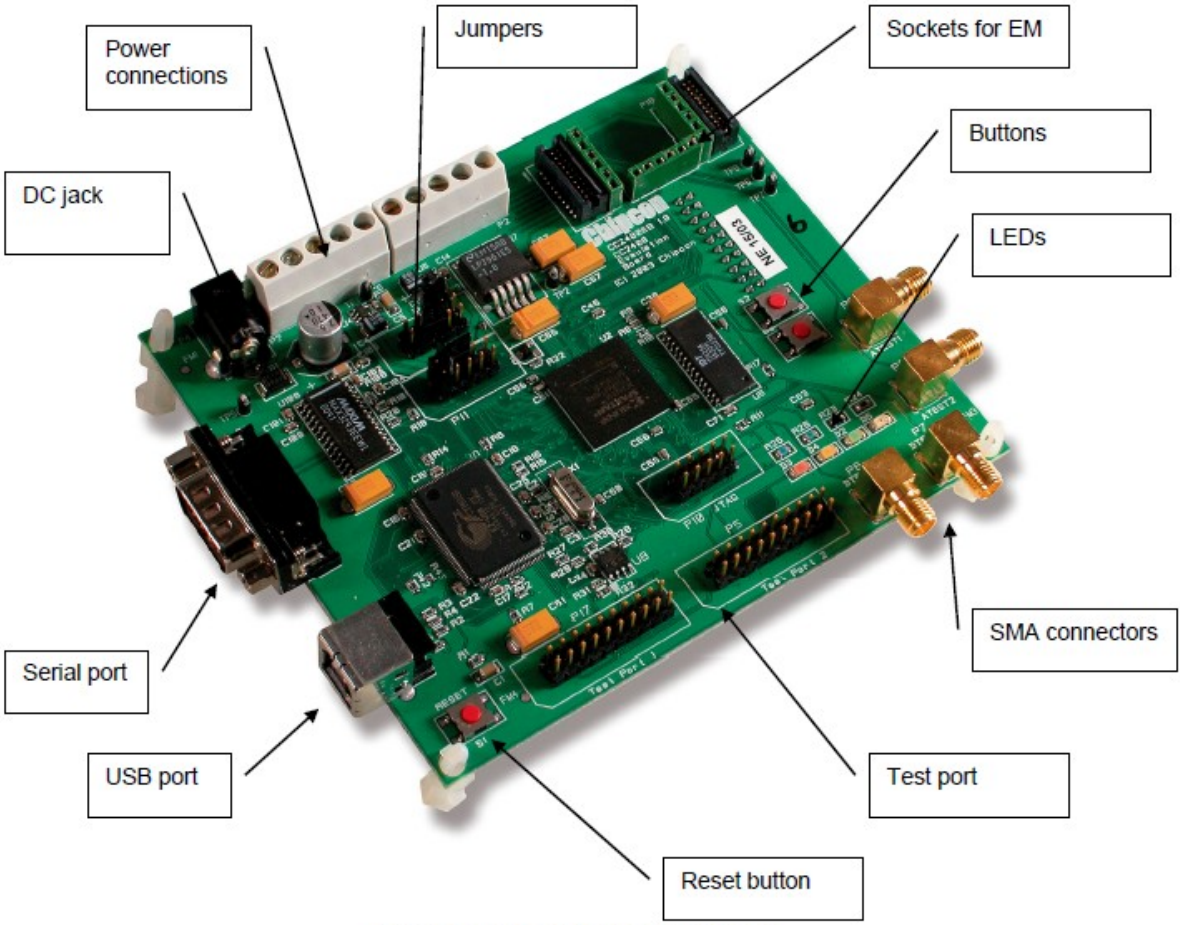
A fejlesztő modul felhasználható más mikroprocesszorhoz való csatlakozás létesítéséhez is. Magas frekvencián üzemelő IC-k érzékenyek a NYÁK (Nyomtatott huzalozású áramkör) fizikai elrendezésére (layout). A Chipcon a CC2420EM elrendezését optimalizálta, így saját NYÁK gyártásához is ajánlja megtartani azt.

A NYÁK 4 rétegű , 1 mm vastagságú. A legfelső réteg megjelölése a 1-es, lefelé haladva a 2-es és 3-as réteg belső réteg. Az alsó réteg a 4-es, amely összeköttetéseket valósít meg, csakúgy mint az 1-es réteg. A tápellátásért a 3-as réteg felelős.



26.ábra: CC2420 EM

A CC2420EB alaplapként funkionál a CC2420 számára. Tápellátást, USB portot, nyomógombokat, ledeket, konfiguráló jumpereket és csatlakozókat biztosít a fejlesztő számára. A kártya NYÁK-ja szintén 4 rétegű. A PC-hez USB interfészen keresztül kapcsolódik, melynek segítségével beállításokat is elvégezhetünk a CC2420-on, illetve adatküldést is vezérelhetünk.



27.ábra: CC2420 EB

A felsorolt hardvert kezelő szoftver a SMART RF Studio, melynek installálása során a Windows-os USB driver szintén települ.

A DSP-hez kapcsolódó kártya a ProDSP Kft. tervei alapján készült. A kártya segítségével a Chipcon adóvevő csatlakoztatható a DSP kártyához.

28.ábra: CC2420 rádiós kártya

A SPI kommunikációhoz szükséges lábak (SPI_CS, SPI_SCLK, SPI_SI, SPI_SO, TX, RX, FIFO, FIFOP, SFD, RESET, SPI_EN, CCA) csatlakozó sorra vannak kivezetve, és csatlakoznak a DSP kártya UART illetve SPI csatlakozójához, ahol a megfelelő multiplexálási beállítások elvégeztével elérhetőek a kívánt lábak a DSP F és J portján (48. ábra). A vezérlők SPI portjairól és a használt jelek funkcióiról a hatadik fejezet számol be.

5.2 Az ADSP-BF537

A DSP [15] a Digital Signal Processing vagy Digital Signal Processor betűszó rövidítése, ami magyarul digitális jelfeldolgozást illetve digitális jelfeldolgozó processzort jelent. A DSP processzorok megtalálhatóak a mobiltelefonoktól, MP3 lejátszótól kezdve az orvosi diagnosztikai eszközökön át a PC-ig szinte mindenütt. A digitális jelfeldolgozás céljaira készített processzorok az 1980-as évek elején jelentek meg a piacon. Ebben az időben a mikroprocesszorok a nagyobb sebességű jelek feldolgozása és az ezekhez kapcsolódó folyamatok irányítása terén való alkalmazhatóságának egyik korlátja a technológiailag korlátos működési/műveletvégzési sebesség volt. Az elvégzendő - sokszor elég speciális műveletsorokat követelő - feladatok aritmetikai alapműveletekből való összeállítása tovább osztotta a műveletvégzési sebességet. Ezeknek a területeknek a meghódítására indult el a jelfeldolgozó processzorok fejlesztése, melyek meghatározó tulajdonságai:

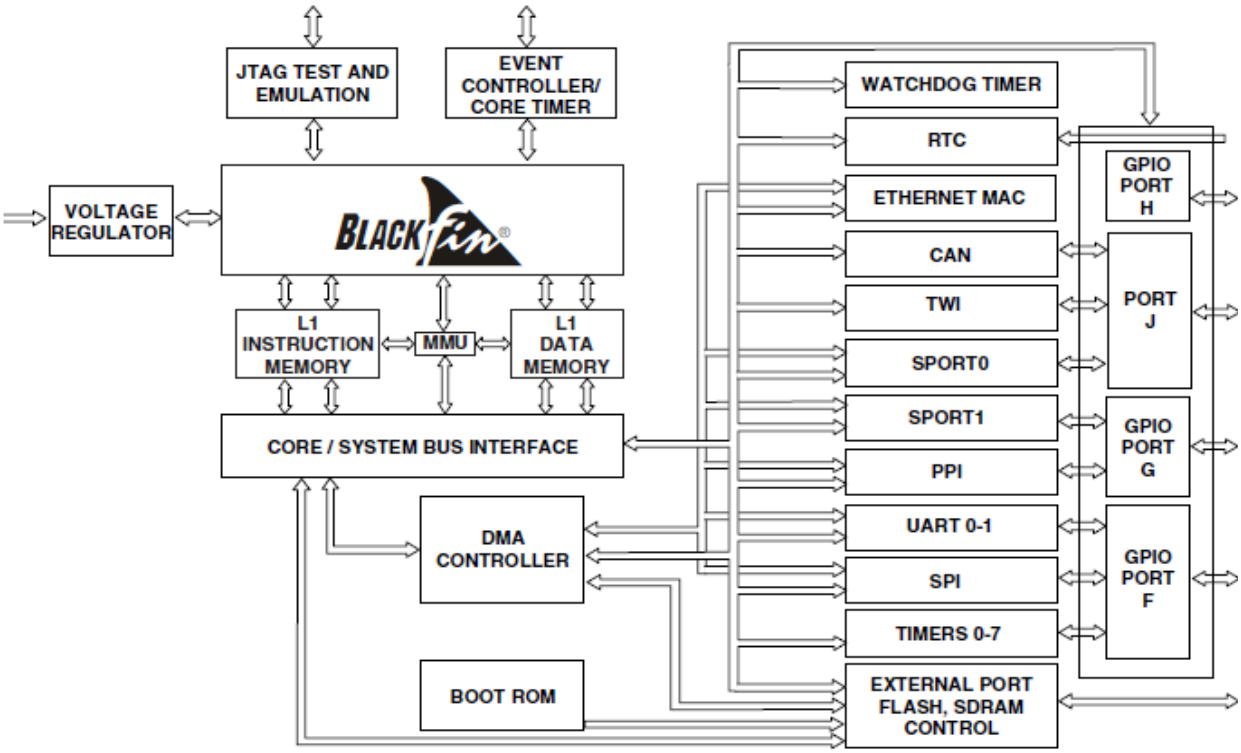
- nagy műveletvégzési sebesség,
- speciális – de tipikus – műveletsorok összefogása egyetlen műveletbe,
- speciális címzési módok ezen műveletek támogatására,
- integrált nagysebességű elemek,
- RISC architektúra.

A DSP és a hagyományos mikroprocesszorok több tekintetben is hasonlítanak egymásra. Az eltérés az utasításkészletben van, mivel a jelfeldolgozó processzorok rendelkeznek speciális műveletekkel, melyek főleg diszkrét jeleken végezhető aritmetikai műveletek. Egyes esetekben több műveletvégző egységet is tartalmazhatnak a DSP-k, melyek párhuzamos működésével teljesítménynövelést érhetünk el.

A digitális jelfeldolgozásra való képesség megkövetelte az architektúra és az utasításkészlet olymódon való kialakítását, hogy az lehetővé tegye minimum a három alapművelet (osztás hiányzik), a logikai és eltolási műveletek (barrel shifter) valóban nagysebességű hardverrel történő végrehajtását. A sebesség növelése érdekében kezdettől fogva megpróbálták átlapolni az utasítások elvégzését (pipeline műveletvégzés), ez azonban a technológia akkori állása szerint igen speciális utasításstruktúrát eredményezett. A belső buszrendszernél Harvard architektúrát alkalmaztak a memória hozzáférések gyorsítása érdekében. Az adat- és a programmemória szétválasztásával ezek

a területek is párhuzamosan hozzáférhetők. Így egy órajelciklus alatt kiolvashatunk két operandust, ezeket összeszorozhatjuk, illetve még ugyanebben a ciklusban hozzáadhatjuk egy harmadik számhoz is.

A jelfeldolgozó processzorok tervezői kezdettől fogva sok tekintetben hasonló célokat tűztek ki maguk elé, mint a RISC architektúra szakemberei. A két architektúra igen sok hasonló tulajdonságot mutat. A jelfeldolgozó processzorok is „átestek” a legfontosabb memória és perifériaelemek integrálásán. Az mikrokontrollerekkel való architekturális kialakítás hasonlósága révén a közelmúltban a Texas Instruments (mint az egyik legnagyobb DSP gyártó) elkezdte használni a DSC (Digital Signal Controller) megjelölést jelprocesszoraira. A határvonal manapság már egyre kevésbé éles.



29.ábra: Az ADSP-BF537 blokkvázlata

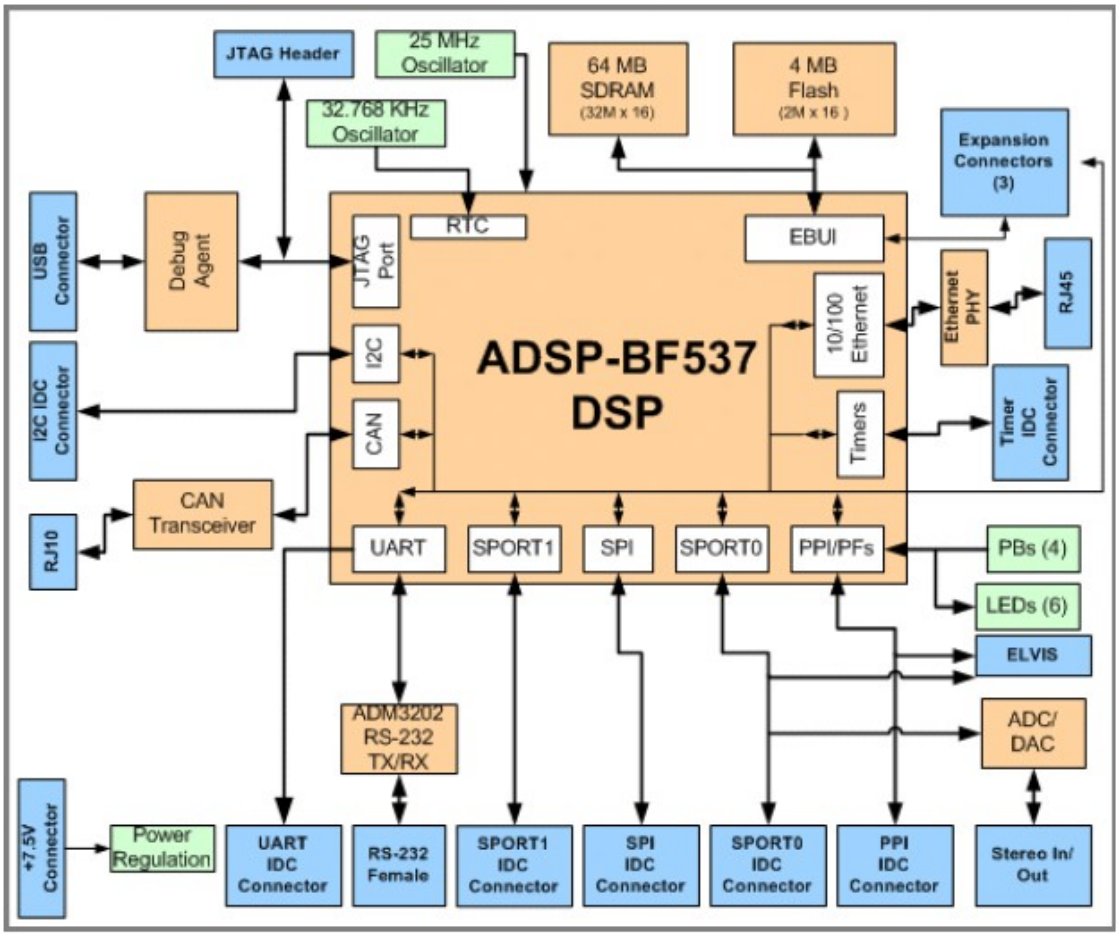
A Blackfin család az Analog Devices 16 és 32 bites mikroprocesszor családja. A szakdolgozat feladatom szerint ehhez a jelfeldolgozó processzorhoz kell illesztenem a CC2420 rádiós adóvevőt.

Az ADSP-BF537 [10] jellemzői összefoglalva:

- 600 Mhz órajel.
- A processzor 48 Kbyte belső programmemóriát és 64 Kbyte adatmemóriát tartalmaz.
- Az utasítások 16 és 32 bitesek lehetnek.
- Az adatok 8, 16 vagy 32 bitesek lehetnek.
- Az aritmetikai műveleteket 3 egység végzi: az aritmetikai, logikai egység, azaz ALU (Arithmetical Logical Unit), a szorzó-összeadó egység, vagy másnéven MAC (Multiply-Accumulate) , valamint egy léptető (shift) regiszter .
- Címzés:a memória direkt és indirekt címzése is támogatott. Az indirekt címzés számára a processzorok külön regisztereket biztosítanak. Lehetséges a címek inkrementálása és dekrementálása is. Indirekt címzésre a P0…5 regiszterek használatosak. Cirkuláris buffer is létrehozható, megvalósításukhoz a B0…3, I0…3, L0…3, M0…3 regiszterek használatosak. A B regiszter tartalmazza a buffer kezdőcímét (Base), az L regiszter a hosszát (Length), az M regiszter a léptetés értékét (Modify), míg az I regiszter az aktuális memóriacímet tárolja (Index).

A BME-MIT DSP laborban ez a processzor EZ-KIT LITE fejlesztőkártya [11] formában használható, ami USB porton csatlakozik a PC-hez. Az algoritmusok fejlesztése során megkönnyítik munkánkat a fejlesztőkártyák, illetve a szoftveres fejlesztőkörnyezet.

A kártyán a processzor egy 182 lábas BGA tokozásban található meg. Külső memóriát is illesztettek a DSP-hez, méghozzá 64 Mbyte SDRAM és 4 Mbyte Flash formájában. Az analóg jelek digitalizálására az AD1871 48 kHz mintavételi frekvenciájú átalakító található a kártyán, míg a D/A átalakításhoz az AD1854 átalakító. Egy kimeneti és egy bemeneti sztereó jack csatlakozóval is fel van szerelve a kártya. Ethernet, CAN és ELVIS interfész, valamint univerzális aszinkron adóvevőv (UART) is van a fejlesztő board-on. A kártyán találhatóak programozható led-ek és nyomógombok is.

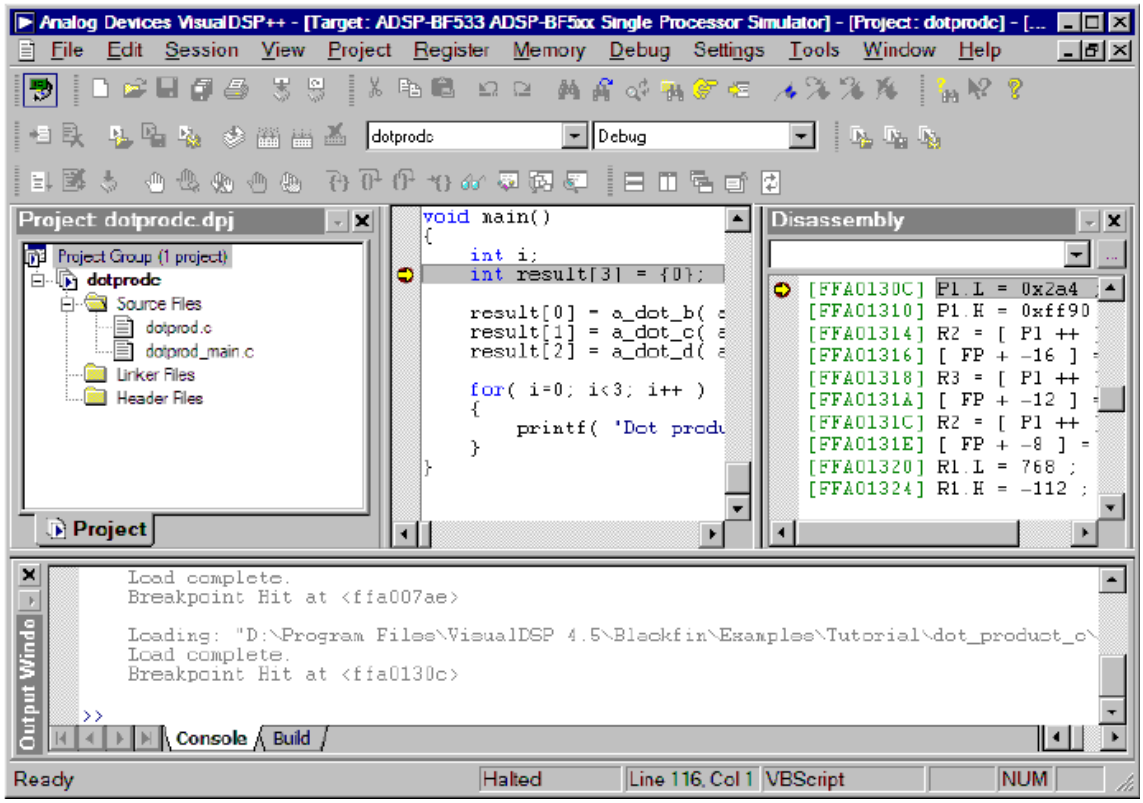


30.ábra: Az ADSP-BF537 EZ KIT LITE kártya felépítése és blokkvázlata

5.3 Smart RF Studio és a Visual DSP fejlesztőkörnyezet

A Blackfin DSP programjainak fejlesztése és feltöltése a Visual DSP integrált fejlesztői környezet [12] segítségével történt. Ez egy Windows alkalmazás, amelyet két független modul alkot. Ezek a projekt szerkesztő, illetve a debugger.

A Visual DSP fejlesztő környezet előnye, hogy hatékonyan képes együttműködni a DSP-hez készült kártyákkal. A projekt editor segítségével tudunk forrásfájlokat a projekthez adni illetve szerkeszteni azokat.



31.ábra: Visual DSP minta nézet

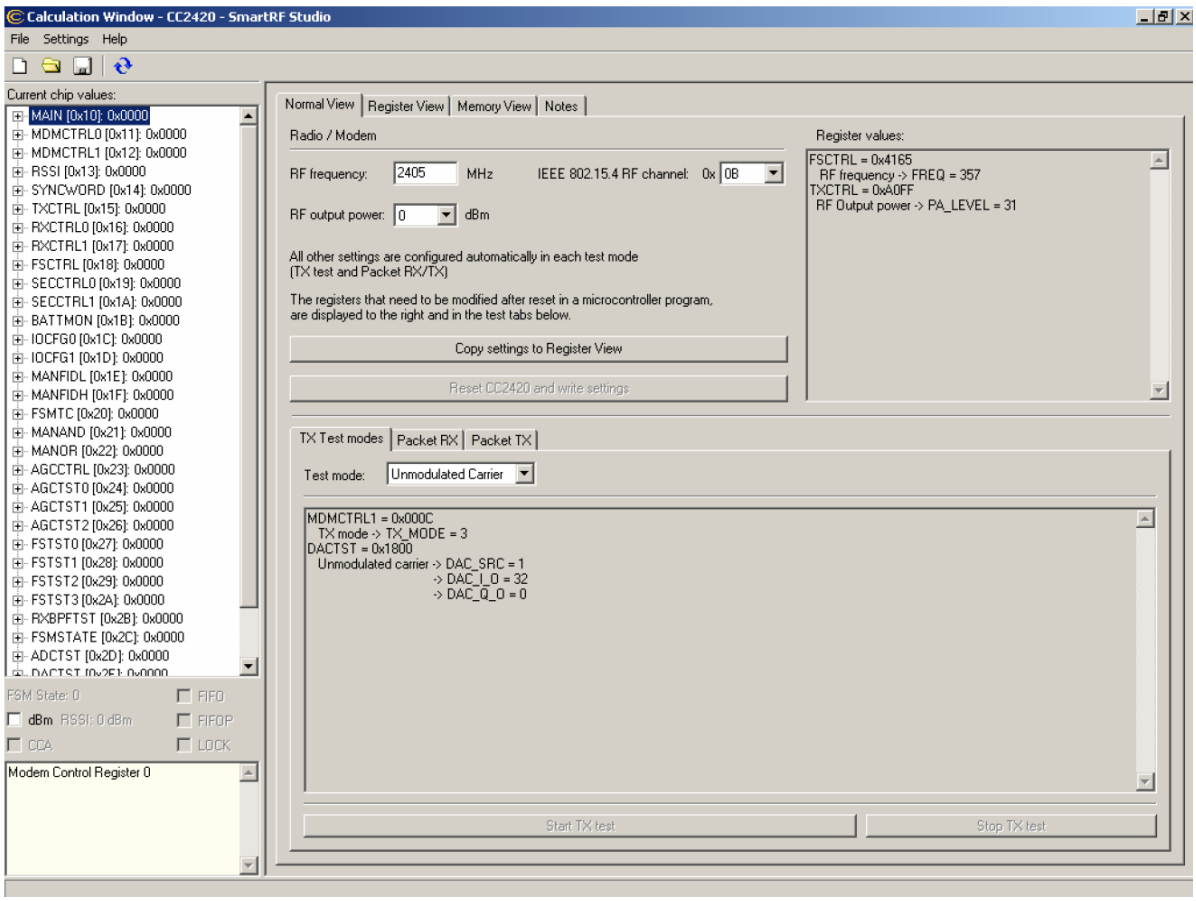
A fejlesztés Assembly, C, C++ nyelven is történhet, továbbá különböző nyelvű szoftver modulokat keverhetőek is. A programunk működését a debuggerben nyomon követhetjük. A processzor regisztereinek, illetve memóriaterületeinek tartalmát figyelemmel tudjuk követni a fejlesztőkörnyezetben. Szoftveres környezetben szimulátor, hardveres környezetben pedig JTAG segítségével.

Egyszerűbb jelfeldolgozó szoftverekre jellemző felépítés a következő lépésekből tevődik össze:

- Inicializálás: a processzor perifériáinak beállítása, adott interruptok engedélyezése.
- Egyszeres műveletek: azon lépéseket, melyeket elég egyszer elvégeznünk a végtelen ciklus(lásd köv.) előtt kell megvalósítani, mivel így időt spórolunk meg.
- Végtelen ciklus: tulajdonképpen ez egy várakozó állapot. A processzor idle állapotban van, akkor történik adatfeldolgozás, amikor valamelyik periféria kéri ezt egy megszakítással.
- Megszakítás rutin: az interrupt rutin hajtja végre a beérkező adatok feldolgozását, majd

visszatérünk a várakozó álláspontra.

A Chipcon a CC2420 felhasználói számára biztosítja a Smart RF Studio vezérlő szoftvert [9]. Rádiós funkciók megvalósítására és fejlesztésre használható. A program a Chipcon internetes oldaláról ingyenesen letölthető. A 32. ábra mutatja a Smart RF Studio felhasználói interfészét. A program indításakor felugró ablak a device manager. Ez a modul felelős a megfelelően csatlakoztatott Chipcon eszközök felismeréséért. Minden másodpercben frissíti az aktív eszközök listáját. Az aktív eszközök listáján a kívánt egységre kattintva jelenik meg az úgynevezett Calculation window, amely segítségével már konfigurálhatjuk az adóvevőnket. Egyszerre több ilyen ablakot is megnyithatunk természetesen, amennyiben több eszközzel kívánunk foglalkozni. Két jól elkülöníthető részből áll az ablak. A bal oldali sáv az úgynevezett Register Status sáv, míg az ablak nagyobb felületét képviselő rész az úgynevezett System Configuration modul. A regiszter sávban a gyakran használt regiszterek értékeit követhetjük nyomon. A regiszter neve nagy betűkkel szerepel, majd mögötte szögletes zárójelekben szerepel a regiszer címe, ezt követi kettőspont után az aktuális érték, ahogyan ezt a 33.ábra is mutatja:



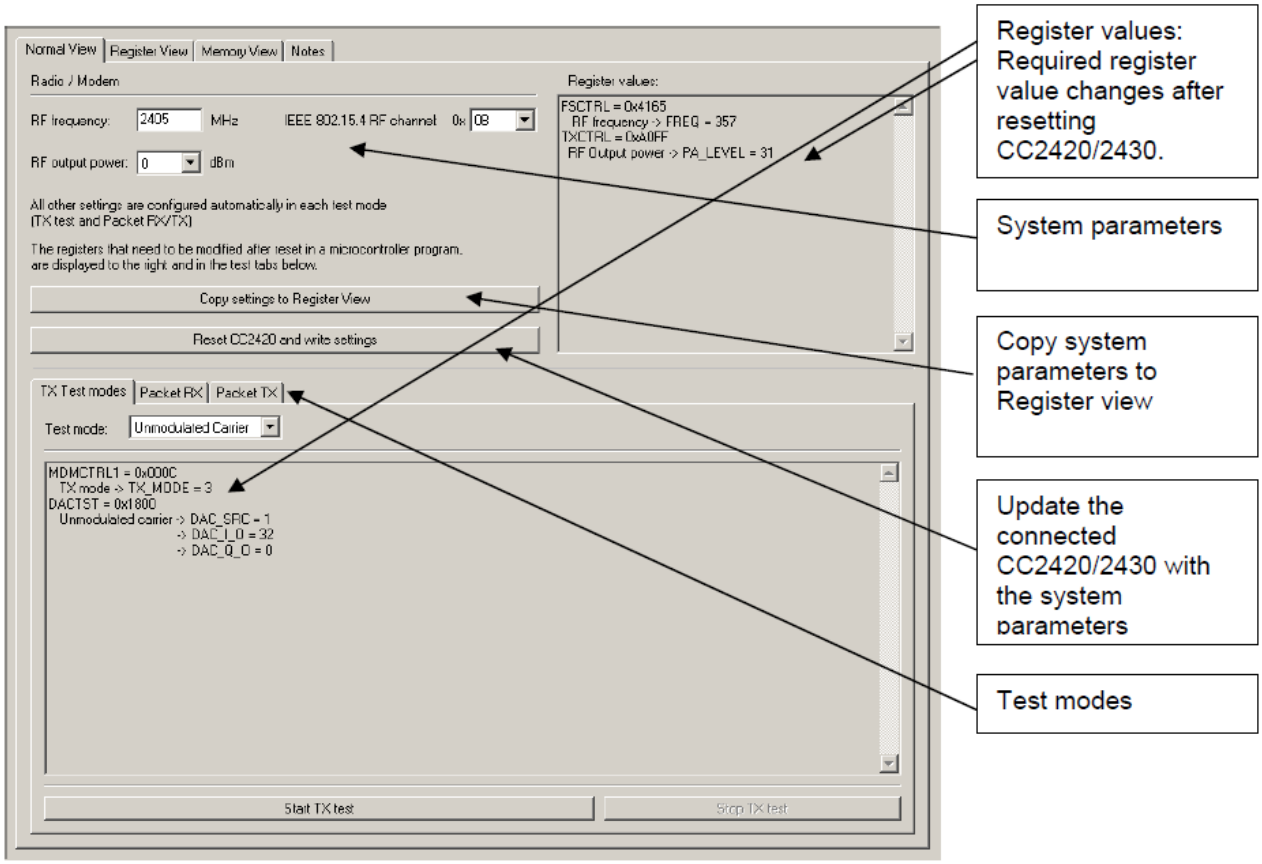
32.ábra: Smart RF Studio

A rendszer konfiguráció nézetben három fül közül választhatunk. Normál nézet, Regiszter nézet és Memória nézete érhető el, illetve további egy fül áll rendelkezésre jegyzetelés céljából.

A rendszer konfiguráció normál nézetében tudjuk a beállításainkat a CC2420-on realizálni, a rendszer paramétereket és bizonyos regiszter értékeket figyelemmel kísérni, illetve a teszt módok közül választani. A rendszer paraméterek változásakor a regiszter értékek automatikusan frissülnek. A CC2420 lehetséges üzemmódjai:

- TX test Mode: folyamatos adás üzemmód, főként a rádiós paraméterek vizsgálatára használatos
- Packet RX : bufferelt fogadás üzemmód
- Packet TX : bufferelt küldés

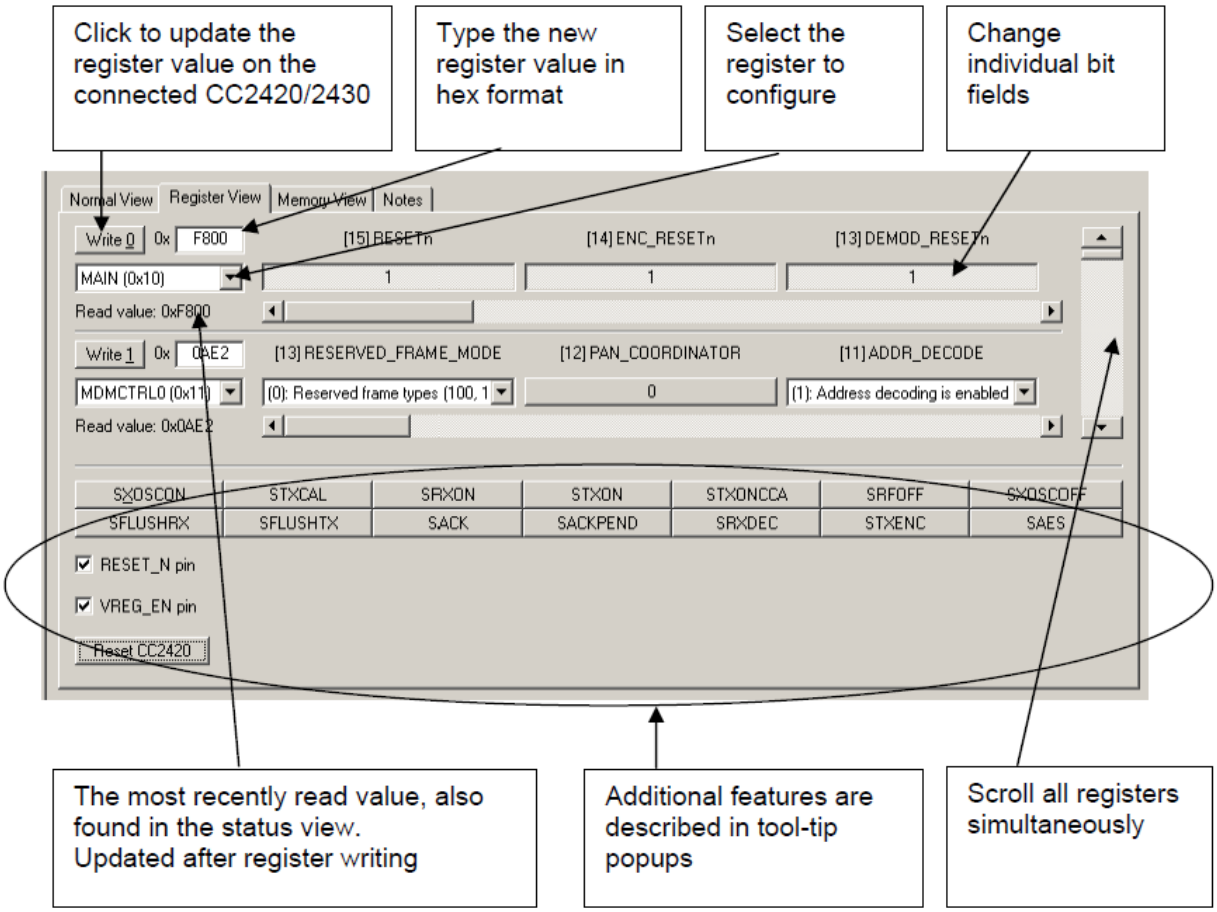
A normál nézet az alábbi ábrán látható:



33.ábra: Smart RF Studio – Normál nézet

A rendszer konfiguráció regiszter nézetében manuális beállításokat végezhetünk el. Az ajánlott

eljárás, hogy a normál nézetben beállított értékeinket vezessük át erre a fülre is a normál nézetben található ”Copy settings to Regiszter View” gombbal. Ezzel nagyban csökkentjük a hibák lehetőségét. A regiszter nézetet az alábbi ábra mutatja:



34.ábra: Regiszter nézet

Egyszerűen kiválasztjuk a listából a kívánt regisztert, és updateljük az értékét.

A rendszer konfigurációs memória nézetében hozzáférünk a CC2420 RAM-hoz. Ez a memória tartalmazza a TxFIFO-t és az RxFIFO-t valamint az úgynevezett biztonsági bankokat. Az egyszerű csomagküldés ebből a nézetből is elindítható MDMCTRL0 és MDMCSTRL1 megfelelő beállításával majd a SXOSCON gomb megnyomásával két CC2420 eszközön. A vevő berendezés ablakában az SRXON gomb megnyomásával állítjuk vételi üzemmódba, a küldő fél oldalán a TxFIFO-ba manuálisan írunk egy sztringet, majd az STXON gomb megnyomásával küldő üzemmódba kapcsol. Ennek eredményeként a sztring a vevő RxFIFO-jába kerül. Sztringek mellet lehetőség van hexadecimális információ átvitelére is, továbbá nem IEEE 802.15.4 szabvány szerinti

csomagok küldésére.

Ezekkel az eszközökkel már lehetséges egy IEEE 802.15.4 szabványnak megfelelő keret küldése is.

Tételezzük fel, hogy a következő jellemzőkkel bíró keretet akarjuk elküldeni:

- Keret típus: adat,
- Biztonság: nem alkalmazott,
- Keret felfüggesztés: nem,
- Nyugta(ACK) kérés: igen,
- Intra pan: igen,
- Cél címzése: 16 bites rövid cím,
- Forrás címzése: 16 bites rövid cím,
- Sequence number: 0x01,
- Adat: 0x01,0x23,0x45,0x67,0x89.

Ahhoz, hogy egy ilyen keret küldését megvalósítsuk az TxFIFO-jába be kell írnunk manuálisan a következőket:

10 61 88 01 20 24 34 12 FE CA 01 23 45 67 89.

Az első bájt (0x10) a keret hosszát adja meg, ehhez tudnunk kell, hogy a CC2420 2 bájtnyi FCS -t illeszt a keret végéhez, valamint, hogy a hosszba nem számít bele az első bájt. Így képződik jelen esetben tehát a 16 bájt hosszú keret. A következő két bájton (0x6188) található kontroll mező segítségével állíthatjuk be a keret típusát, biztonsági funkciót, keret felfüggesztést, nyugtázás igénylését is. A fenti esetben adat keretet küldünk ki, amit a második bájt legkisebb három helyiértékén 001 bitsorozattal jelzünk az adóvevőnek, a további funkciókat a megfelelő bitek 0-ba illetve 1-be állításával kapcsolhatjuk ki vagy be (14.ábra). Szintén a kontroll mezőben kell megadnunk a címzett és a forrás címzési módját. A fenti példában a 16 bites rövid címet az 10 bitsorozat jelzi az FCF 10-11(cél) és 14-15-ös (forrás) biten. Az FCF két bájtja "LSB first" sorrendben kerül elküldésre. A következő bájt a DSN, jelen esetben 0x01, majd a PAN ID, melynek

a fenti példában 0x2420 az értéke (LSB first). Ezt követi a már megjelölt címzési módon a cél-és forráscímek megadása (LSB first), valamint az átvinni kívánt adat maximálisan 127 bájton.

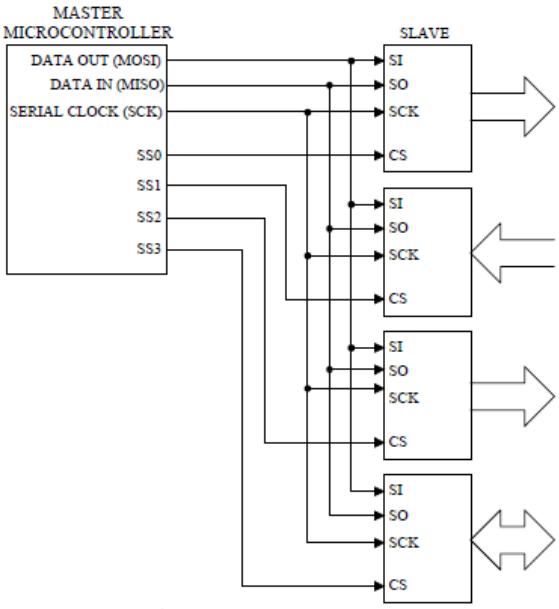
A küldés Smart RF Studioval többféle módon is lehetséges. Ha a normál nézetet választjuk, akkor első lépésként meg kell győződnünk arról, hogy az adó és a vevő is ugyanazt a csatornát használja. Válasszuk a PacketTX fület, ahol az ”IEEE 802.15.4 compilant” jelölőnégyzetet bepipálva biztosítjuk a szabványnak megfelelő keretformátumot. Végül a ”Start Packet TX” gomb megnyomásával indíthatjuk az adást. A vevő oldalon kiolvashatjuk az RxFIFO-ból a megérkezett csomagokat.

6. A vezérlők kommunikációja SPI interfészen

6.1 Soros kommunikációs egységek

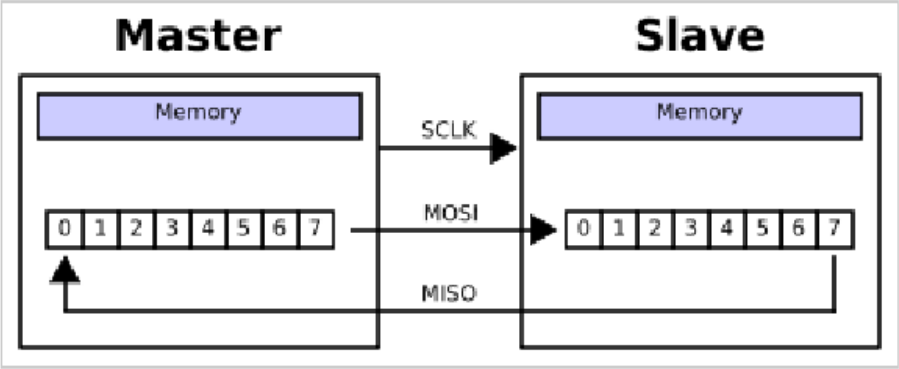
6.1.1 SPI

Az SPI egy szinkron, soros periféria adatátviteli interfész, amely nagy sebességű (1-25 MBit/s), full duplex adatátvitel lehetőségét nyújtja. A rendszer egy master egységből és egy vagy több slave egységből áll. A master vezérlő biztosítja az órajelet a többi résztvevőnek. Az interfész elemei 3 + (1...n) vezetéken keresztül kapcsolódnak egymáshoz, ahol n a slave egységek száma, amelyet a kiválasztó vonalak száma korlátoz. Az egységek közötti összeköttetések 3 vonalon busz rendszerűek, az n darab kiválasztó vonal pont-pont kapcsolatot létesít a master és egy-egy slave egység között. A master az egyik vezetéken biztosítja az órajelet (Serial Clock=SCLK), egy másik vezetéken választ kommunikációs partnert (Chip Select=CS/ Slave Select=/SS), míg az adat két vezetéken kerül továbbításra. Ezek az adatátvitel irányától függően kerülenek kiválasztásra. Ha a master az adó, akkor a MOSI (Master Out Slave In) vonalat használja, amikor pedig a vevő szerepét tölti be, akkor a MISO(Master In Slave Out)-t. A slave egységek közvetlenül a kiválasztó jelek azonosítják, az átvitelben nincs sebességet csökkentő címzési fázis. A busz topológiának megfelelően csak a kiválasztott slave egység kimenete engedélyeződik. Egy tipikus SPI kapcsolatot mutat a 35. ábra.



35.ábra: Tipikus SPI kapcsolat felépítése

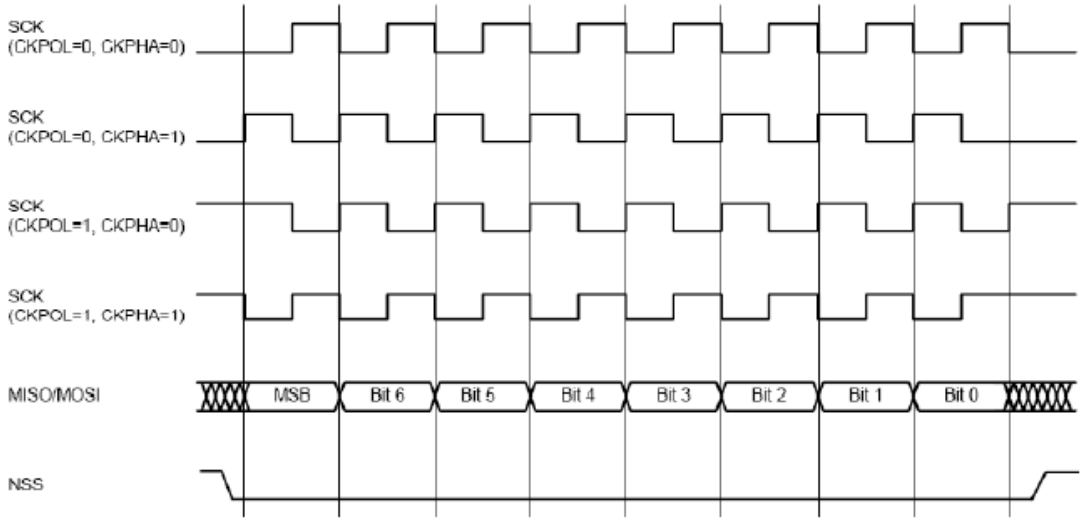
Az adatbiteket a két egység ki- és bementi shift regiszterében egyszerre léptetjük, közös órajel hatására. Az órajel generálása az /SS kiválasztójel aktiválásakor indul és a 8 adatbit ki-beléptetése után leáll (lásd 36. ábra).



36.ábra: SPI adatkapcsolat modellje

Az adatok kiléptetése és a beérkezett adatok mintavételezése az órajel egymással ellentétes éleire történik. Ennek megfelelően az órajel két lehetséges fázishelyzete és kétféle lehetséges kiindulási állapotának megfelelően négyféle SPI átviteli ciklus (úgynevezett SPI üzemmód) definiálható (lásd 37. ábra).

SPI-mód	CKPOL	CKPHA	SCK nyugalmi helyzete	SCK felfutó élére	SCK lefutó élére
0	0	0	alacsony (0)	Mintavétel	Adatbit váltása
1	0	1	alacsony (0)	Adatbit váltása	Mintavétel
2	1	0	magas (1)	Adatbit váltása	Mintavétel
3	1	1	magas (1)	Mintavétel	Adatbit váltása



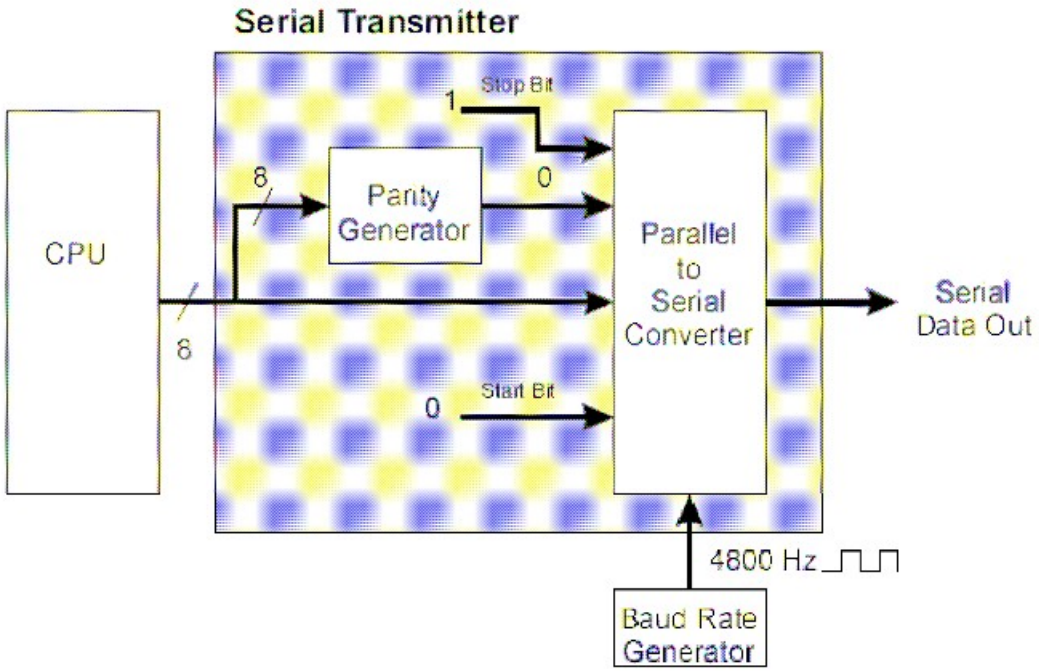
37.ábra: Az SPI üzemmódok értelemezése

A szakdolgozat során használt rendszerben a beágyazott rendszer elemei az SPI interfészt használják adatcserére, a DSP-n egy monitorozó alkalmazás is fut, melynek segítségével figyelemmel kísérhetjük az adatforgalmat. Ez az alkalmazás az UART-on, azaz a soros adóvevő egységen kommunikál a PC-vel.

6.1.2 UART

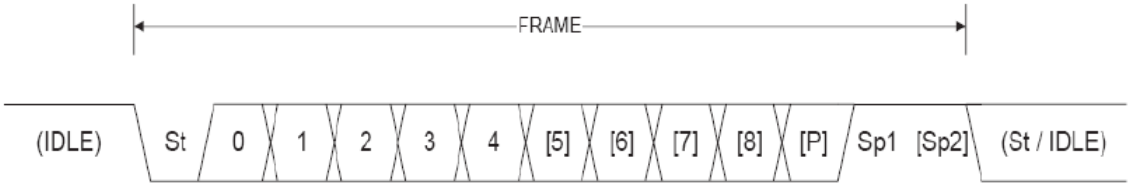
Az egymáshoz fizikailag közel elhelyezkedő eszközök között nagyobb sebességű adatforgalom zajlik, itt a közös órajellel történő pontos időzítés fontos követelmény. Ilyen esetekben alkalmazunk szinkron átvitelt. A fizikailag máshol található perifériák között a távolság egyes esetekben már nagyobb lehet. Az ilyen egységek közötti vezetékek csak kisebb adatátviteli sebességeket engednek meg. A 0 - néhány 100 kbit/s adatsebesség számára még megfelelő lesz a közös órajel meglétét nem igénylő aszinkron soros adatátviteli technika is. Az aszinkron soros átvitel pont-pont típusú, tehát egyszerre két eszköz közötti kommunikációt tudunk megvalósítani. Amennyiben az átvitel egy adattovábbító csatornán alapul, azon csak egyirányú (szimplex), vagy váltakozó irányú (fél duplex) kapcsolatot tudunk kialakítani. A fizikai jeltovábbításhoz szabványos jelszintekre van szükség. Ez utóbbiakat és a jelek további tulajdonságait írja elő asszimetrikus jelvezetés esetén az RS232C (Revised Standard 232) szabvány.

A soros adóvevő egység segítségével minden információs bit azonos ideig jelenik meg egymás után a jelvezetéken. Az átvitel sebességét ez a bitidő szabja meg, ezt nevezzük bitsebességnek (bit rate). Az elterjedten használt baud (=bit/sec) összefüggés ez esetben megegyezik a bitsebességgel. Az átvitel egy bitidőnyi mindenképpen 0 szintű startbit átvitelével kezdődik. Ezt követik a léptető regiszterrel sorosított adatbitek a legkisebb helyiértéktől és egy esetleges – nem kötelező – paritásbit hibaellenőrzés céljából. A paritásbit az adatbitekben található egyesek számát párosra (páros paritás) vagy páratlanra (páratlan paritás) egészíti ki. Az adatbitek száma megválasztható 5-től 8-ig.Az adategység átvitelét egy mindenképpen 1 szintű stopbit jelzi, és az adatvonal szintje ezen az értéken is marad a következő átviteli adatkeret kezdetéig. Így a startbit 0 szintje mindenképpen jelszint-váltást jelent.



38.ábra: Univerzális aszinkron adóvevő (UART)

Az adatátvitel tehát itt is keretekben történik. Mivel a kapcsolat aszinkron, ezért nem igényel külön órajel vezetéket az eszközök között. Az egyes egységek időzítése saját lokális rendszeridőzítésükön alapul. Ebből kifolyólag a berendezések közötti adatátvitelben használható sebességértékek egy véges listából választhatók, valamint ezen sebességértékek forrásának frekvenciahibája nem lehet nagyobb kb 1-2 %-nál. Az adatkeretek általános felépítése tehát a 39. ábrán látható:

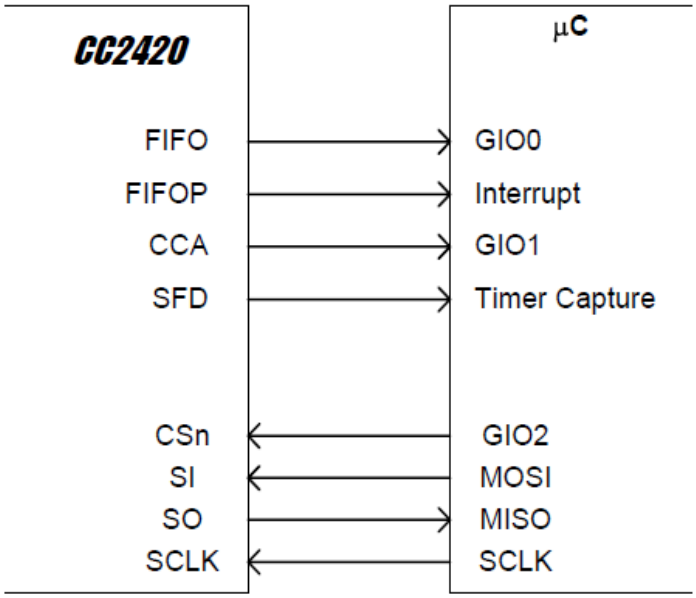


39.ábra: Soros aszinkron adatkeret általános felépítése

6.2 A CC2420 SPI portja

A CC2420 egy 4 vezetékes SPI-kompatibilis interfészen konfigurálható, ahol a CC2420 a slave egység [7]. Minden cím és adatátvitel a magasabb helyiértékétű bittől kezdődően kerül átvitelre. Az interfész digitális bemenetei (SCLK, SI, CSn) nagy impedanciás bemenetek, melyekhez külső felhúzó ellenállások illesztése szükséges, amennyiben nincsenek meghajtva. A nem használt I/O

lábak kimentre állíthatók fix "0" szintek, így elkerülhetjük a szivárgó áramokat. A kimenő adatokat az SO lábon érhetjük el. A CC2420 egy tipikus illesztését a 40. ábra szemlélteti.



40.ábra: A CC2420 konfigurációs és adatátviteli interfésze

A 40. ábrán látható lábak funkciói:

- FIFO: A FIFO láb magas értéke jelzi, hogy legalább egy bájt található az RXFIFO-ban. A vételi FIFO-ba egy új keret érkezésekor elsőként a keret hossz mező íródik be. Így ezen láb magas értéke jelzi egy beérkező keret hosszának sikeres vételét, újra alacsony szintre váltása pedig a vételi FIFO teljes kiürülését. Küldés üzemmódban nincs funkciója.
- FIFOP: Magas értéke jelzi, hogy az RXFIFO-ból kiolvasatlan bájtok száma meghaladja az előzőleg beállított küszöbértéket (FIFOP_THRESHOLD). Ha a CC2420 címfelismerő funkcióját bekapcsoljuk, akkor a FIFOP láb addig nem veszi fel a magas értéket, amíg a bejövő keret át nem megy a címfelismerésen. Szintén magas értéket vesz fel amikor egy új keret utolsó bájtja is megérkezett.
- CCA (Clear Channel Assessment): Az IEEE 802.15.4 szabvány hardveres támogatásának részeként a CC2420 a CCA lábon jelzi az átviteli csatorna foglaltságát.
- SFD (Start of Frame Delimiter): Vételi üzemmódban az SFD lábon magas érték jelzi a keret elválasztó mező hiánytalan megérkezését. Amennyiben a keret utolsó bájtja is megérkezett

az SFD láb újra alacsony értéket vesz fel. Amennyiben a keret nem felel meg a címfelismerő funkciónak (a CC2420 megfelelő konfigurálás szükséges a funkció bekapcsolásához), az SFD láb azonnali alacsony értékével jelzi ezt. Küldés üzemmódban is jelzi egy keret SFD mezőjének sikeres küldését, szintén magas értékkel. Újra alacsony szintre vált, ha a teljes keretet sikeresen elküldte a CC2420.

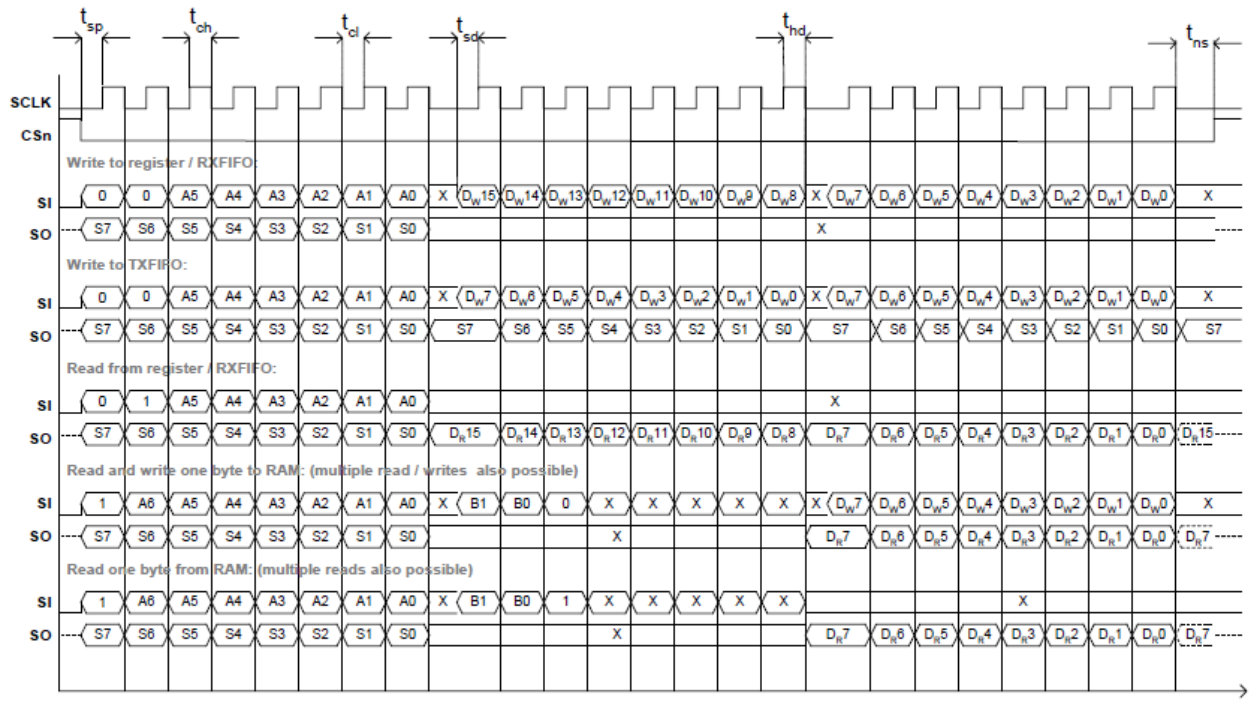
A CC2420 regiszterei között 33 darab 16 bites konfigurációs- illetve státuszregisztert találunk. Emellett további 15 parancs engedélyező (command strobe) és kettő 8 bites regiszter áll rendelkezésre, amelyek segítségével hozzáférünk a különálló adó- és vevő FIFO egységekhez. A command strobe-okat például az oszcillátor engedélyezésére, vételi vagy küldési módba való kapcsolásra használjuk. A belső memória összesen 368 bájtos, és az SPI interfészen keresztül érhető el. Három memória bankra osztották a memória területet: TXFIFO (bank0), RXFIFO (bank1) és biztonsági rész (bank2). A TXFIFO és az RXFIFO egyenként 128, a biztonsági memória 112 bájtos. A konfigurációs regiszterek segítségével állíthatjuk be a kívánt működési feltételeket. Leírásukat a 41. táblázat tartalmazza.

Address	Register	Register type	Description
0x0E	SAES	S	AES Stand alone encryption strobe. SPI_SEC_MODE is not required to be 0, but the encryption module must be idle. If not, the strobe is ignored.
0x0F	-	-	Not used
0x10	MAIN	R/W	Main Control Register
0x11	MDMCTRL0	R/W	Modem Control Register 0
0x12	MDMCTRL1	R/W	Modem Control Register 1
0x13	RSSI	R/W	RSSI and CCA Status and Control register
0x14	SYNCWORD	R/W	Synchronisation word control register
0x15	TXCTRL	R/W	Transmit Control Register
0x16	RXCTRL0	R/W	Receive Control Register 0
0x17	RXCTRL1	R/W	Receive Control Register 1
0x18	FSCTRL	R/W	Frequency Synthesizer Control and Status Register
0x19	SECCTRL0	R/W	Security Control Register 0
0x1A	SECCTRL1	R/W	Security Control Register 1
0x1B	BATTMON	R/W	Battery Monitor Control and Status Register
0x1C	IOCFG0	R/W	Input / Output Control Register 0
0x1D	IOCFG1	R/W	Input / Output Control Register 1
0x1E	MANFIDL	R/W	Manufacturer ID, Low 16 bits
0x1F	MANFIDH	R/W	Manufacturer ID, High 16 bits
0x20	FSMTC	R/W	Finite State Machine Time Constants
0x21	MANAND	R/W	Manual signal AND override register
0x22	MANOR	R/W	Manual signal OR override register
0x23	AGCCTRL	R/W	AGC Control Register
0x24	AGCTST0	R/W	AGC Test Register 0
0x25	AGCTST1	R/W	AGC Test Register 1
0x26	AGCTST2	R/W	AGC Test Register 2
0x27	FSTST0	R/W	Frequency Synthesizer Test Register 0
0x28	FSTST1	R/W	Frequency Synthesizer Test Register 1
0x29	FSTST2	R/W	Frequency Synthesizer Test Register 2
0x2A	FSTST3	R/W	Frequency Synthesizer Test Register 3
0x2B	RXBPFTST	R/W	Receiver Bandpass Filter Test Register
0x2C	FSMSTATE	R	Finite State Machine State Status Register
0x2D	ADCTST	R/W	ADC Test Register
0x2E	DACTST	R/W	DAC Test Register
0x2F	TOPTST	R/W	Top Level Test Register
0x30	RESERVED	R/W	Reserved for future use control / status register
0x31-0x3D	-	-	Not used
0x3E	TXFIFO	W	Transmit FIFO Byte Register
0x3F	RXFIFO	R/W	Receiver FIFO Byte Register

R/W - Read/write (control/status), R - Read only, W – Write only, S – Command Strobe (perform action upon access)

41.táblázat: A CC2420 konfigurációs regiszterei

Minden regiszter írási vagy olvasási ciklusban 24 bit kerül elküldésre az SPI vonalon. A kiválasztó jelnek (CSn) az átvitel során alacsony szinten kell lennie. Az első küldött bit a RAM vagy regiszter választó bit, ahol "0" érték jelenti a regiszter hozzáférést. A következő bit az írás/olvasás (R/W) bit, ahol "0" érték jelenti az írást. A fentieket követi a 6 címbit, ahol a küldés során a magasabb helyiértéktől haladva érkeznek az adatok. A 16 adatbit ezek után következhet, szintén a fent említett módon. A MISO lábon az első 8 bitidőre a státusz bájt jelenik meg. Az időzítést a 42. ábra szemlélteti.



42.ábra: SPI időzítések

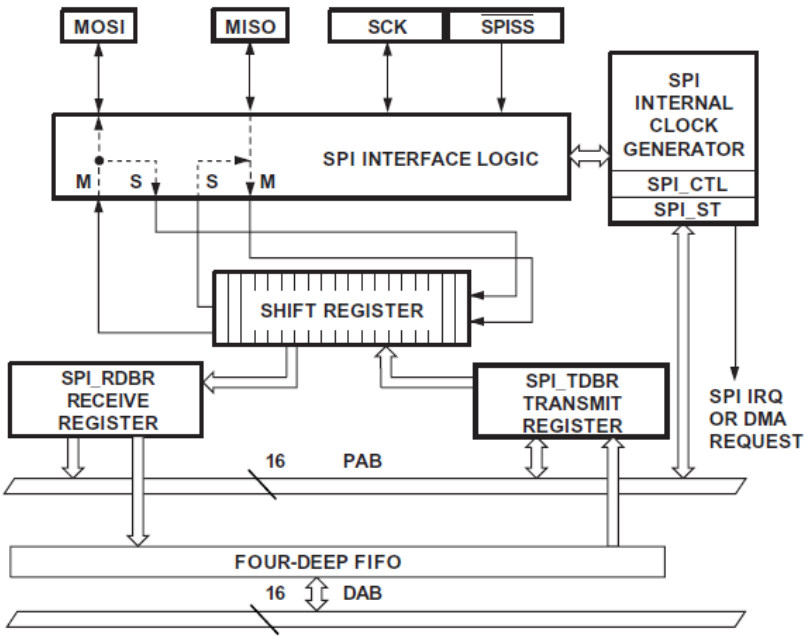
6.3 Az ADSP-BF537 SPI portja

A Blackfin 537-es DSP SPI portja I/O interfészt biztosít az SPI kompatibilis perifériáknak [10]. A következő fontosabb jellemzőkkel rendelkezik:

- teljes duplex, szinkron soros átvitel,
- 8 és 16 bites szóhossz,
- programozható baud rate, órajel fázis, és órajel polaritás,

- multimaster környezet támogatása,
- integrált DMA (Direct Memory Access) vezérlő,
- duplán bufferelt adás és vétel,
- 7 kiválasztó kimenet, 1 kiválasztó bemenet,
- átvitel helyiérték szerinti programozhatósága,
- megszakítás generálása,
- úgynevezett árnyék regiszter a debuggolás segítésére .

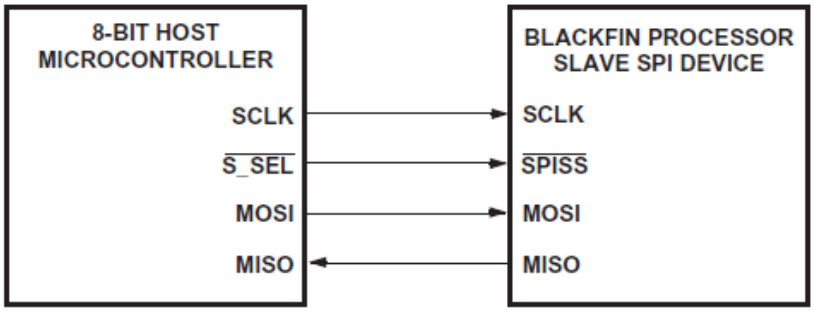
Az SPI port lényegében egy shift regiszter, ami sorosan küld, illetve fogad biteket. A shift regiszter segítségével képes egyidőben küldésre és fogadásra az interfész. Amint egy küldendő bit kitolódik a regiszterből, egy bit fogadása megtörténik a másik végen. Az órajel (SCK) szinkronizálja a biteltolást és az adat mintavételezését a két soros adat vonalon. A legtöbb SPI jel a DSP F portján érhető el. Az öt legfontosabb jel (SCK, MISO, MOSI, /SPISS, /SPISSEL1) nem multiplexelt más perifériával. Alapértelmezés szerint GPIO (General Purpuse I/O) lábként funkcionálnak. Az F port további három slave kiválasztó jelet biztosít, csakúgy, mint a J port. Ezek a jelek már multiplexelve vannak időzítő (timer) és CAN (Controller Area Network) jelekkel. Az SPI blokkvázlatát az alábbi ábra mutatja:



43.ábra: Az ADSP-BF537 SPI interfész blokkvázlata

A jelek funkciói:

- SCK: soros órajel, a master biztosítja, ez a jel vezérli az adatátvitelt. Az órajel fázisa és polaritása változtatható az adatfolyamhoz képest (SPI_CTL regiszter), így meghatározható az SPI üzemmód. A processzor PF13 lábán érhető el.
- MOSI: kétirányú adatláb, amely kimenetként funkcionál a DSP számára master üzemmód esetén, és bemenetként funkcionál a slave egységek számára. Ha a DSP slave üzemmódban van, akkor a MOSI adatfogadó bemenetként funkcionál. A processzor PF11 lábán érhető el.
- MISO: kétirányú adatláb, amely bemenetként funkcionál a DSP számára master üzemmód esetén, és kimenetként funkcionál a slave egységek számára. Ha a DSP slave üzemmódban van (44. ábra), akkor a MOSI adatküldő kimenetként funkcionál. A processzor PF12 lábán érhető el.



44.ábra: Az ADSP-BF537 SPI slave-ként

- /SPISS (Serial Peripheral Slave Select Input Signal): alacsony-aktív jel, amely a processzor engedélyezését határozza meg, amikor az slave üzemmódban van. Bemenetként funkcionál a DSP számára, amit a master egység hajt meg, a PF14 lábon érhető el.
- /SPISSELx (Serial Peripheral Interface Slave Select Enable Output Signals): master módban a Blackfin processzor ezeket a jeleket használja a slave egységek kiválasztására. A jelek alacsony-aktívak, a segítségükkel maximálisan 7 slave egységet lehet illeszteni a DSP-hez. Amennyiben ez nem elegendő, az SPI_FLG regiszter segítségével további lábakat ilyen üzemmódba. A kiválasztó jelek a processzor más-más lábain érhetők el, multiplexelve egyéb jelekkel (45. ábra).

Signal Name	Pin Name	Port Control To Enable Signal
$\overline{\text{SPISSEL1}}$	PF10	Set bit 10 in PORTF_FER = 1
$\overline{\text{SPISSEL2}}$	PJ11	Set PJSE in PORT_MUX = 1
$\overline{\text{SPISSEL3}}$	PJ10	Set PJSE in PORT_MUX = 1
$\overline{\text{SPISSEL4}}$	PF6	Set bit 6 in PORTF_FER = 1 and Set PFS4E in PORT_MUX = 1
$\overline{\text{SPISSEL5}}$	PF5	Set bit 5 in PORTF_FER = 1 and Set PFS5E in PORT_MUX = 1
$\overline{\text{SPISSEL6}}$	PF4	Set bit 4 in PORTF_FER = 1 and Set PFS6E in PORT_MUX = 1
$\overline{\text{SPISSEL7}}$	PJ5	Set PJCE in PORT_MUX = 10

45.ábra: SPI slave kiválasztó jelek elérhetőségei

Az SPI-on történő regiszter írási és olvasási ciklusok során az /SPISSEL jelnek folyamatosan alacsony szintűnek kell lennie az átvitel során (42. ábra). Normál GPIO módba állítjuk ezt a lábat, így a cs_on() és cs_off() függvényekkel addig tudjuk a slave kiválasztó jelet alacsonyan tartani, amíg az szükséges. Az ADSP-BF533 képes slave kiválasztó jeleket automatikusan is generálni, viszont a CC2420 regiszter írási és olvasási ciklusai 4 bájt átviteléből állnak, a DSP SPI egysége automatikus működésben megszakítaná a kiválasztó jel alacsonyan tartását.

Adatküldés során, a tényleges küldés előtt a bitek egy bufferbe töltődnek. Ez az SPI_TDBR regiszter (SPI Transmit Data Buffer). Nagysága 16 bites, írható és olvasható egyaránt. Ugyanígy adatfogadás végén a shift regiszterből az SPI_RDBR (SPI Receive Data Buffer) regiszterbe töltődik az információ. Ez a tároló csak olvasható, szintén 16 bites.

Az adatsebességet master üzemmódban az SPI_BAUD regiszterben állíthatjuk be. A maximális soros órajel frekvencia negyede a rendszer beállított periféria órajel (PCLK) frekvenciájának. A soros órajel frekvenciája a következő formula alapján állítható elő:

```
SCK Frequency = (Peripheral clock frequency SCLK)/(2 x SPI_BAUD)
```

A 46. táblázat néhány lehetséges értéket mutat.

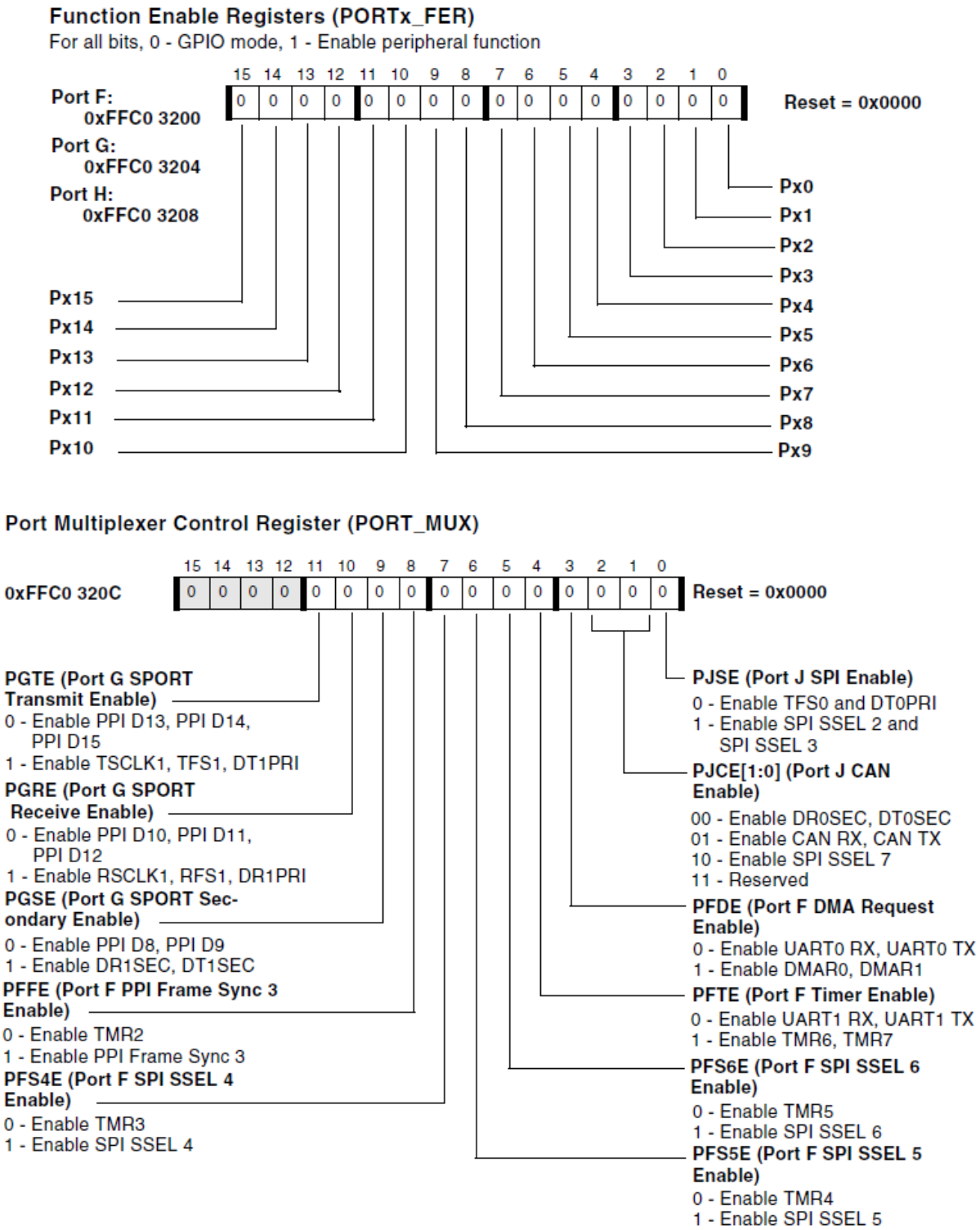
SPI_BAUD Decimal Value	SPI Clock (SCK) Divide Factor	Baud Rate for SCLK at 100 MHz
0	N/A	N/A
1	N/A	N/A
2	4	25 MHz
3	6	16.7 MHz
4	8	12.5 MHz
65,535 (0xFFFF)	131,070	763 Hz

46.táblázat: Lehetséges SPI Baud rate-ek

Az SPI_STAT regiszter segítségével érzékelhető, ha egy SPI átvitel befejeződött. Ez a regiszter bármely időpontbam kiolvasható. Az SPI port rendelkezik két megszakító kimenettel is (adat- és hibamegszakítások).

Master üzemmódban az SPI kommunikáció műkdéséhez a következő lépéseket kell helyesen megtenni:

1. A kiválasztott lábak engedélyezése az F és J portról slave kiválasztó funkcióra. Ez a PORTF_FER (Function Enable Register) és a PORT_MUX (Multiplexer Control Register) regiszterek megfelelő beállításával érhető el (47. ábra). A feladat megoldása az SPI működéshez a PF5, PF11, PF12, PF13 lábakat használja, a PF2, PF3, PF4, PF6, PF10, PF14 bemeneti és kimeneti flag funkciót látnak el.



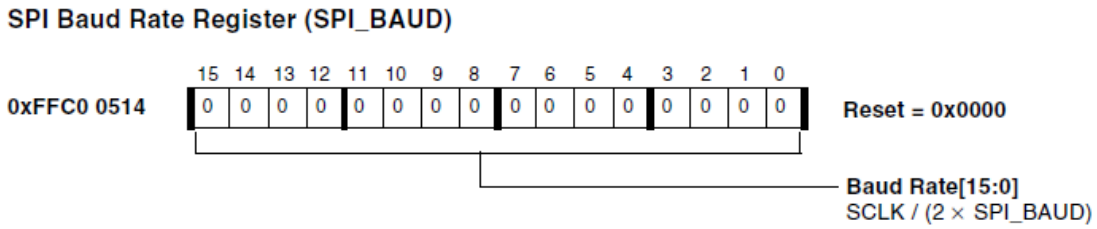
47.ábra: PORT_FER és PORT_MUX regiszterek értelmezése

2. Az SPI_FLG regiszter beállítása. Ezzel biztosíthatjuk, hogy a kívánt slave eszköz ne legyen kiválasztva, amíg a master konfigurálása zajlik.

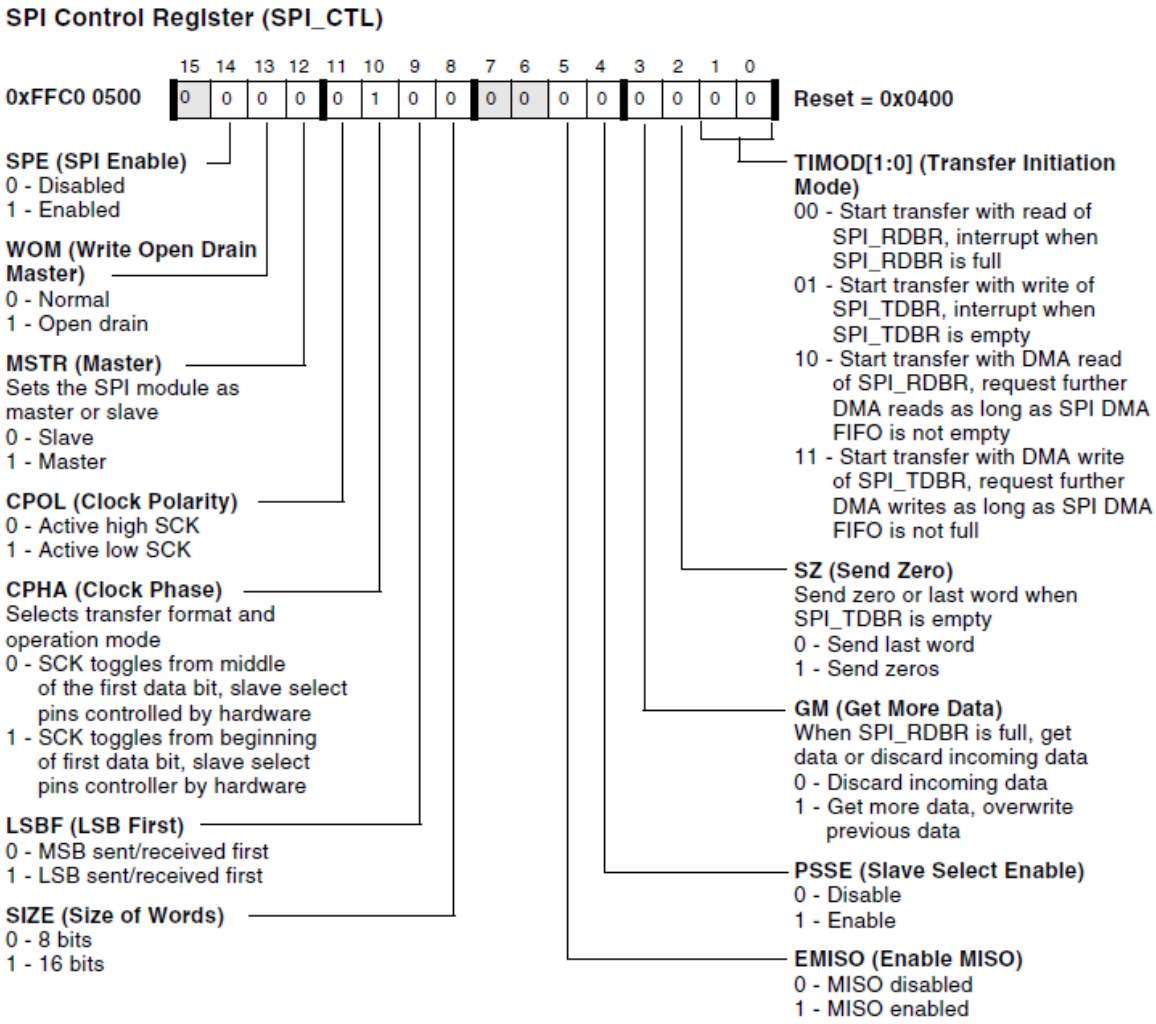
Bit	Name	Function	Port Pin	Default
0		Reserved		0
1	FLS1	$\overline{\text{SPISSEL1}}$ Enable	PF10	0
2	FLS2	$\overline{\text{SPISSEL2}}$ Enable	PJ11	0
3	FLS3	$\overline{\text{SPISSEL3}}$ Enable	PJ10	0
4	FLS4	$\overline{\text{SPISSEL4}}$ Enable	PF6	0
5	FLS5	$\overline{\text{SPISSEL5}}$ Enable	PF5	0
6	FLS6	$\overline{\text{SPISSEL6}}$ Enable	PF4	0
7	FLS7	$\overline{\text{SPISSEL7}}$ Enable	PJ5	0
8		Reserved		1
9	FLG1	$\overline{\text{SPISSEL1}}$ Value	PF10	1
10	FLG2	$\overline{\text{SPISSEL2}}$ Value	PJ11	1
11	FLG3	$\overline{\text{SPISSEL3}}$ Value	PJ10	1
12	FLG4	$\overline{\text{SPISSEL4}}$ Value	PF6	1
13	FLG5	$\overline{\text{SPISSEL5}}$ Value	PF5	1
14	FLG6	$\overline{\text{SPISSEL6}}$ Value	PF4	1
15	FLG7	$\overline{\text{SPISSEL7}}$ Value	PJ5	1

48.ábra: Az SPI_FLG regiszter értelmezése

3. Az SPI_BAUD és az SPI_CTL regiszterek megfelelő beállításával az eszközt master üzemmódba állítjuk, és az SPI rendszerhez meghatározzuk a szóhosszt, átviteli formátumot, baud rate-et és más szükséges paramétereket (49. ábra, 50. ábra).



49.ábra: SPI_BAUD regiszter értelmezése



50.ábra: SPI_CTL regiszter értelmezése

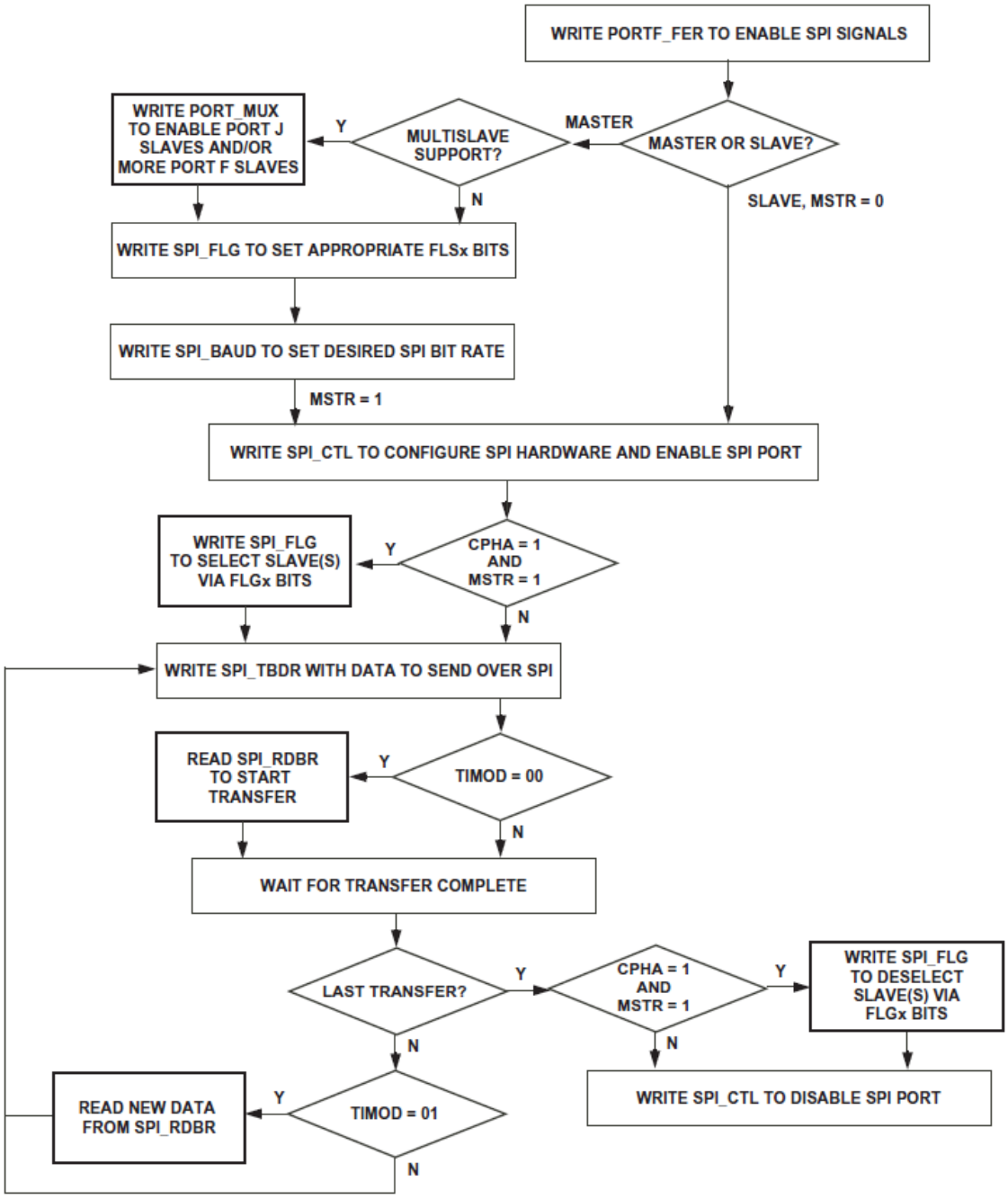
4. Ha CPHA=1 a megfelelő slave egység vagy egységek kiválasztásának kell megtörténnie. Ezt az SPI_FLG regiszter segítségével tehetjük meg, a megfeleő SPI flag biteket (FLGx) törölve (48.ábra).

5. Az SPI_CTL regiszterben található TIMOD bitek (50. ábra) meghatározzák az SPI átvitel kezdeményezésének esetét. Az átvitel az SPI összeköttetésen vagy az SPI_TDBR-be történő íráskor vagy az SPI_RDBR-ből történő olvasáskor kezdődik.

6. Az SPI ezt követően előállítja a programozható órajelet és párhuzamosan kilépteti az adatokat a MOSI vonalra, valamint belépteti az adatokat a MISO vonalról.

7. Minden átvitelkezdeményezéskor az SPI folytatja az adatok küldését és fogadását a beállításainak megfelelően.

A folyamat összefoglalva, folyamatábrán alább látható:



51.ábra: SPI működés Master üzemmódban

6.4 Alacsony szintű SPI rutin

A megvalósító 5 függvény értelmezése:

- `void spi_write_reg(char value)`: 8 bites adatot ír ki az SPI portra, hívás előtt a portot inicializálni kell az erra alkalmas függvénnyel. A CC2420 kiválasztó jelét (/SPISSEL) a megfelelő függvénnyel aktív állapotba kell állítani. Paraméterként a kiírandó bájtot adjuk át. A függvény addig nem tér vissza, amíg sikeres a sikeres írás le nem zajlott.

```
void spi_write_reg(char value)
{
     *pSPI_TDBR = value;
     return;
}
```

- `char spi_read_reg(void)` : beolvassa az SPI portról a 8 bites adatot. A függvény hívása előtt inicializálni kell a portot, valamint gondoskodni a kiválasztó jel aktív állapotáról. Az SPI_STAT regiszter segítségével detektálhatjuk a teljes és helyes SPI átvitelt, illetve a hibás működést. A lenti függvény ezt a regisztert hívja segítségül, hogy a kiolvasás akkor történjen meg, amikor az adat már a bufferben van.

```
char spi_read_reg(void)
{
     short temp;
     char readed_value;

     temp = *pSPI_STAT;
     while( (temp & 0x20) == 0)
     {
          temp = *pSPI_STAT;
     }

     readed_value = *pSPI_RDBR;

     return readed_value;
}
```

- `void spi_init(void)` : inicializálja az SPI portot. A paramétereket a CC2420 adatlapja alapján állítja be.

```
void spi_init(void)
{
     *pSPI_CTL = 0x5025;
    *pSPI_BAUD = 0x2000;
}
```

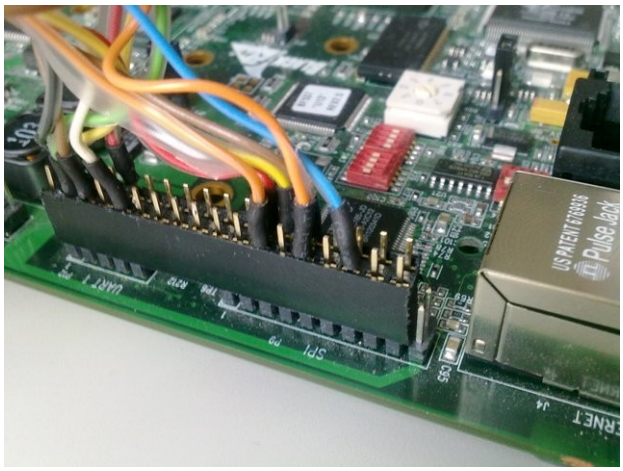
- `void spi_cs_off(void)` : a kiválasztó jelet kikapcsoljuk. Mivel ez egy alacsony-aktív jel, a függvény az /SPISSEL jelet 0-be állítja. Mivel a CC2420-al való kommunikáció során bájtok kerülnek küldésre, alacsony szintű GPIO műveletet jelent.

```
void spi_cs_off(void)
{
     *pPORTFIO_SET = 0x0020;
}
```

- `void spi_cs_on(void)`: bekapcsoljuk az eszközkiválasztást. A függvény az /SPISSEL jelet 1-be állítja.

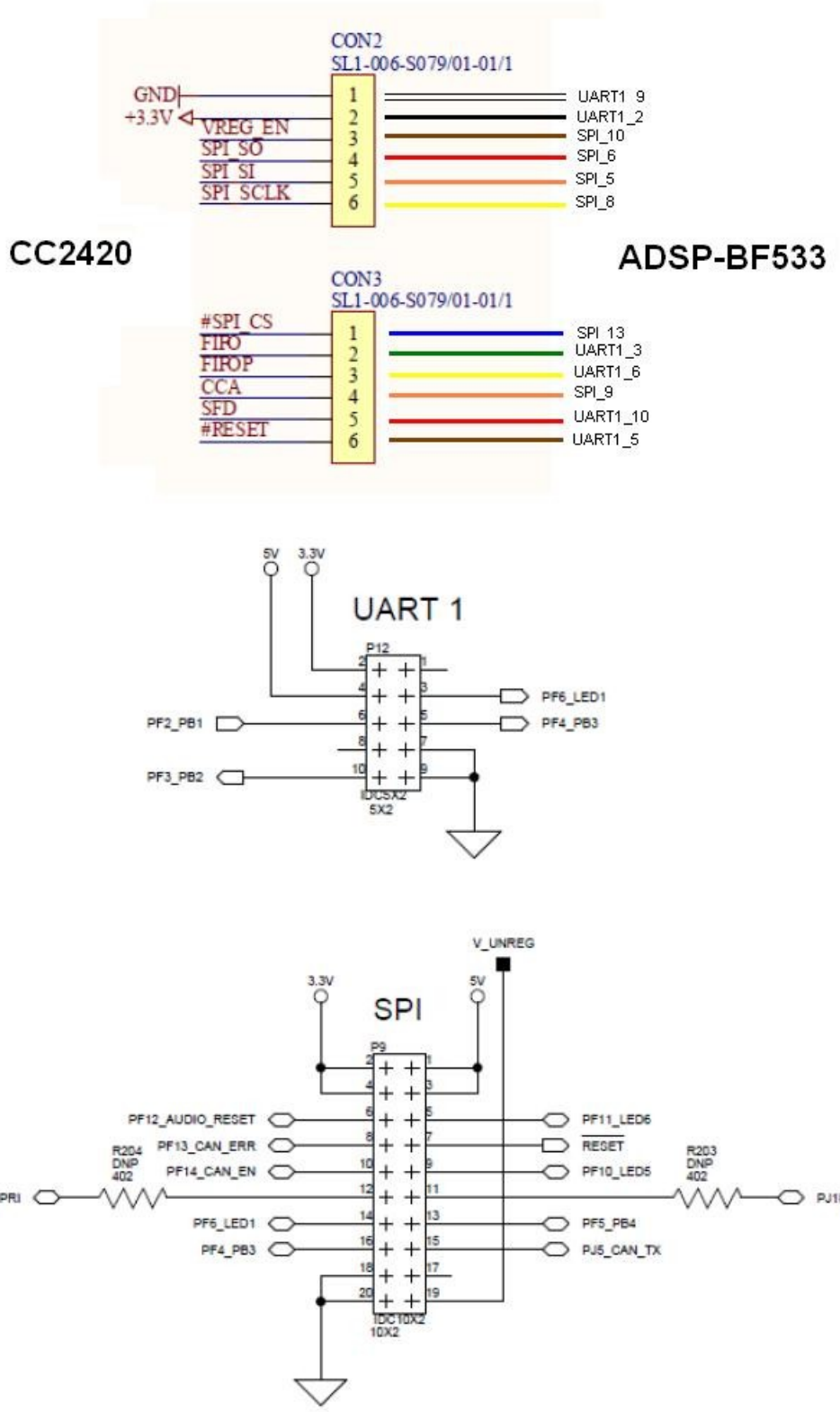
```
void spi_cs_on(void)
{
     *pPORTFIO_CLEAR = 0x0020;
}
```

A két eszköz fizikai összeköttetését egy speciális csatlakozó oldja meg (52. ábra).



52.ábra: SPI csatlakozó

A rádiós kártya CON2 és CON3 csatlakozó valamint a DSP kártya UART1 és SPI csatlakozója között az alábbiak szerint vannak összekötve:

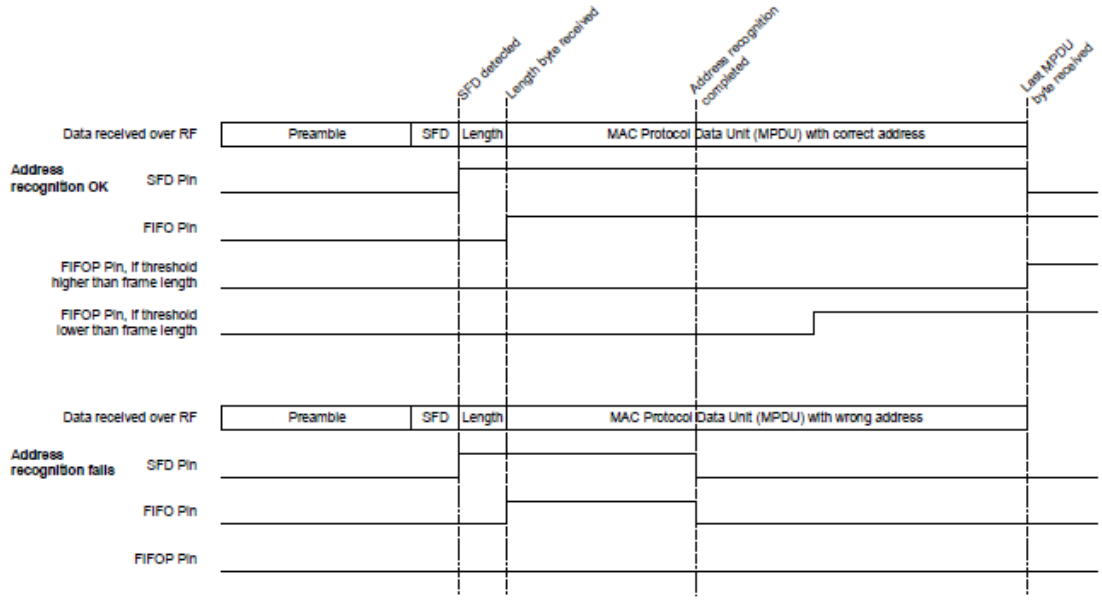


53.ábra: A két processzor összekötése

6.5 Tesztelés

Az összeállított rendszerben egy PC-ről vezérelt CC2420-as adó állomás sugároz egy másik CC2420-as modulnak. Az adó modul a Chipcon Develepmont Kit-hez csatlakozik, míg a vevő SPI porton át kapcsolódik a DSP-hez. A küldött adatokat a DSP kiírja a soros portjára, melyet egy megfelelő átalakítóval a PC USB portjára csatlakoztattam. Így a PC-n nyomon követhető az adatok áramlása egy megfelelően beállított Hyper Terminal kapcsolatban (115200 bit/sec, 8N1).

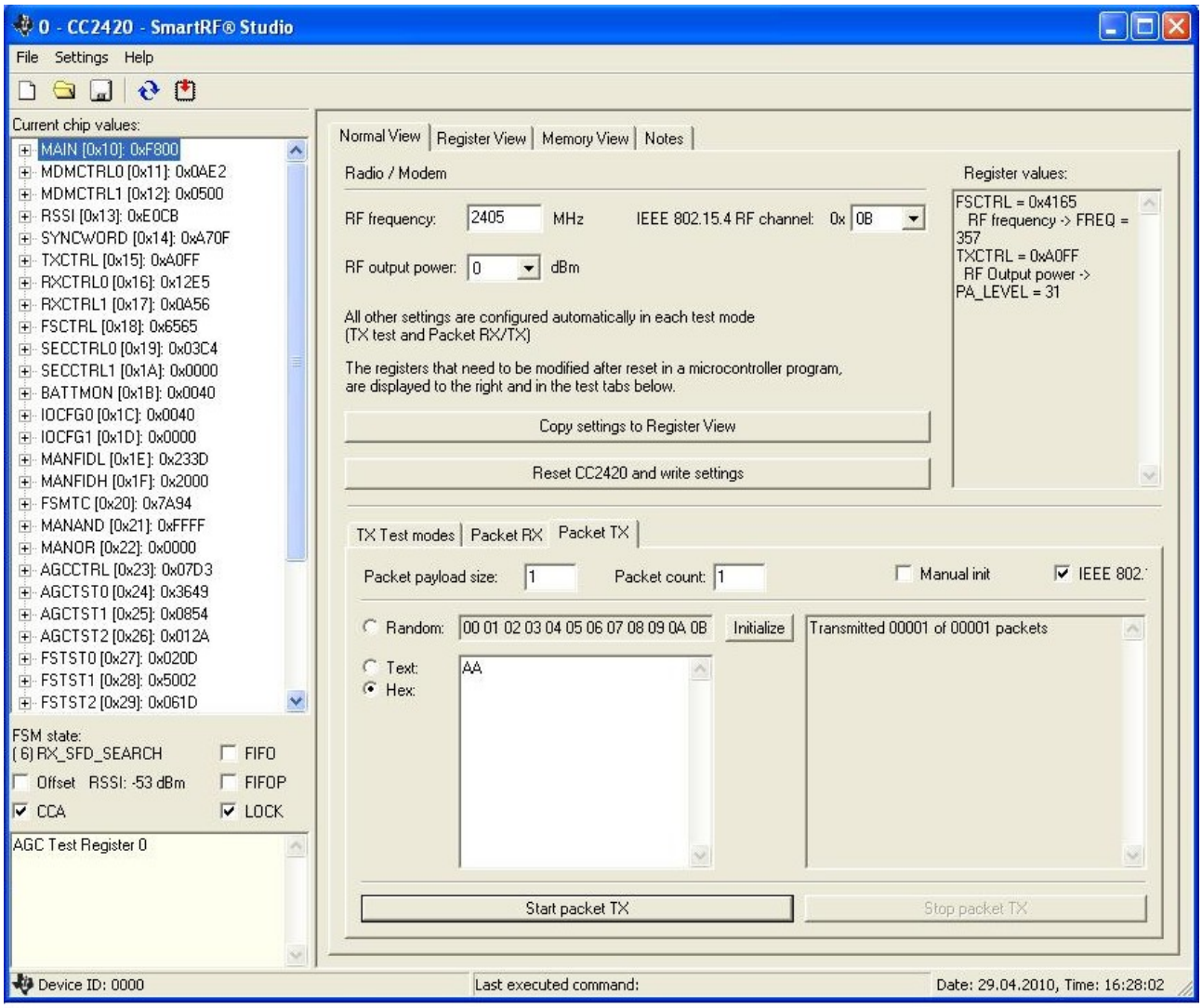
A mérés során a DSP-n egy folyamatosan futó ciklus írja ki a státusz bájtot. Amikor Az RXFIFO-ba megérkezik a vétel során az utolsó bájt is, akkor megfelelő konfigurálás után ezt a CC2420 a FIFOP lábon jelzi (lásd 54. ábra).



54.ábra: FIFOP működése

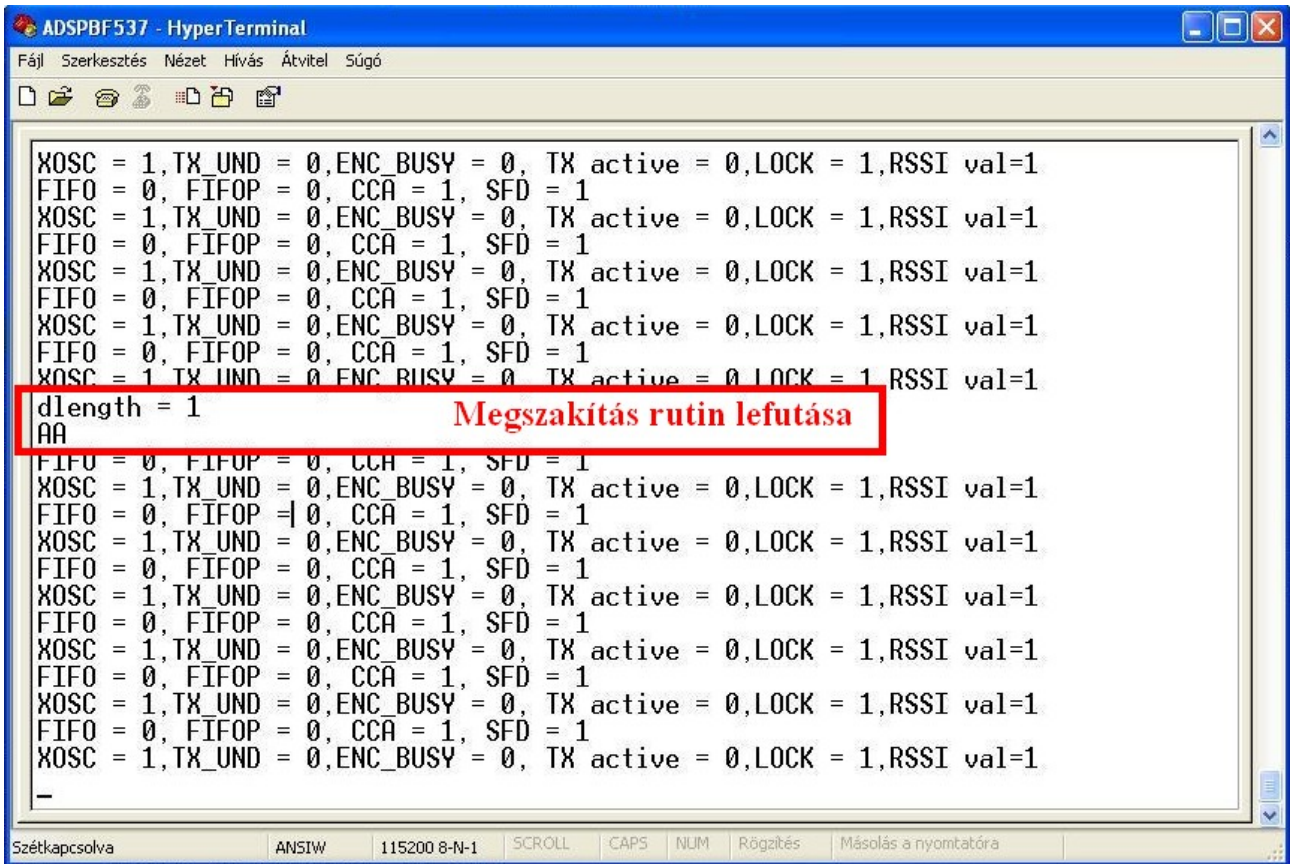
Így a FIFOP-hez rendelt megszakítási rutin egy teljes keret megérkezése után fut le, amennyiben a beállított küszöbérték magasabb a keret hosszánál. A tesztelés során használt rutin kiírja a DSP soros portjára a vett adatot és annak hosszát bájtokban. A mérés során a 1. számú mellékletben látható függvények kerültek felhasználásra.

A szemléltető adatcsere során egy hexadecimális számot (AA) tartalmazó adatcsomagot küld az egyik CC2420-as modul a DSP-hez csatlakoztatott vevőnek. Ezt a művelet a Smart RF Studio már ismert eszközeivel végrehajtható.



55.ábra: Adatcsomag küldése "AA" tartalommal

Azt várjuk, hogy a küldött csomag a vevő oldalon is megjelenjen. Így könnyedén ellenőrizhető a helyes működés. A vétel során az adóvevő státusz bájtját folyamatosan figyelemmel kísérhetjük. Csomag érkezésekor annak tartalmát illetve hosszát megjelenítve:



56.ábra: Az "AA" adatcsomag vétele

Látható, hogy megérkezett a bájtnyi hosszúságú adatcsomag, a tartalma megegyezik a küldött csomag tartalmával. A vételi FIFO-ba történő írás során az eszköz státusz bájtja a MISO lábon jelenik meg (42. ábra). Maga a státusz bájt 6 státusz bitet tartalmaz (57. ábra). A konkrét példában látható, hogy az IC-hez illesztett 16 Mhz-es kristály oszcillátor éppen fut, nem történt FIFO alulcsordulás, a titkosító modul nem fut, rádiós adást nem sugároz az eszköz, a frekvencia szintetizáló PLL zárt állapotban van, illetve érvényes RSSI értéket kapott a vevő.

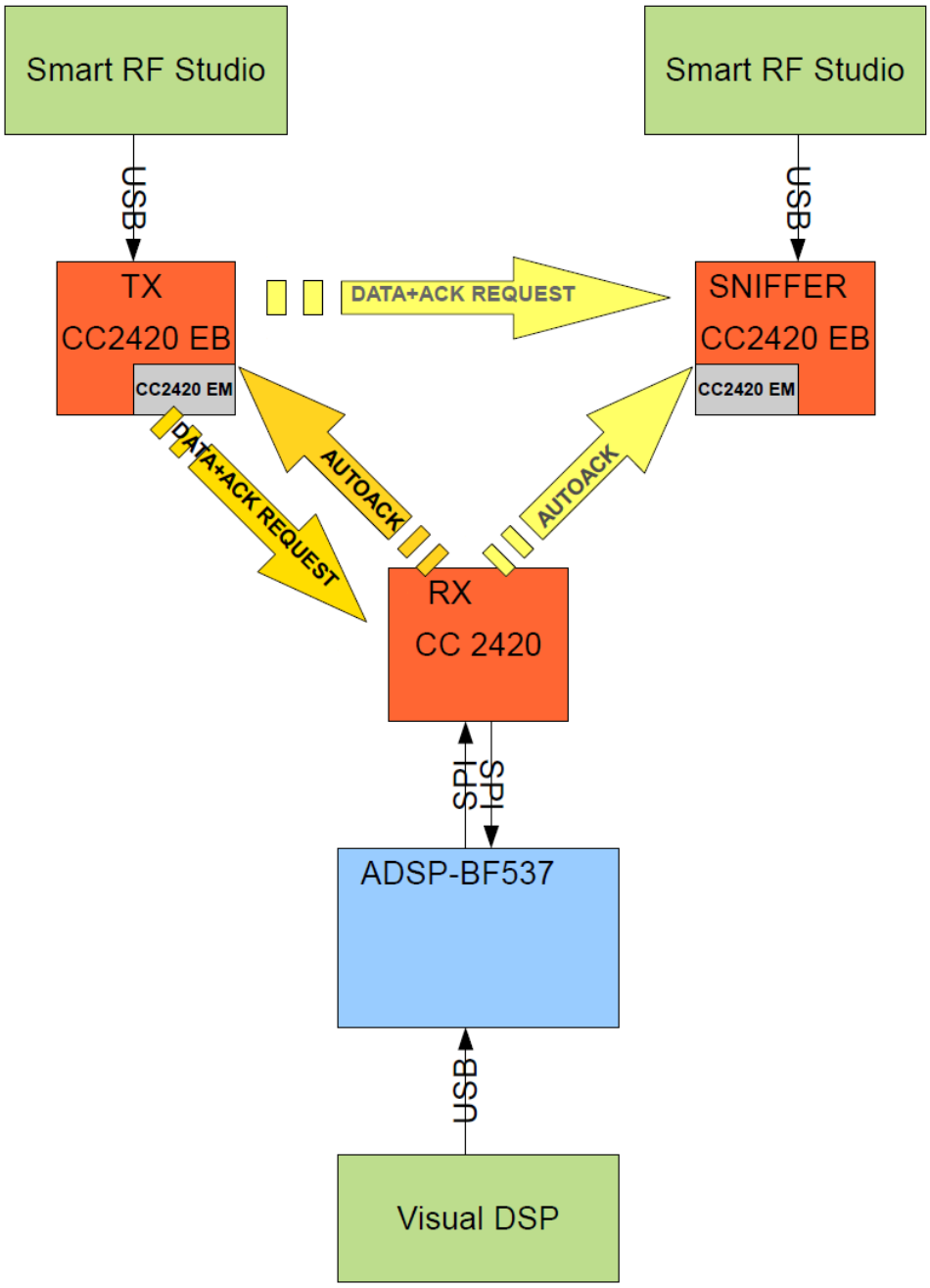
Bit #	Name	Description
7	-	Reserved, ignore value
6	XOSC16M_STABLE	Indicates whether the 16 MHz oscillator is running or not 0 : The 16 MHz crystal oscillator is not running 1 : The 16 MHz crystal oscillator is running
5	TX_UNDERFLOW	Indicates whether an FIFO underflow has occurred during transmission. Must be cleared manually with a SFLUSHTX command strobe. 0 : No underflow has occurred 1 : An underflow has occurred
4	ENC_BUSY	Indicates whether the encryption module is busy 0 : Encryption module is idle 1 : Encryption module is busy
3	TX_ACTIVE	Indicates whether RF transmission is active 0 : RF Transmission is idle 1 : RF Transmission is active
2	LOCK	Indicates whether the frequency synthesizer PLL is in lock or not 0 : The PLL is out of lock 1 : The PLL is in lock
1	RSSI_VALID	Indicates whether the RSSI value is valid or not. 0 : The RSSI value is not valid 1 : The RSSI value is valid, always true when reception has been enabled at least 8 symbol periods (128 us)
0	-	Reserved, ignore value

57.ábra: A CC2420 státusz bájtja

7. Nyugtázás

7.1 A működés demonstrálása

A CC2420 hardveresen támogatja a nyugtaküldést [7]. Ennek demonstrálására alkalmas a következő mérési összeállítás:



58.ábra: Nyugta keret elkapása

Az összeállításban egy CC2420-as adóállomás adatkeretet küld, és a keret megérkezéséről illetve feldolgozásáról nyugtát vár a vevőtől. Ezt az állomást Smart RF Studio segítségével konfiguráljuk. A vevő oldalon beállítjuk a nyugtázás automatikus működését, az állomás DSP-n fut. A párbeszédet lehallgatja egy harmadik állomás (sniffer), amely vételi üzemmódba van kapcsolva. Erre azért van szükség, mert a küldő egységet manuálisan kellene átállítanunk vételi üzemmódba még ezelőtt, hogy a nyugtázó keret megérkezne. Egy második vevő egységgel egyszerűbben megoldható ez a probléma.

Az automatikus nyugtaküldés beállításához az MDMCTRL0.AUTOACK bitet kell beállítanunk megfeleően. Az MDMCTRL0 kontroll regiszter értelmezése a következők szerint történik:

MDMCTRL0 (0x11) - Modem Control Register 0

Bit	Field Name	Reset	R/W	Description
15:14	-	0	W0	Reserved, write as 0
13	RESERVED_FRAME_MODE	0	R/W	Mode for accepting reserved IEE 802.15.4 frame types when address recognition is enabled (MDMCTRL0.ADR_DECODE = 1). 0 : Reserved frame types (100, 101, 110, 111) are rejected by address recognition. 1 : Reserved frame types (100, 101, 110, 111) are always accepted by address recognition. No further address decoding is done. When address recognition is disabled (MDMCTRL0.ADR_DECODE = 0), all frames are received and RESERVED_FRAME_MODE is don't care.
12	PAN_COORDINATOR	0	R/W	Should be set high when the device is a PAN Coordinator. Used for filtering packets with no destination address, as specified in section 7.5.6.2 in 802.15.4, D18
11	ADR_DECODE	1	R/W	Hardware Address decode enable. 0 : Address decoding is disabled 1 : Address decoding is enabled
10:8	CCA_HYST[2:0]	2	R/W	CCA Hysteresis in dB, values 0 through 7 dB
7:6	CCA_MODE[1:0]	3	R/W	0 : Reserved 1 : CCA=1 when RSSI_VAL < CCA_THR - CCA_HYST CCA=0 when RSSI_VAL ≥ CCA_THR 2 : CCA=1 when not receiving valid IEEE 802.15.4 data, CCA=0 otherwise 3 : CCA=1 when RSSI_VAL < CCA_THR - CCA_HYST andnot receiving valid IEEE 802.15.4 data. CCA=0 when RSSI_VAL ≥ CCA_THR or receiving a packet
5	AUTOCRC	1	R/W	In packet mode a CRC-16 (ITU-T) is calculated and is transmitted after the last data byte in TX. In RX CRC is calculated and checked for validity.
4	AUTOACK	0	R/W	If AUTOACK is set, all packets accepted by address recognition with the acknowledge request flag set and a valid CRC are ack'ed 12 symbol periods after being received.
3:0	PREAMBLE_LENGTH [3:0]	2	R/W	The number of preamble bytes (2 zero-symbols) to be sent in TX mode prior to the SYNCWORD, encoded in steps of 2. The reset value of 2 is compliant with IEEE 802.15.4, since the 4th zero byte is included in the SYNCWORD. 0 : 1 leading zero bytes (not recommended) 1 : 2 leading zero bytes (not recommended) 2 : 3 leading zero bytes (IEEE 802.15.4 compliant) 3 : 4 leading zero bytes ... 15 : 16 leading zero bytes

59.ábra: Modem Control0 Regiszter értelmezése

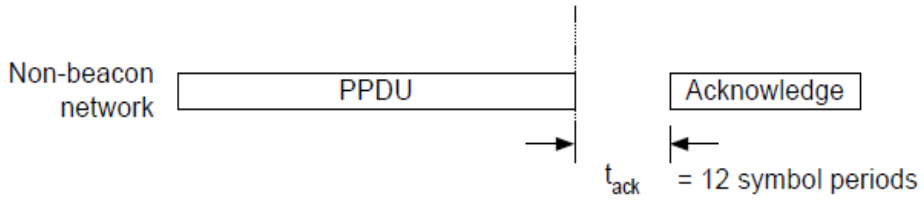
Az automatikus nyugtázási funkciót a következő függvény látja el:

```
void cc2420_turn_on_auto_ack(void)
{
        unsigned short temp_short;

    ReadRegister(CC2420_CONTROL_REGISTER_MDMCTRL0,&temp_short);
    temp_short &=  0xFFEF; //clear bits, see cc2420 data sheet page 64.
    temp_short += 0x10;
    WriteRegister(CC2420_CONTROL_REGISTER_MDMCTRL0,temp_short);

}
```

Ha az automatikus nyugtaküldés beállítása megtörtént, minden olyan keret fogadását követően nyugta keretet küld az egység, amely nyugtázást kér, és megfelelő formátumú a címe, illetve az ellenőrző szekvencia a keret végén érvényes. Ebből következik, hogy a cím felismerő funkciónak és az automatikus keret ellenőrző funkciónak is engedélyezettnek kell lenni a helyes működéshez. Ilyen esetben a beérkező keret utlsó szimbólumát követően 12 szimbólum periódussal történik meg a nyugtázás.

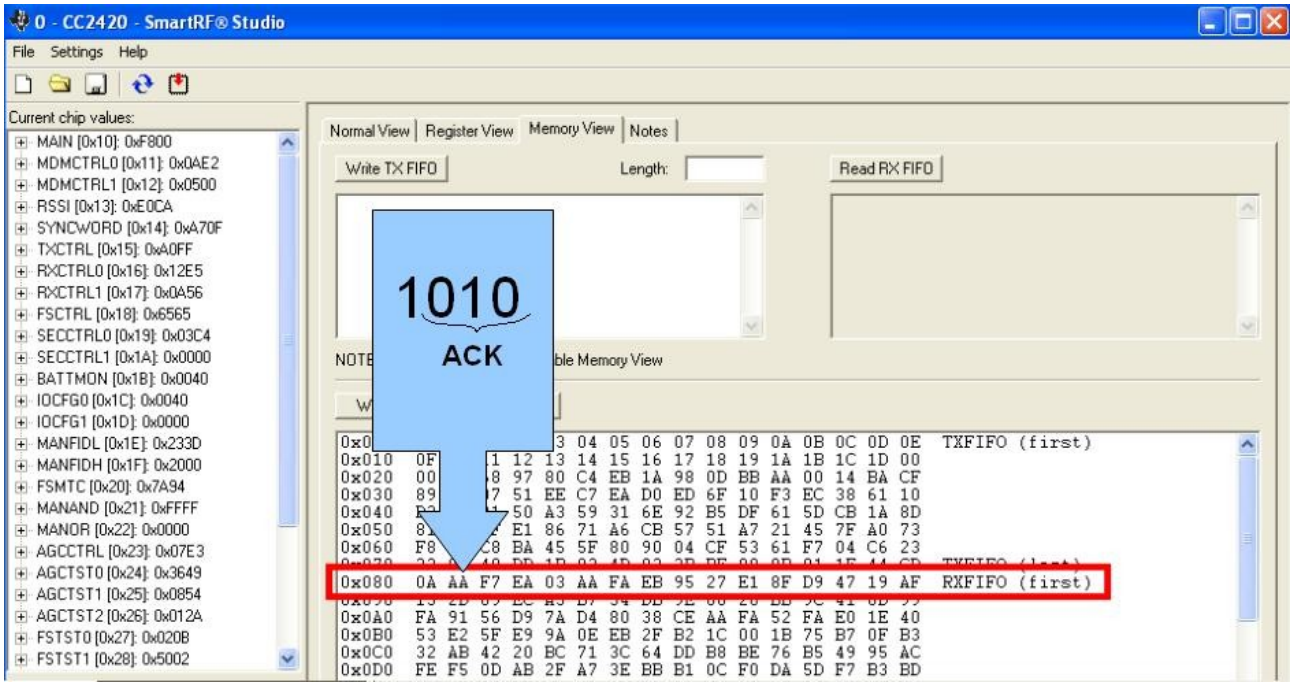


60.ábra: A nyugtázás időzítése non-beacon hálózatok esetében

A nyugta keret kiolvasása a vevő oldalon (sniffer) ezek után végrehajtható, a Smart RF Studio memória nézetében kiolvassuk az RxFIFO regiszter tartalmát. Az IEEE 802.15.4 szabványnak megfelelően felépített keretek a hálózati réteg kommunikációs egységeit képezik. Az ilyen keretek kiolvasása után a biteket értelmeznünk kell. A kiolvasott értékek hexadecimális formában jelennek meg a Smart RF Studioban. A keret felépítésének tudatában (14. ábra) értelmezhetjük a fogadott információt. Az első bájt jelenti a keret hosszát. A hosszat az SFD-t követő bájtok jelentik, nem számítva magát a hosszbájtot. A második és harmadik bájt a keret kontrollmező, melynek tartalma a 3. szakaszban értelmezésre került. Ez alapján a 0-2 bitek jelentik a keret típusát. A hozzárendelés a következő:

000 Beacon
001 Data
010 ACK
011 MAC command
100-111 Reserved

A vételi FIFO olvasásakor figyelembe kell vennünk továbbá, hogy a kontroll mező bájtjai helyi érték szerint csökkenő sorrendben érkeznek ("big-endian"). Azaz vételkor a második bájtot kell figyelnünk, annak is az utolsó három bitjét ahhoz, hogy megállapítsuk milyen típusú kerettel van dolgunk. A fenti ismeretek tudatában megállapíthatjuk, hogy a mérés során a lehallgató állomás RxFIFO-jába megérkezett az a nyugta keret, amelyet a DSP-hez csatlakoztatott vevő állomás küldött vissza az adónak (lásd 61. ábra).



61.ábra: RxFIFO olvasása

7.2 Nyugtázó függvény megvalósítása

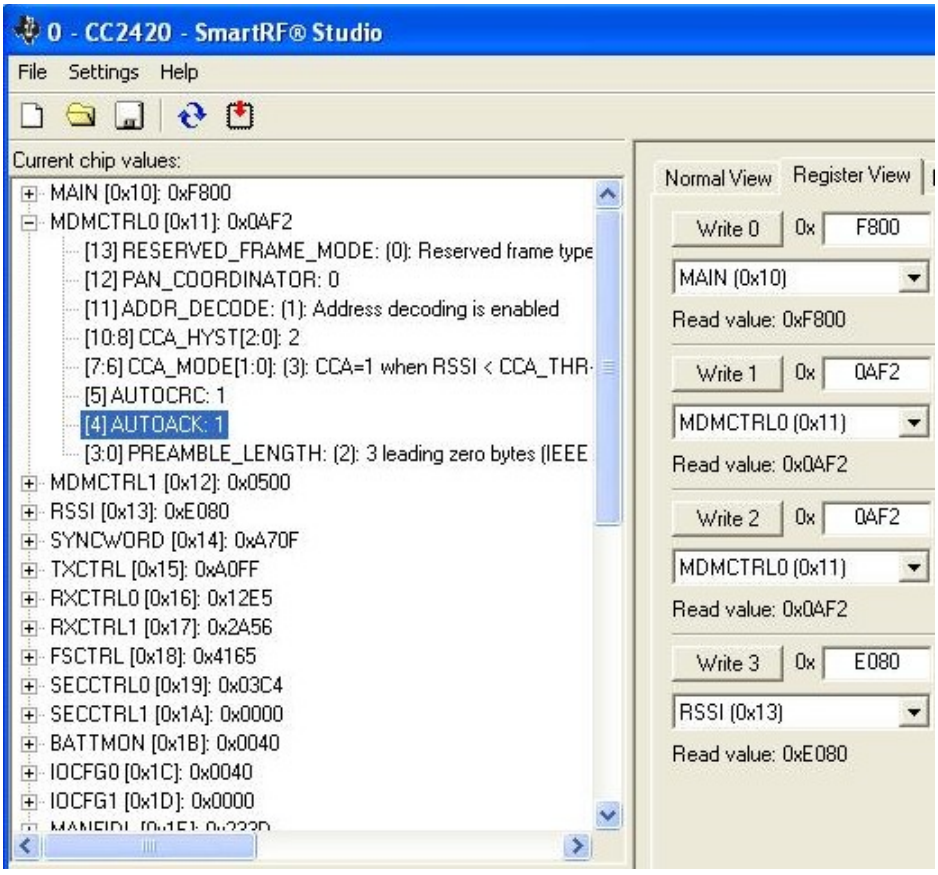
Az előző fejezetben láthattuk, hogy a CC2420 képes automatikusan nyugtázni. Ahhoz, hogy a rádiós IC illesztése teljes legyen, a nyugtafogadást is meg kell valósítani. Ehhez az szükséges, hogy a nyugta keretet képesek legyünk megkülönböztetni a többitől. A megoldás azt használja ki, hogy a nyugta keretek hossza előre ismert. A közölt keretek felépítését írja le a 3. fejezet.

Bytes:	4	1	1	2	1	2
Preamble Sequence		Start of Frame Delimiter (SFD)	Frame Length	Frame Control Field (FCF)	Data Sequence Number	Frame Check Sequence (FCS)
Synchronisation Header (SHR)			PHY Header (PHR)	MAC Header (MHR)		MAC Footer (MFR)

62.ábra: Nyugta keret

A funkciót megvalósító függvény tehát olyan keretek érkezését várja, amelyek a hossz mezőn kívül öt bájtot tartalmaznak.

A mérési elrendezésben két állomás van jelen. Az egyik állomás a Blackfin DSP-hez csatlakozik, amelyen a szűrő program fut, tehát vételi üzemmódban fut. A másik állomás egy IEEE 802.15.4 szabványnak megfelelő nyugta keretet küld.

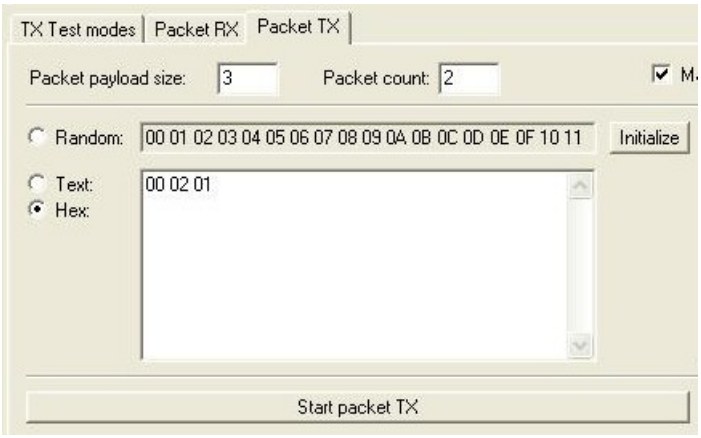


63.ábra: Nyugtaküldő állomás beállítása

Az Chipcon adóvevő MDMCTRL0 kontroll regiszterének értékét 0x0AF2-re állítjuk (63. ábra). A beállítások miatt automatikusan generálódó CRC ellenőrző mező a keret végére kerül (AUTOCRC=1).

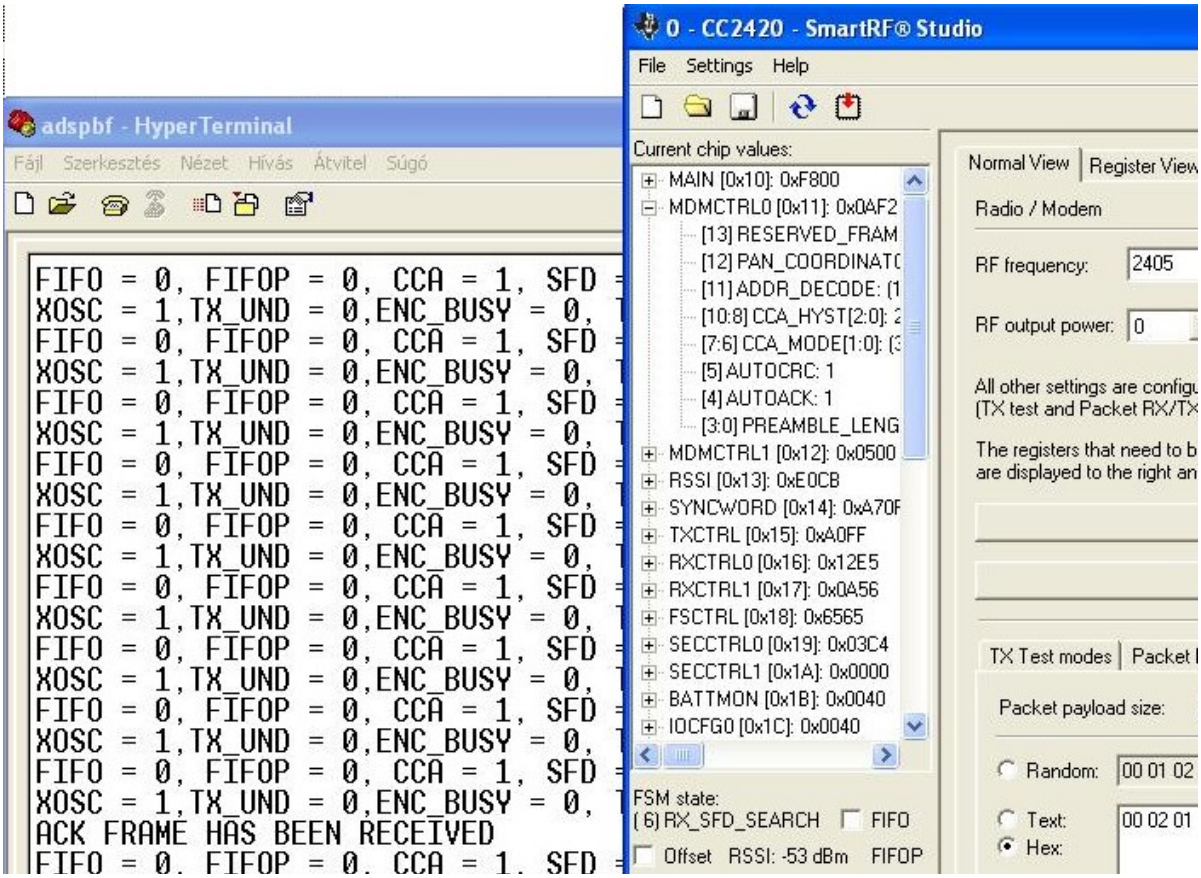
A beállítások elvégzése után egy 0x01 adat szekvencia számú (Data Sequence

Number=DSN) nyugta keretet küld ki az egység (a 0x00 0x02 értékek jelentik a nyugta keret típust):



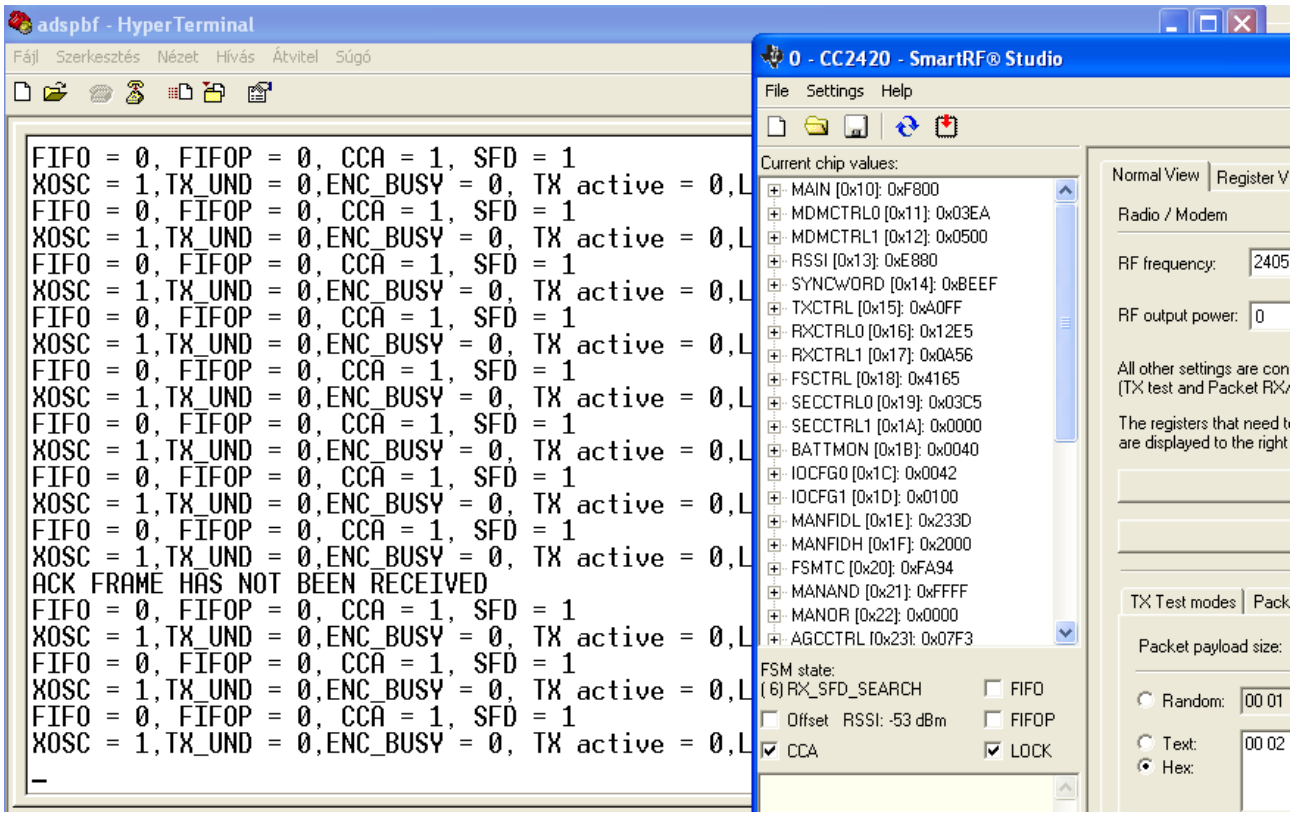
64.ábra: Nyugta keret kiküldése

A DSP-n futó függvény végrehajtja az RxFIFO olvasását SPI interfészen keresztül, majd eltárolja tartalmának hosszát bájtokban. Ezt minden FIFOP-n érkező megszakításkor megteszi. Amennyiben ez az érték nyugta keretet jelez, akkor ezt jelzi a felhasználónak egy UART üzenet formájában. Amennyiben nem nyugta típusú keret érkezett, erről is informálja a felhasználót. A megszakítási rutin a 2.számú mellékletben található. A fenti beállításokkal az adó állomás képes nyugtázást kérő bejövő adatkeretek részére nyugta keretek visszaküldésének megvalósítására is. Ilyenkor a DSN mezőt a bejövő keretből másolja át a CC2420. A függvény helyes működése a 65. ábrán látható.



65.ábra: ACK keret felismerése

Nem szabványos nyugta keret kiküldésével bizonyosodhatunk meg a program működésének helyességéről:



66.ábra: A működés ellenőrzése

8. Összefoglalás

A hálózati alapok összegzését követően a CC2420 által megvalósított IEEE 802.15.4 szabványt mutatja be a dolgozat. Elkészült a CC2420 és a BF-537-es hardveres összekötése és a két egység kommunikációjának szoftveres megvalósítása.

A két vezérlő illesztése során implementált nyugtázási funkciók a vezeték nélküli hálózati kommunikáció biztonsági mechanizmusát valósítják meg. Ez magában foglalja a nyugtázási kérelemmel beérkező adatkeretek számára az automatikus nyugtázási funkciót, melyet a CC2420 hardveresen támogat. A működés prezentálásához egy mérési elrendezés került összeállításra, ahol egy lehallgató állomás segítségével sikerült nyugta keretet venni, amely másik két állomás kommunikációja során került forgalomba. A második mérés a nyugta keretek vételét demonstrálja. A megvalósított C függvény figyeli, hogy a bejövő keret hossza megegyezik-e a szabvány által meghatározott nyugta keret hosszával. A küldés és vétel folyamatának megértését követően ilyen módon prezentálásra került a rádiós kommunikáció illetve az alacsony szintű SPI átviteli rutin helyes működése.

Amennyiben egy több résztvevős hálózatot szeretnénk ezekkel az eszközökkel megvalósítani, akkor a DSP-re implementálnunk kell a 802.15.4 szabvány által meghatározott CSMA/CA algoritmust, amely a szakdolgozatban szintén bemutatásra került. A csatorna hozzáférések ilyen módon történő szabályozását a CC2420 hardveresen támogatja, a továbbfejlesztéshez szükséges információt a megfelelő lábakon biztosítja. A MAC protokoll helyes alkalmazásával adó állomások adatainak ütközését és elvesztését akadályozhatjuk meg a vevői oldalon, és egy biztonságosabb hálózatot építhetünk ki.

Irodalomjegyzék:

- [1] Andrew S. Tannenbaum: Számítógép-hálózatok , Panem, 2003
- [2] James T. Geiger: Wireless Networking Handbook, New Riders, 1996
- [3] Cooklev, Todor: Wireless Communicaton Standards, IEEE Press, New York, 2004
- [4] ZigBee Specification, ZigBee Standards Organization, San Ramon, 2008
- [5] IEEE Std 802.15.4d™, New York, 2009
- [6] Sushant Jain, Raul Mahajan: Wireless LAN MAC Protocolls, Washingtom, 2000
- [7] CC2420 Datasheet: 2.4 GHz IEEE 802.15.4 /ZigBee-ready RF Transceiver, Dallas, 2006
- [8] CC2420DK Development Kit: Quick Start Instructions, Dallas, 2006
- [9] SmartRF® Studio User Manual Rev. 6.10.1, Dallas, 2008
- [10] ADSP-BF537 Blackfin® Processor Hardware Reference, Norwood, 2005
- [11] ADSP-BF537 EZ-KIT Lite® Evaluation System Manual, Norwood, 2005
- [12] Visual DSP++ 4.5 Getting Started Guide, Norwood, 2006
- [13] ADSP-BF53x/BF56x Blackfin® Processor Programming Reference, Norwood, 2006
- [14] Dr. Tevesz Gábor: Mikrokontroller alapú rendszerek, Budapest, 2008
- [15] Steven W. Smith, Ph.D.: The Scientist and Engineer's Guide to Digital Signal Processing, 1997
  http://www.dspguide.com

Mellékletek

1.számú melléklet (Test/main.c)

```
#include <sysreg.h>
#include <ccblkfn.h>
#include <cdefBF537.h>
#include <stdio.h>

#include "cc2420_driver.h"
#include "cc2420_lowlevel_driver.h"
#include "gpio_lowlevel_driver.h"
#include "cc2420_rx.h"
#include "uartlib.h"
#include "pll.h"
#include "print_status.h"

//-----------------------------------------------------

unsigned char databuf[256];

char temp_str[1024];

void proba(void)
{
        unsigned int dlength;
        unsigned int ii;

        rxfifo_getnext_packet();
        dlength = rxfifo_get_data_length();
        rxfifo_get_data(databuf);

        sprintf(temp_str,"dlength = %X\r\n",dlength);
        uart_send_char(temp_str);

        if(dlength>0)
        {
                for(ii=0;ii<dlength;ii++)
                {
                        sprintf(temp_str,"%2X ",databuf[ii]);
                        uart_send_char(temp_str);
                }
                sprintf(temp_str,"\r\n");
                uart_send_char(temp_str);
        }

    Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SFLUSHRX);
    Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SFLUSHRX);

}


void main(void)
{

    int kk;
    char ch;
    char current_status;
    CC2420_Status current_status_cc2420;

    char pin_FIFO,pin_FIFOP,pin_CCA,pin_SFD;

    Init_PLL();
    init_uart();
    uart_send_char("Started...\r\n");
    cc2420_init()
    install_interrupt_handler_fifop(proba);
    cc2420_Setup_802_14_5_Channel(0x0B);
    cc2420_setup_PANID(0x2420);
    cc2420_setup_short_address(0x1234);
    cc2420_turn_on_address_decode();
```

```
    cc2420_turn_off_auto_ack();
    cc2420_turn_on_PAN_coordinator();

    cc2420_setup_FIFOP_threshold(2);

    Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SRXON);
    Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SFLUSHRX);
    Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SFLUSHRX);

    kk = 0;

    while(1)
    {
        kk++;
        if(0)
                        PrintfPinStatus();


        if((kk % 1500000) == 0)
        {

Uart0_PrintfPinStatus();
ch = Write_Command_StrobeRegister(CC2420_COMMAND_STROBE_SNOP);
ParseStatusByte(ch,&current_status_cc2420);

sprintf(temp_str,"XOSC = %d,TX_UND = %d,ENC_BUSY = %d, TX active = %d,LOCK = %d,RSSI val=
%d\r\n",current_status_cc2420.xosc16M_stable,current_status_cc2420.tx_underflow,current_status_cc24
20.enc_busy,current_status_cc2420.tx_active,current_status_cc2420.lock,current_status_cc2420.rssi_v
alid);

uart_send_char(temp_str);

       }

        if(0)break;
    }

    return;
}
```

2.számú melléklet (Ack/main.c)

```
void ack_check(void)
{
        unsigned int dlength;
        unsigned int ii,kk;
        unsigned char rxfiforead;

        for(ii=0;ii<100;ii++)
                                {
                                asm("nop;");
                                }

        rxfifo_getnext_packet();
        dlength = rxfifo_get_data_length();
        //read_rx_fifo(&rxfiforead,2);

        if(dlength==5)
                {
                ack=1;
                sprintf(temp_str,"ACK FRAME HAS BEEN RECEIVED\r\n");
                uart_send_char(temp_str);
                }
        else
                {
                ack=0;
                sprintf(temp_str,"ACK FRAME HAS NOT BEEN RECEIVED\r\n");
                uart_send_char(temp_str);

                }
```