Alma Mater Studiorum · Università di Bologna

facoltà di scienze matematiche, fisiche e naturali
Corso di Laurea in Informatica

FritzBee
Router multifunzionale con supporto del protocollo 802.15.4/Zigbee

Tesi di Laurea in Basi di Dati

Relatore:
Chiar.mo Prof.
Danilo Montesi

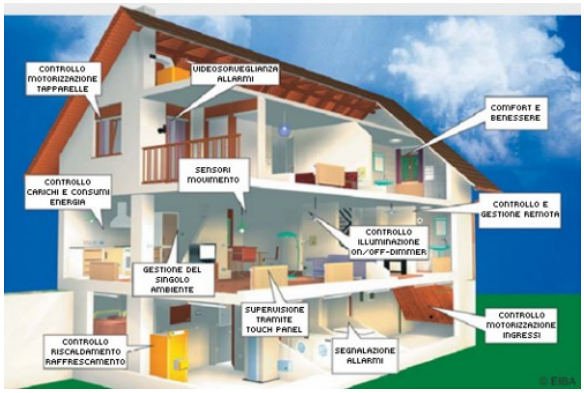
Presentata da:
Matias Plumari

Sessione I
Anno Accademico 2009/2010

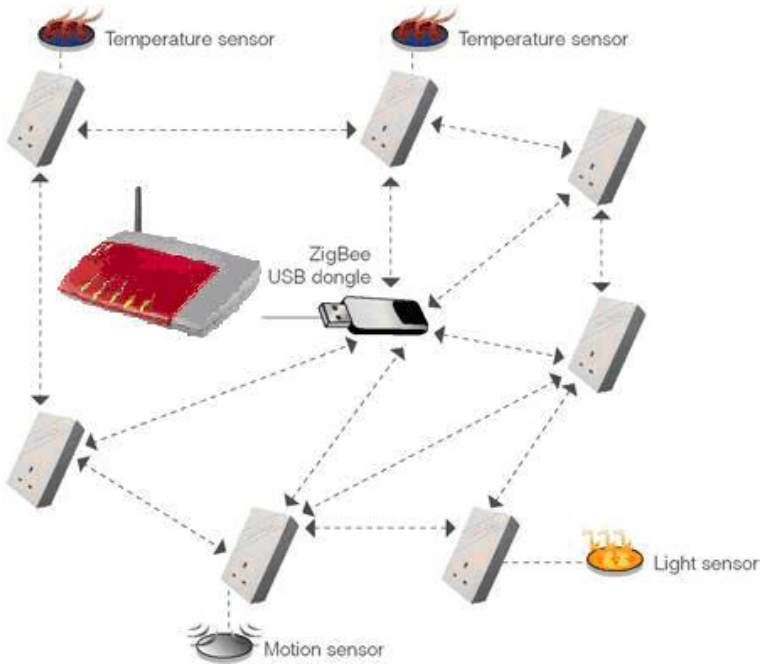
Introduzione

Negli ultimi anni,il tema dell'energia è divenuto sempre piú di vitale importanza e le attenzioni rivolte ad un uso intelligente di essa hanno fatto si che le fonti energetiche siano usate trovando un punto d'incontro con un altro tema fondamentale: l'ambiente e la sua protezione. Questa tesi si basa su questo principio, scritta e pensata per promuovere uno sviluppo verso quelle tecnologie in grado di gestire, monitorare e controllare che l'energia venga correttamente gestita, al fine di evitare sprechi. Altro punto fondamentale di questa tesi nasce da un'altra ben diversa riflessione. Di questi tempi,molti sono i dispositivi ed elettrodomestici che ci circondano, praticamente ogni uomo appartenente ad un paese modestamente sviluppato possiede un cellulare,una casa con decine di tanti apparecchi elettronici, computer, cellulari, televisori, lettori (dvd, cd); questa crescita esponenziale e questa integrazione sempre piú radicata della tecnologia nella vita di un uomo fa si che le persone posseggano svariati telecomandi,ognuno con uno standard proprio determinato dal proprio produttore o da alleanze tra piú produttori, rendendo la vita sempre piú complicata. Sarebbe molto piú piacevole per una persona controllare e gestire praticamente ogni dispositivo elettronico, utilizzando un solo telecomando, un solo controllo; si pensi ad esempio alla quantitá di oggetti che la persone devono portarsi dietro, solo per far fronte a questo problema (cellulare, chiavi dell'auto, chiave elettronica per garage o per cancello, chiavi di casa). Una realtá dove venga usato un solo standard di comunicazione è una realtá ben lontana da quella in cui si vive oggi, la concorrenza, il mercato, giocano le loro parti in questa gara per ottenere un controllo sempre

maggiore di piú dispositivi.



L'oggetto della tesi in questione è FritzBee, un dispositivo che non ha nulla di nuovo, spesso infatti reinventare la ruota non serve, ma basta ingegnarsi per capire cosa puó essere riusato, cosa puó essere assemblato a partire da dispositivi o tecnologie gia esistenti. FritzBee nasce dall'integrazione di un protocollo basato sullo standard Zigbee(protocollo di comunizaione 802.15.4) su un router multifunzionale che gestisce gia altri protocolli: 802.11b/g e Dect: il Fritz!Box. L'idea è stata quella di fornire una sorta di magic box che fosse in grado di gestire e controllare il piú possibile l'ambiente casalingo o comunque di un ambiente non industriale e che fosse alla portata di tutti (HOME AREA NETWORK). Il protocollo Zigbee è infatti usato principalmente nella domotica e si presenta come uno standard composto da vari profili, quali possono essere l'home automation, la gestione della sicurezza e infine l'energia intelligente, tema fondamentale sulla quale si è poi sviluppata la tesi. Nel primo capitolo della tesi si parlerá dello standard Zigbee e di come esso viene applicato, nel secondo verranno presentate le varie funzionalitá del router tedesco Fritz!Box mentre nel terzo verrá descritto come si è integrata la tecnologia Zigbee all'interno del router per garantire la comunicazione tra di essi.



La tesi puó essere vista come l'introduzione ad un progetto molto piú articolato e complesso, fornendone un buon punto d'inizio atta ad offrire uno spunto per sviluppare in futuro dispositivi sempre piú complessi in grado di garantire l'interoperabilitá tra tecnologie e standard che con il passare degli anni andranno ad aumentare sempre piú. Di seguito sono riportati gli scopi sulla quale si è basata la tesi:

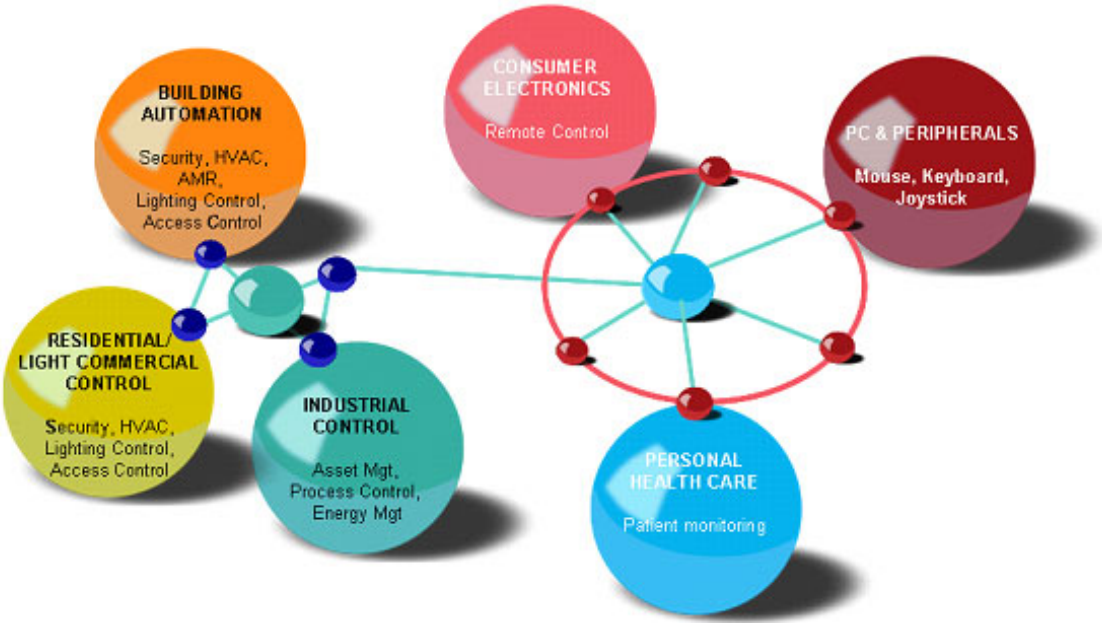
- Garantire l'interoperabilitá tra tecnologie diverse;
- Ottenere uno strumento che sia in grado di minimizzare i supporti di controllo tra dispositivi che usano comandi diversi per semplificare la vita degli utente;
- Utilizzo intelligente dell'energia con particolare attenzione all'uso che se ne fa e dell'impatto che essa puó avere sull'ambiente, questo scopo viene garantito dall'utilizzo di dispositivi sviluppati che richiedono un basso consumo energetico

Indice

Elenco delle figure

Capitolo 1

Lo standard Zigbee



1.1 Cos'è

ZigBee è il nome utilizzato per indicare una specifica per un particolare insieme di protocolli di comunicazione, che basano il loro funzionamento sull'impiego di radio digitali a bassa potenza; queste reti sono nate a seguito della necessitá di molti ingegneri e progettisti di poter sviluppare reti radio digitali dedicate e con una propria autonomia a basso consumo da destinare alle nuove applicazioni. Dopo un primo periodo di scarso interessamento da parte di aziende e societá del settore, ZigBee ha iniziato a diffondersi rapidamente, riscuotendo parecchio successo soprattutto per soluzioni embedded in ambito industriale, in particolare in quello delle telecomunicazioni e nel settore della domotica per creare, ad esempio, una rete intelligente di sensori; la caratteristica principale di questo insieme di protocolli è sicuramente il basso consumo per tutte le applicazioni con un tranfer rate ridotto: teoricamente, una rete basata su ZigBee riuscirebbe a gestire tutti i suoi nodi sfruttando sempre una sola batteria per almeno due anni. Le potenzialitá di questo prodotto sono enormi, ZigBee ha permesso a molti sviluppatori di realizzare e offrire soluzioni professionali a basso costo, che altrimenti sarebbero state difficili da ottenere, sia dal punto di vista della strutturazione e della comunicazione dell'hardware, sia dal punto di vista dell'efficienza.

1.2 Le specifiche dello standard

Il protocollo Zigbee si basa sullo standard IEEE 802.15.4 che definisce il layer di comunicazione di livello due nel modello OSI e destinato a essere utilizzato all'interno di reti private, caratterizzate da basse velocitá di trasferimenti dei dati; sono previste tre bande diverse di frequenze operative, ognuna delle quali con un numero prestabilito di canali per un totale di ventisette.

Le sue regole di funzionamento lo rendono un sistema abbastanza robusto in presenza di rumore, in quanto prima di inviare le informazioni verso il layer

fisico queste vengono modulate utilizzando la tecnica del DSSS, che prevede la trasmissione di ogni bit secondo una sequenza ridondante di valori, spesso impiegata anche per l'invio e la ricezione di segnali deboli. Questo processo, come prevedibile, porta sicuramente a una maggiore occupazione della banda disponibile, ma permette di ottenere uno spettro di segnale con potenza molto piú bassa; a seconda dell'hardware che compone la rete e dell'insieme di simboli da processare per ogni periodo, lo standard prevede l'utilizzo di una particolare modulazione: Binary Phase Shift Keying, Offset Quadrature Phase Shift e Parallel Sequenze Spread Spectrum.

Dal punto di vista dell'efficienza di comunicazione, lo standard prevede principalmente due tecniche per evitare le collisioni durante gli scambi dei dati. In presenza di una rete senza fili, non potendo utilizzare il CSMA/CD, si è optato per CSMA/CA, che prevede l'attivazione di un timer, nel caso in cui il buffer di comunicazione è occupato, e che viene decrementato solo durante i periodi di inattivitá del canale, sino a raggiungere il valore zero; in questo caso si ritenta la trasmissione del pacchetto. In aggiunta, vi è anche la possibilitá di impiegare la tecnica, nota col nome di Guarantee Time Slots, che consente di definire un nodo centralizzato che assegna time slot predefiniti a ogni altro nodo in modo che ognuno di questi sappia perfettamente quando trasmettere.

Il fatto di poter lavorare con ridotti duty cycles permette al trasmettitore di essere in standby per la maggior parte del tempo a seconda del modello di comunicazione impostato: questa modalitá fa risparmiare una quantitá enorme di energia ed è anche il fattore principale che ne ha motivato la diffusione nel campo industriale e domotico.

1.3 Le modalitá di funzionamento

ZigBee é in grado di offrire quattro diversi tipi di servizi di rete, che riguardano: la crittografia delle chiavi di rete e la sicurezza delle informazioni scambiate dai nodi; l'implementazione del protocollo di routing, denominato AODV, capace di ricercare dei particolari percorsi nella rete solo quando viene ricevuta una richiesta; l'associazione e l'autenticazione facilitate per ogni nuovo dispositivo connesso insieme anche all'introduzione dei cluster.

Utilizzando il concetto astratto di cluster, la rete viene come suddivisa virtualmente in piccole zone, aventi ciascuna delle determinate caratteristiche; ad esempio, se si vuole comandare a distanza una finestra, si puó definire un cluster con un'etichetta del tipo gestione finestre e aggiungerci i nodi interessati, che saranno quindi predisposti a compiere principalmente due funzioni: l'apertura di una singola finestra e la sua chiusura.

ZigBee permette di organizzare la rete e ottimizzarne il controllo. Ogni volta che si aggiunge un nuovo nodo, questo deve richiedere inizialmente al dispositivo coordinatore un indirizzo di rete valido, che verrá poi impiegato per l'invio e lo scambio dei dati al posto del MAC address. Un nodo é in grado di inviare informazioni agli altri nodi della rete solo facendo passare i propri pacchetti attraverso dei router predisposti, che sono sempre in attesa di ricevere chiamate; quando uno di questi riceve il pacchetto, ne rileva la destinazione e verifica se il nodo é attivo o in standby, procedendo poi all'inoltro delle informazioni, o all'avvio di un ciclo di attesa, rispettivamente.

Tipicamente, una rete ZigBee é composta da: un coordinatore, che risulta essere il dispositivo master, il quale detiene il controllo di tutte le informazioni; da vari router che hanno il compito di instradare correttamente i pacchetti; e dai sensori o nodi, che posti alle estremitá periferiche della rete, si preoccupano di agire sull'ambiente circostante o rilevarne dei dati utili.

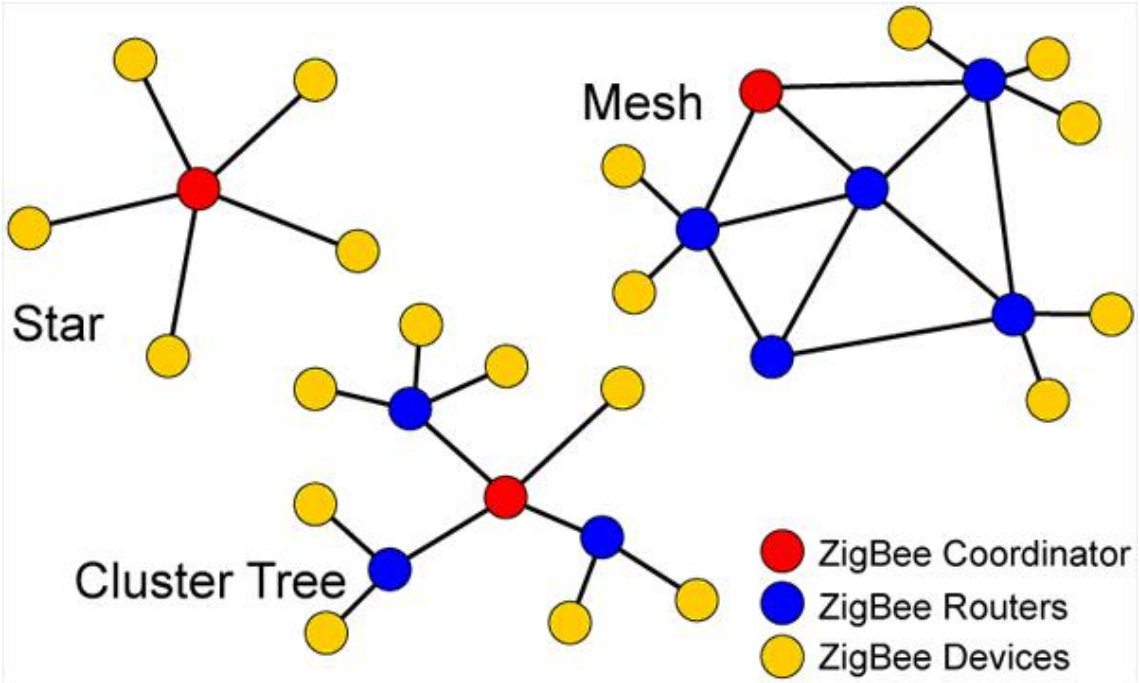


Figura 1.1: Zigbee Network
(Immagine fornita da http://www.meshnetics.com)

1.4 Il protocollo 802.15.4 e le differenze con Zigbee

Da un punto di vista prettamente funzionale, all'interno di una rete ZigBee, con topologia a stella, é possibile avere dei nodi connessi direttamente a un router o al coordinatore, cosí come i vari router esistenti possono connettersi tra di loro, creando una sottorete, o con il coordinatore; un aspetto importante da tenere in conto nel calcolo dei consumi energetici é che, mentre i sensori possono essere mandati in standby e alimentati a batterie, i router e il coordinatore devono essere alimentati regolarmente, oltre a essere sempre attivi perché hanno il compito di salvare le informazioni da inviare ai nodi nel proprio buffer, aspettando che si verifichi la possibilitá dell'effettivo inoltro.

Benché ZigBee sia fortemente basato sul protocollo 802.15.4, pensato per ottenere delle comunicazioni piú efficienti sia dal punto di vista del consumo

energetico che dell'instradamento dei pacchetti, esso introduce nuovi servizi extra e crea delle reti semicentralizzate, dove solo i sensori che fanno da nodi hanno la possibilitá di entrare in modalitá di standby. Inoltre, con 802.15.4 ogni nodo é in grado di comunicare direttamente con gli altri se questi sono raggiungibili, mentre con ZigBee é richiesta sempre la presenza di un router, o l'invio di una richiesta specifica al coordinatore di rete.

Non di meno, come segnalato anche prima, utilizzando ZigBee, le batterie possono essere usate solo con i dispositivi che si comportano da nodi, mentre con il protocollo 802.15.4 é possibile utilizzare le batterie come unica fonte di energia per tutti i dispositivi, in quanto viene implementato un algoritmo di sincronizzazione di tipo asincrono. Date le differenti tipologie di comunicazione, non é possibile integrare una rete nell'altra, ed é quindi essenziale decidere sin dall'inizio quale delle due impiegare, a seconda delle necessitá e delle situazioni progettuali.

1.5 I livelli dello stack Zigbee

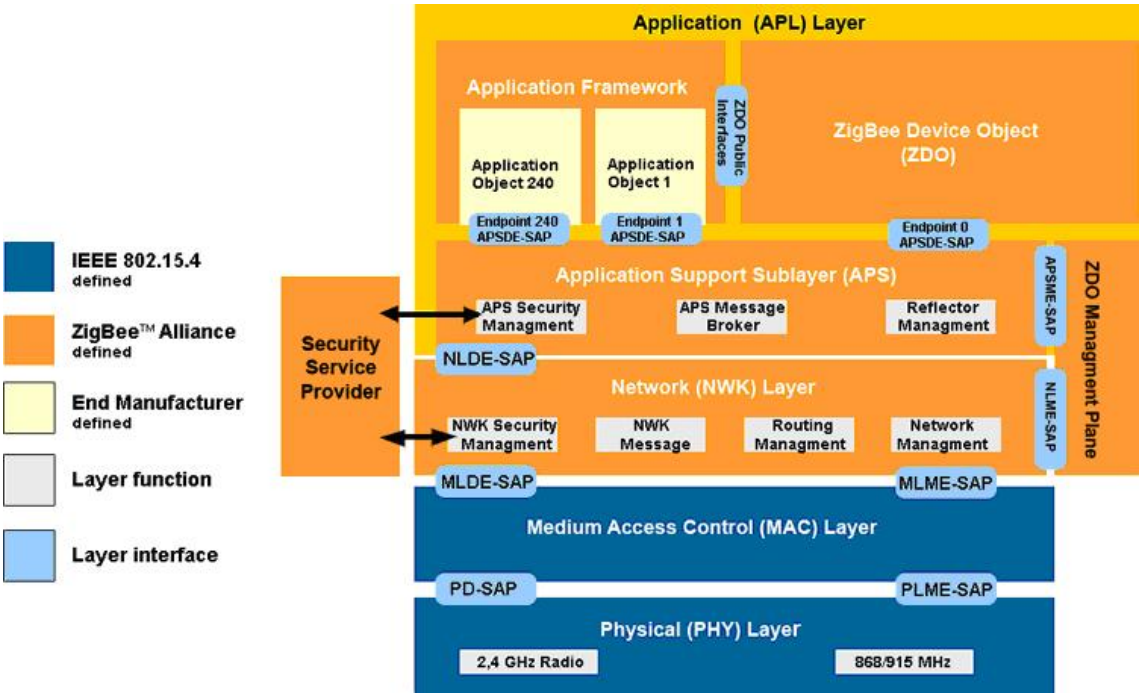


Figura 1.2: I livelli dello standard Zigbee (Immagine fornita da http://www.meshnetics.com)

ZigBee è basato su architettura stack che somiglia allo standard del modello OSI a sette strati, ma si limita a ridefinire quei livelli pertinenti alla realizzazione delle funzionalitá previste per l'implementazione di una rete basata sul proprio standard.

1.5.1 Livello Fisico (PHY)

Alla base della pila del protocollo ZigBee, si trova il layer fisico (PHY), il quale fornisce due differenti servizi: quello dati e quello gestione. Le caratteristiche principali implementate a questo livello sono l'attivazione e la disattivazione del trasmettitore radio, il rilevamento dell'energia (ED), l'indicazione della qualitá del collegamento (LQI), la selezione del canale, la stima della disponibilitá del canale (CCA) e la trasmissione e ricezione dei pacchetti sul mezzo fisico (canale radio). Lo standard offre due possibilitá per la trasmissione/ricezione basate su differente frequenze. Entrambi i metodi sfruttano la tecnica di modulazione di DSSS (Direct Sequenze Spread Spectrum). La velocitá di trasmissione è di 250kbit/s a 2.4GHz, 40kbit/s a 915MHz e 20kbit/s a 868MHz. Il tasso di trasmissione dati maggiore è da attribuirsi all'ordine dello schema di modulazione superiore. Frequenze piú basse assicurano, invece, una range ed una sensibilitá maggiore grazie alla minore attenuazione. Frequenze piú alte garantiscono una velocitá maggiore ed una latenza e duty-cycle minore. Nel range di frequenza tra 868-868.3MHz esiste un solo canale, mentre ci sono 10 canali tra i 902.0MHz e 928.0MHz, infine tra le frequenze 2.4GHz e 2.4835GHz si trovano ben 16 canali.

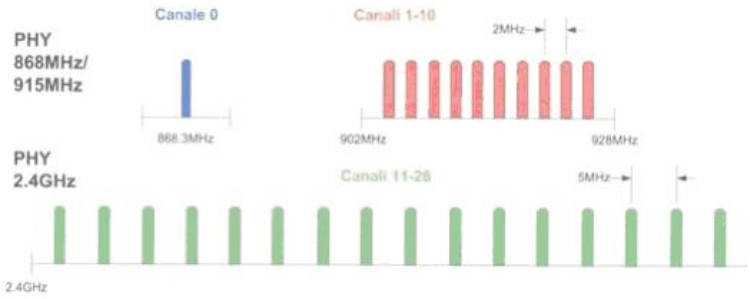


Figura 1.3: Livello fisico Zigbee
(Immagine fornita da http://www.meshnetics.com)

Ciascun simbolo consiste, dopo tale processo, di 32 bit detti chips e questo comporta un notevole aumento nella frequenza del segnale, spalmando lo spettro del segnale iniziale su un ampio range di frequenza (concetto di modulazione DSSS). Oltre alle operazioni brevemente descritte, un dispositivo che implementa il PHY deve essere capace di svolgere altri compiti, tra i quali il rivelamento dell'energia del ricevitore (ED). Si tratta di una caratteristica utilizzata dal layer superiore network (rete) all'interno dell'algoritmo di selezione del canale. In pratica, l'ED rappresenta una stima del segnale ricevuto e nessun tentativo viene fatto per decodificare i bit ricevuti. Il tempo di ED è pari alla durata di 8 simboli e il risultato viene riportato in una variabile intera di dimensione 8 bit (da 0x00 a 0xFF). Un'altra indicazione essenziale effettuata a questo livello è la qualitá del segnale (LQI). Essa puó essere effettuata utilizzando il parametro ED, una stima del rapporto segnale/rumore oppure una combinazione di entrambi. Tale valore è utilizzato nei layers superiori di rete o di applicazione e viene indicato come un valore intero ad 8 bit. Infine è necessario ricordare un'altra funzione svolta dal PHY, ossia la stima della disponibilitá del canale (CCA), indispensabile per implementare gli algoritmi di CSMA-CA per la gestione delle collisioni. Prima di poter avviare una trasmissione, un dispositivo ZigBee deve accertarsi se un altro sta utilizzando il mezzo radio in quel momento. Esistono tre possibili soluzioni per implementare tale funzione:

- energia oltre la soglia: il CCA riporta l'indicazione di mezzo occupato se il livello di energia ricevuto (ED) supera una prestabilita soglia
- rilevamento della portante: il CCA riporta l'indicazione di mezzo occupato solo se rileva un segnale con le stesse caratteristiche fissate nel protocollo IEEE802.15.4. Non è importante se tale segnale supera oppure no la soglia di energia
- rilevamento della portante con superamento della soglia: si tratta dei due metodi combinati insieme; viene riportata l'indicazione di mezzo occupato se viene rilevata la portante e la sua energia supera la soglia

1.5.2 Livello MAC

Coordinamento dell'eccesso al mezzo da parte del tranceiver, creazione ed instradamento dei pacchetti, generazione e riconoscimento dell'indirizzo, verifica del numero di sequenza dei pacchetti sono i principali compiti a cui é chiamato ad assolvere il livello MAC. Esso deve inoltre gestire il processo di rilevamento (Discovery) da parte di un dispositivo, di quelli ad esso vicini. Il tempo richiesto per far ció é dell'ordine di 30ms, mentre le tecnologie concorrenti, come Bluetooth, possono impiegare fino a 5-6s prima di poter iniziare ad utilizzare completamente di dispositivo. Le principali funzioni del livello MAC sono implementate in software a differenza di quanto avviene per il livello fisico (PHY layer) e scritte generalmente in linguaggio C. Esistono 4 possibili tipi di frame a livello MAC:

- Frame di dati
- Frame ACK
- Frame di comando MAC
- Frame di beacon

Il frame di dati è costituito al massimo da 128 bytes; esso è numerato per assicurare l'instradamento di tutti i pacchetti. Il campo Frame Check Sequence assicura che tutti i pacchetti siano ricevuti senza errori. Questo migliora notevolmente l'affidabilitá in condizioni sfavorevoli di trasmissione.

Un altro frame molto importante è il frame ACK. Esso fornisce la conferma che il pacchetto inviato è stato ricevuto correttamente. Questa soluzione garantisce la consistenza dei dati, ma ovviamente aumenta la latenza.

Il frame di comando MAC fornisce un meccanismo per il controllo e configurazione remota dei nodi client.

Infine, il frame di beacon ha il compito di svegliare i dispositivi client, i quali sono in ascolto del loro indirizzo e vanno in modalitá sleep se non lo

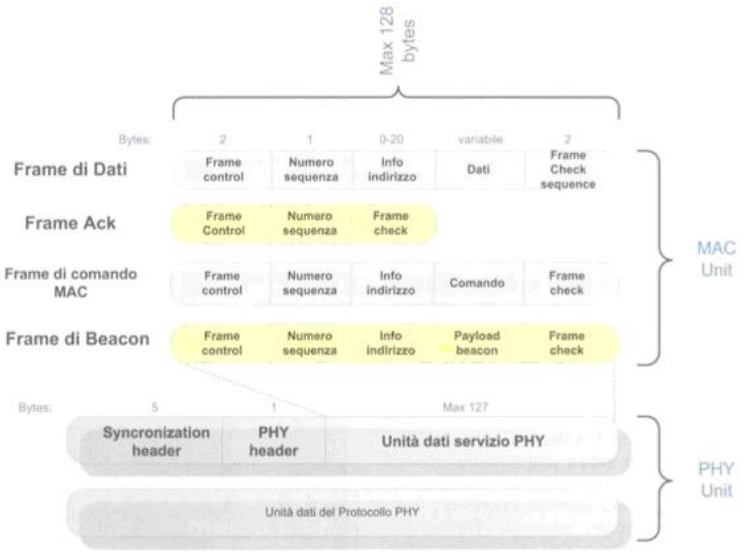


Figura 1.4: Livello Mac di Zigbee
(Immagine fornita da: http://www.zigbeenetwork.it)

ricevono. I beacon (che sono in pratica segnali di sincronismo) sono importanti per le reti a maglia e cluster-tree per mantenere tutti i nodi sincronizzati senza la necessitá che essi rimangano in ascolto per luoghi periodi di tempo, consumando cosí le batterie. Trattandosi di una trasmissione in cui il mezzo (radio) è condiviso da tutti i dispositivi, è necessario disporre di qualche metodo di arbitraggio della trasmissione, affinché due dispositivi non inviiono pacchetti contemporaneamente. Esistono due tecniche utilizzate: la CSMA-CA ed il beacon. A differenza di quanto avviene nelle reti LAN su cavo (IEEE802.3), per le reti wireless è stata adottata la tecnica di accesso multiplo con rilevamento della portante ed eliminazione delle collisioni, CSMA-CA (Carrier Sense Multiple Access with Collision Avoidance). In sostanza, significa che ogni dispositivo prima di iniziare una trasmissione deve ascoltare il mezzo e capire se è giá in corso una trasmissione. Se c'è giá un nodo che sta trasmettendo, allora sará effettuata la ritrasmissione successivamente con un ritardo casuale. La CSMA-CA viene adottata nelle reti ZigBee semplici di tipo peer-to-peer come per esempio sistemi di sicurezza in cui il dispositivo

è in modalitá sleep per il 99,99% del tempo. La seconda tecnica consiste nell'invio dalla parte del coordinatore di un superframe (modalitá beacon) ad intervalli regolari di tempo (multipli di 15.38ms, fino a 252s). Tutti i dispositivi si contendono i primi 9 time slot, mentre gli ultimi slot temporali sono invece assegnati dal coordinatore ad un nodo specifico e sono detti GTS (Guaranteed Time Slot). Nel caso un nodo debba trasmettere una grande quantitá di informazione, il coordinatore puó assegnarli anche piú di un GTS. Tale struttura garantisce una banda dedicata ed una bassa latenza rispetto alla prima tecnica. Inoltre consente di ridurre notevolmente il consumo delle batterie, poiché ciascun dispositivo sa esattamente quando trasmettere ed è sicuro che non ci saranno collisioni.

1.5.3 Livello Network

Il layer network (NWK) ha il compito di associare e dissociare i dispositivi al coordinatore, implementare la sicurezza ed instradare i frame alla loro destinazione. Inoltre, l'NWK è responsabile della creazione di una nuova rete e dell'assegnazione di un indirizzo ai nuovi dispositivi associati. Dal punto di vista del livello rete, i dispositivi ZigBee possono suddividersi in:

- Coordinatore (ZC): è unico per ogni rete; ha il compito di formare la rete ed agisce come router una volta che la rete si è costituita
- Router (ZR): si tratta di un componente opzionale e si puó associare con lo ZC oppure con altri ZR; ha il compito di instradare i messaggi
- EndDevice (ZED): si tratta di un componente opzionale e non partecipa al routing dei messaggi

Il procedimento con qui vengono stradati i pacchetti attraverso i vari router è basato su una tabella di routing contenuta all'interno dei ZR. Quando arriva un pacchetto, viene estratto l'indirizzo di destinazione e se presente all'interno della tabella di routing, allora si effettua il prelievo dell'indirizzo successivo. Come si nota in figura 1.5, la formazione di una nuova rete da parte di un coordinatore è originata con una richiesta dal layer applicazione ed è poi gestita a livello network e MAC. In basso in figura 1.6, si mostra invece la richiesta di un nodo ZigBee di legarsi ad una rete.

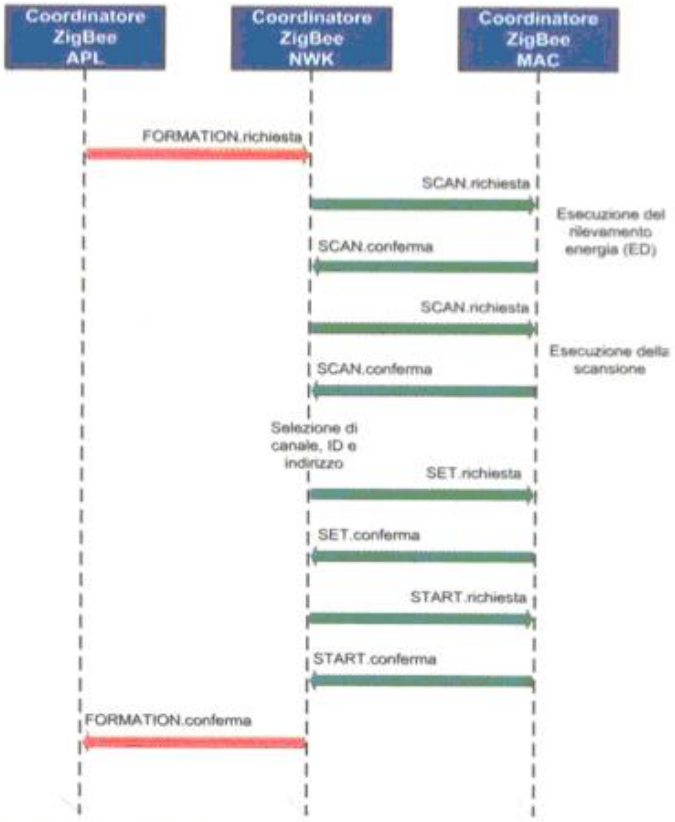


Figura 1.5: Formazione di una rete da parte del Coordinatore

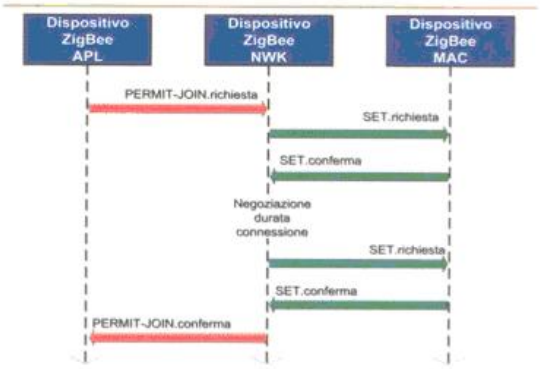


Figura 1.6: Richiesta di unione da parte di un nodo

1.5.4 Livello Applicazione

Il layer applicazione è costituito dai driver e dal codice, contenuti nella ROM del microcontrollore. Schematicamente di un nodo ZigBee si possono evidenziare, oltre al blocco relativo all'alimentazione, anche quelli inerenti il transceiver, l'antenna, il microcontrollre e l'interfaccia utente (rappresentata da Input/Output). Il transceiver implementa il layer fisico, ossia si occupa della modulazione del segnale come descritto in precedenza. All'interno della ROM del microcontrollore è presente l'implementazione del livello MAC, NWK e applicazione.

1.6 Sicurezza ed affidabilitá

Uno dei principali vantaggi di ZigBee rispetto ad altre tecnologie wireless di prossimitá, è l'elevato livello di sicurezza che viene supportata a livello di collegamento fra 2 nodi di rete, ma anche a livello rete ed applicativo. I servizi di sicurezza forniti da ZigBee includono meccanismi e protocolli per la generazione ed il trasporto sicuro delle chiavi, per la protezione dei frame e per la gestione dei dispositivi. In particolare la protezione dei dati è garantita da algoritmi di crittografia avanzati (AES a 128bit) e da meccanismi di integritá e di autenticazione per la protezione da eventuali attacchi provenienti da dispositivi non autorizzati che tentano di accedere alla rete o al contenuto informativo trasmesso. E' definito anche un concetto di Trust Center per la gestione centralizzata della sicurezza, a livello di politiche e di aggiornamento delle chiavi. Secondo lo standard, un nodo ZigBee puó operare sia in modalitá sicura che non sicura. Ovviamente, non implementando la sicurezza dei dati si ottiene un codice piú leggero.

Sono previsti 4 differenti servizi di sicurezza:

- Controllo degli accessi. Ogni dispositivo deve mantenere una lista di tutti i potenziali trasmettitori. In questo modo un dispositivo non autorizzato non puó accedere ad una rete ZigBee
- Codifica dei dati. I dati non sono trasmessi in chiaro, ma codificati mediante una chiave di crittografia posseduta solo dai componenti di rete
- Rinnovo sequenziale. Ogni frame viene confrontato con il precedente per evitare che ci siano ripetizioni
- Integrita dei frame. Sui bit di tutto il frame viene calcolato un check, tramite il quale è possibile risalire a modifiche del frame da parte di nodi non autorizzati

1.7 I profili

Un profilo è un approccio di gestione dei messaggi fra le applicazioni per dispositivi differenti. Un profilo descrive le componenti logiche e le loro interfacce. I profili sono sviluppati per far fronte alle differenti esigenze tecnologiche dei diversi produttori di dispositivi conformi al protocollo. In altre parole i profili sono un mezzo per unificare le diverse soluzioni tecniche in modo da renderle interoperabili all'interno dello standard ZigBee. Per esempio, si prevede che i fornitori di apparecchi di illuminazione vorranno fornire profili ZigBee che interagiscono con diverse varietá di tipi di illuminazione o tipi di controller. L' Application Profile (o profilo zigbee) non è un'entitá o un qualche tipo di software, bensí un accordo sui messaggi, i formati di messaggio e le azioni che consentono alle applicazioni che risiedono su dispositivi separati di inviare comandi e dati ad un'applicazione interoperabile, distribuita.

1.8 Il profilo SMART ENERGY

Il profilo smart energy definisce vari tipi di dispositivi e le specifiche per ogni dispositivo. Tuttavia, la specifica non definisce come ogni dispositivo deve essere implementato . Ci sono molte funzioni opzionali che possono essere fornite dai diversi produttori, ció aiuta a differenziare i propri dispositivi dagli altri, il profilo permetterá dunque di farli interfacciare tra loro Per esempio:

- Un semplice contatore elettrico prodotto da un determinato costruttore potrebbe registrare il consumo di energia e comunicarlo ad un altro contatore che ne determina il consumo in termini economici oppure tenere traccia dei diversi valori energetici periodicamente
- Un IPD(In premise Display) fornito da un certo produttore potrebbe mostrare testualmente dati relativi al consumo energetico o di acqua,mentre un altro poggiandosi ad esso potrebbe mostrare nel display un grafico invece che testo

1.8.1 I dispositivi legati al profilo Smart-Energy

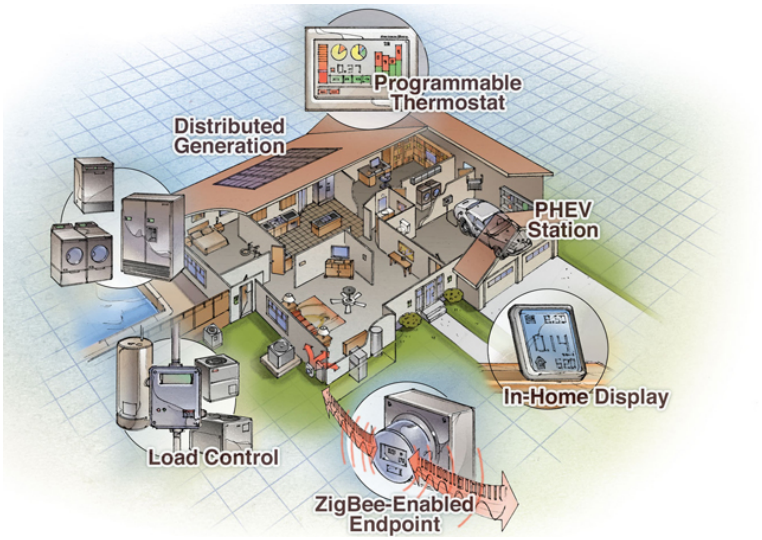
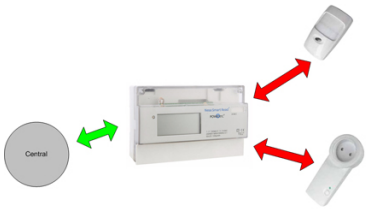


Figura 1.7: Dispositivi Smart-Energy
(Immagine fornita dal sito: http://www.telegesis.com)

Dispositivo di misurazione



Un dispositivo di misurazione misura elettricitá,gas o acqua. Dipendentemente da cosa viene misurato, il dispositivo è capace di leggere immediatamente oppure leggere e inviare peridiodicamente delle informazioni. Un dispositivo di misurazione è altrettanto capace di misurare lo stato delle batterie di altri dispositivi, della temperatura o dell'umiditá dell'ambiente.

Dispositivo In-Premise Display

Un dispositivo IPD trasmette il consumo di energia in maniera grafica o testuale. Il display puó anche non essere un dispositivo interattivo, limitandosi a visualizzare i dati solamente. Questo dispositvo puó mostrare l'uso corrente dell'energia, uno storico dell'energia usata in certi periodi, i consumi in termini economici o semplici messaggi di testo. Come dispositvo interattivo esso puó essere usato per mostrare un semplice messaggio, ad esempio:è stato premuto il bottone OK. Infine il display è utile per avvisare l'utente di limiti di consumi energetici o sprechi.

Termostato programmabile

Un termostato programmabile fornisce la possibilitá di monitorare e gestire il riscaldamento e il raffreddamento dei locali.

Load Control Device

Questo dispositivo puó ricevere comandi sulla gestione di un set di altri dispositivi, ad esempio riscaldamenti dell'acqua, gestori di luce e pompe per piscina.

Elettrodomestici intelligenti

Gli elettrodomestici intelligenti sono elettrodomestici conformi allo standard Zigbee che possono partecipare ad una corretta gestione dell'energia. Informano l'utente del consumo energetico o dello stato corrente. Questi dispositivi possono essere ad esempio:

- Lavastoviglie che scelgono di usare l'acqua fredda quando c'è molto consumo energetico
- Lavatrici che riportano lo stato
- Freezer e frighi che informano l'utente sulla loro temperatura

1.9 Telegesis Etrx2usb

Il disposistivo USB Etrx2usb di Telegesis é un ricetrasmettitore da 2.4GHz ISM band a rendimento elevato e a basso consumo. Esso comunica attraverso la porta seriale del proprio Pc o in questo caso specifico con il FritzBox. Nel riquadro sottostante sono mostrate le caratteristiche principali di questo dispositivo:

- ETRX2USB connette attraverso l'interfaccia USB2.0 un bridge seriale connesso a un modulo ETRX2.
- Alimenazione via USB
- Dispositivo a basso consumo
- Compatibile sia con USB 2.0 che con USB 1.1
- Dimensioni: USB Stick L80mm x W26mm x H8.5mm.
- Possibilitá di sviluppare un proprio firmware e caricarlo nel dispositivo
- Temperature d'utilizzo: min: -40 gradi , max: +85 gradi.
- Non c'è bisogno di avere delle conoscenze su RF
- 2.4GHz ISM Band digital
- Accelerazione hardware per le operazioni IEEE 802.15.4

- Basato su ETRX2 Ember EM250 single chip ZigBee / 802.15.4 solution
- Hardware supported encryption (AES-128).
- Comandi AT pre-programmati
- Puó essere configurato come Coordinatore, router oppure end device di una rete Zigbee
- E' ideale per instaurare e inizializzare una nuova rete Zigbee

1.9.1 Il Firmware

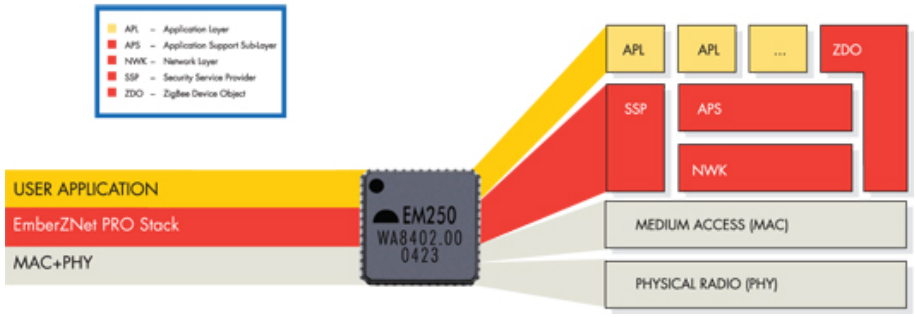


Figura 1.8: EM250 Firmware
(Immagine fornita dal sito: http://www.ember.com)

Esso possiede un firmware EM250 sviluppato da Ember e conforme con lo standard IEEE 802.15.4. Il firmware fornisce la possibilitá di operare attraverso il protocollo Zigbee e di interfacciarsi con sensori e dispositivi conformi allo stesso protocollo. In particolare la chiavetta USB è in grado di comunicare con i dispositivi che implementano il profilo Smart Energy; attualmente il firmware residente nel dispositivo implementa i comandi AT in grado di comunicare con sensori di luce, gas, temperatura, umidità con estrema facilitá e di informare il consumatore sul consumo di energia sia graficamente che testualmente fornendo inoltre modalitá d'uso che possono garantire il risparmio e la corretta gestione dell'energia. Usando il set di comandi dell'ETRX2USB i progettisti possono rapidamente sviluppare ed integrare il profilo smart energy grazie a piú di 30 comandi che per semplificare la comunicazione con il modulo presente all'interno del dispositivo sono simili ai comandi di controllo per i modem che utilizzano lo standard Hayes.

1.9.2 I comandi AT

Lo standard Hayes definisce un formato comune dei comandi AT dove ogni comando deve essere preceduto dal prefisso AT e terminato con il carattere di escape: carriage return. Ogni comando che non utilizza questo formato non verrá accettato dal dispositivo e di conseguenza riporterá un messaggio di errore. Se il comando avrá esito positivo, il dispositivo risponderá con l'output di quest'ultimo seguito da un OK, altrimenti verrá riportato un messaggio di errore. Settando alcuni registri interni al dispositivo è possibile nascondere o mostrare alcuni messaggi di prompt descritti nella tabella 1.1 .

Command	Description
OK	OK terminator
ERROR:XX	Error number XX occurred
ACK:XX	Acknowledgement for message XX was received
NACK:XX	Acknowledgement for message no XX was not received
SR:XX,<EUI>,<NodeID>	Route Record Message received
BCAST:[<EUI64>,]XX=<data>	A Broadcast with XX characters has been received
MCAST:[<EUI64>,]XX=<data>	A Multicast with XX characters has been received
UCAST:[<EUI64>,]XX=<data>	A Unicast with XX characters has been received
COO:<EUI64>,<NodeID>	A coordinator announcing itself
FFD:<EUI64>,<NodeID>	A router announcing itself
SED:<EUI64>,<NodeID>	A sleepy end device announcing itself
MED:<EUI64>,<NodeID>	A mobile sleepy end device announcing itself
ZED:<EUI64>,<NodeID>	An end device announcing itself
NEWNODE: <NodeID>, <EUI64>, <Parent NodeID>	Shown on Coordinator: New node has joined the PAN
LeftPAN	Local Node has left the PAN
LostPAN	End Device has lost contact with Parent
JPAN:<channel>,<PID>,<EPID>	Local Node has joined PAN with given parameters
SINK:<EUI64>,<NodeID>	Selected new Sink
ADSK:<EUI64>,<NodeID>	Received Sink Advertisement
SREAD:<NodeID>, <EUI64>,<Register>, <errorcode>[=<Data>]	Reply to a remote S Register Read operation
SWRITE:<NodeID>, <EUI64>, <errorcode>	Reply to a remote S Register Write operation
DataMODE:<NodeID>, <EUI64>	Datamode has been opened remotely
DataMODE:<NodeID>, <EUI64>, <errorcode>	Response to an attempt to open data mode
OPEN	Data mode is open
CLOSED	Data mode is closed
PWRCHANGE:XXXX	Local node has changed Power Mode to XXXX

Tabella 1.1: Messaggi di prompt (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

I comandi AT possono leggere un dato proveniente dal dispositivo o mandare dei dati per gestire la rete Zigbee o ancora, per scrivere nei suoi registri

interni o di quelli remoti dei dispositivi connessi. Nel riquadro sottostante sono riportare le operazioni di lettura/scrittura:

Command	Description
ATXXX?	I comandi che terminano con ? ritornano il valore di un determinato registro
ATXXX=<...>	Questo comando è utilizzato per settare una variabile con un determinato valore
ATXXX	Questo comando è usando per eseguire un'istruzione e ritorna dei parametri

L'esecuzione di un comando potrebbe richiedere dei parametri, questi peró possono essere utilizzati nella scrittura di un'istruzione. Ad esempio se volessi avere le informazioni su un dispositivo remoto, dovró passare al comando un identificativo che mi permetta di identificare questo nella rete. I parametri sono diversi,qui di seguito viene riportata la descrizione di ciascun parametro:

Command	Description
XX	8-bit hexadecimal number. Valid characters are 0-9, a-f and A-F
XXXX	16-bit hexadecimal number. Valid characters are 0-9, a-f and A-F
n	Number from 0-9
s	Sign
b	Bit (0 or 1)
c	character
<PID>	16-bit hexadecimal PAN ID (0000 to FFFF)
<EPID>	64-bit hexadecimal extended PAN ID
<channel>	decimal channel (802.15.4 channel 11-26)
<password>	8 character password
<EUI64>	64-bit IEEE 802.15.4 address in hexadecimal
<ioread>	32-bit hexadecimal number representing the reading of S1A
<data>	Custom Data
<ClusterList>	A list of 16 bit cluster identifiers in hexadecimal representation

Tabella 1.2: Parametri (tabella fornita da:http://www.telegesis.com/downloads/general /TG-ETRXn-R302-Commands.pdf)

Di seguito sono descritti i comandi principali forniti dal dispositivo, bisogna ricordare peró che altri comandi possono essere aggiunti o comunque riscritti dato che il firmware puó essere personalizzato:

Command	Description
ATI	Display Product Identification Information
ATZ	Software Reset
ATF	Restore Factory Defaults
AT+BLOAD	Enter The Bootloader Menu
AT+CLONE	Clone Local Node To Remote Node
AT+RECOVER	Recover From A Failed Clone Attempt
ATS	S-Register Access
ATREMS	Remote S-Register Access
ATSALL	Remote S-Register Access
AT+TOKDUMP	Display All S-Registers

Tabella 1.3: Module control and configuration (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

Command	Description
AT+ESCAN	Scan The Energy Of All Channels
AT+PANSCAN	Scan For Active Pans
AT+EN	Establish Personal Area Network
AT+JN	Join Network
AT+JPAN	Join Specific Pan

Tabella 1.4: Network control and configuration (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

Command	Description
AT+DASSL	Disassociate Local Device From Pan
AT+DASSR	Disassociate Remote Node From PAN (ZDO)
AT+N	Display Network Information
AT+NTABLE	Display Neighbour Table (ZDO)
AT+IDREQ	Request Node's NodeID (ZDO)
AT+EUIREQ	Request Node's EUI (ZDO)
AT+NODEDESC	Request Node's Descriptor (ZDO)
AT+POWERDESC	Request Node's Power Descriptor (ZDO)
AT+ACTEPDESC	Request Node's Active Endpoint List (ZDO)
AT+SIMPLEDESC	Request Endpoint's Simple Descriptor (ZDO)
AT+MATCHREQ	Find Nodes which Match a Specific Descriptor (ZDO)
AT+ANNCE	Announce Local Device in the Network (ZDO)
AT+SR	Set Source Route To Remote Device
AT+FNDSR	Find The Source Route To A Remote Device
AT+POLL	Poll For Data From Parent
AT+REJOIN	Rejoin The Network
AT+SN	Scan Network
AT+KEYUPD	Update the Network Key (ZDO)
AT+BECOMETC	Make Local Device the Trust Centre

Tabella 1.5: Command Overview (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

Command	Description
AT+ATABLE	Display Address Table
AT+ASET	Set Address Table Entry
AT+MTABLE	Display Multicast Table
AT+MSET	Set Multicast Table Entry
AT+BCAST	Transmit A Broadcast
AT+BCASTB	Transmit A Broadcast Of Binary Data
AT+UCAST	Transmit A Unicast
AT+UCASTB	Transmit A Unicast Of Binary Data
AT+SCAST	Transmit Data To The Sink
AT+SCASTB	Transmit Binary Data To The Sink
AT+SSINK	Search For A Sink
AT+MCAST	Transmit A Multicast
AT+MCASTB	Transmit A Multicast Of Binary Data
AT+DMODE	Enter Data Mode (Serial Link Mode)
+++	Leave Data Mode

Tabella 1.6: Messaging 1 (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

1.9.3 I registri

Il dispositivo possiede dei registri interni che possono essere modificati oppure semplicemente letti per modificare o leggere le impostazioni della rete o del modulo stesso. Quasi tutti i registri sono memorizzati all'interno

Command	Description
AT+IDENT	Play A Tune On Remote Devboard
AT+RDATAB	Send Binary Raw Data
AT+FINDMTR	Finds A Meter/ESP On The HAN
AT+ATTACHMTR	Pairs The IHD To A Meter/ESP
AT+DETACHMTR	Un Pairs The IHD From A Meter/ESP
AT+MTRATR	Gets The Requested Attribute From The Metering ClusterServer
AT+MTRPROFILE	Gets The Energy Consumption For Profiling Purposes
AT+PRICELBL	Gets The Label Assigned To The Price Tier 1 To 6 From The Price Cluster Server
AT+CURPRICE	Gets The Current Pricing From The ESP
AT+SCHPRICE	Gets All The Scheduled Pricing From The ESP
AT+LASTMSG	Gets The Last Message From The ESP
AT+ACKMSG	Used To Acknowledge A Message
AT+SETTIME	Set The Time On The IHD
AT+GETTIME	Get The Time On The IHD
AT+SYNCTIME	Sync The IHD's Clock With The ESP
AT+IDENTIFY	Identify The Meter/ESP On The HAN

Tabella 1.7: Messaging 2 (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

di una memoria non volatile e il loro valore puó essere mantenuto finchè non si effettui un reset delle impostazioni di fabbrica.

1.9.4 Consumo energetico

Nella tabella seguente vengono specificati i vari consumi energetici attribuiti a 4 modalitá diverse in cui puó essere configurato sia il dispositivo della Telegesis sia i sensori ad esso connessi:

MODE	MCU	Radio	Timers	I
0	Awake	Awake	User defined	36mA
1	Idle	Awake	User defined	32mA
2	Awake	Awake	User defined	0.7mA
3	Asleep	Asleep	Off	0.7mA

Tabella 1.8: Router, COO (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

MODE	MCU	Radio	Timers	I
0	Awake	Asleep	User defined	9mA
1	Idle	Asleep	User defined	4.5mA
2	Asleep	Asleep	User defined	0.7mA
3	Asleep	Asleep	Off	0.7mA

Tabella 1.9: Mobile End Device, Sleep End Device (tabella fornita da: http://www.telegesis.com/downloads/general/TG-ETRXn-R302-Commands.pdf)

1.9.5 Personalizzazione del firmware

Il dispositivo ETRX2USB offre la possibilitá di personalizzare il proprio firmware attraverso un dispositivo prodotto dalla EMBER chiamato inSight Adapter. Questo dispositivo viene connesso direttamente alla chiavetta offrendo la possibilitá di debuggare e programmare in real time.

Figura 1.9: Ember Insight Adapter (immagine fornita da: http://www.ember.com)

Capitolo 2

Fritz!Box - Un router multifunzionale

2.1 Cos'è

Il dispositivo in oggetto è la versione 7270 della linea Fritz!Box, una serie di apparati SoHo dalle caratteristiche avanzate, pensati e creati da AVM, uno dei maggiori produttori di dispositivi DSL oriented in Europa. Il Fritz!Box Fon WLAN 7270 è una all-in-one solution in grado di gestire interamente ogni esigenza di connessione Internet e di telefonia, in grado di sostituire egregiamente fino a 8 dispositivi (telefono, fax, centralino, print server, ecc.) di comunicazione individuale, con un grande risparmio in termini energetici. Dispone di un centralino integrato con cinque segreterie telefoniche, di porte personalizzabili per collegare telefoni e fax. Il centralino intelligente integrato permette di realizzare telefonate sfruttando, in base alle esigenze, la connessione ADSL (telefonia VoIP), la linea ISDN o quella analogica.

La segreteria telefonica consente di memorizzare i messaggi vocali su una memoria USB o anche di inviarli direttamente al proprio account di posta elettronica, cosí come anche il fax integrato puó inviare i documenti ricevuti sulla propria e-mail o salvarli su un drive USB. Grazie alle molteplici interfacce è possibile connettere con la massima semplicitá i telefoni giá presenti in casa o in ufficio, analogici, cordless o ISDN. La compatibilitá con telefoni DECT è molto alta e alla base si possono collegare contemporaneamente fino a sei dispositivi portatili Inoltre, la porta USB host consente di utilizzare il Fritz!Box come print server o per condividere memorie o dischi USB all'interno della rete locale o anche tramite Internet.

Per venire incontro alle tematiche ambientali, a cui tutti dobbiamo prestare sempre piú attenzione anche all'interno delle nostre case, Fritz!Box Fon Wlan 7270 si adegua perfettamente alle ultime normative della Unione Europea in tema di risparmio energetico. Grazie alla funzione Eco-mode, il dispositivo riduce il consumo di energia con diversi ingegnosi accorgimenti, come ad esempio la riduzione automatica dell'alimentazione quando il dispositivo è in stand-by, l'impostazione per lo spegnimento notturno o la limitazione del consumo sulle porte LAN inutilizzate, abbassando in questo modo considerevolmente l'assorbimento energetico fino all' 85%.

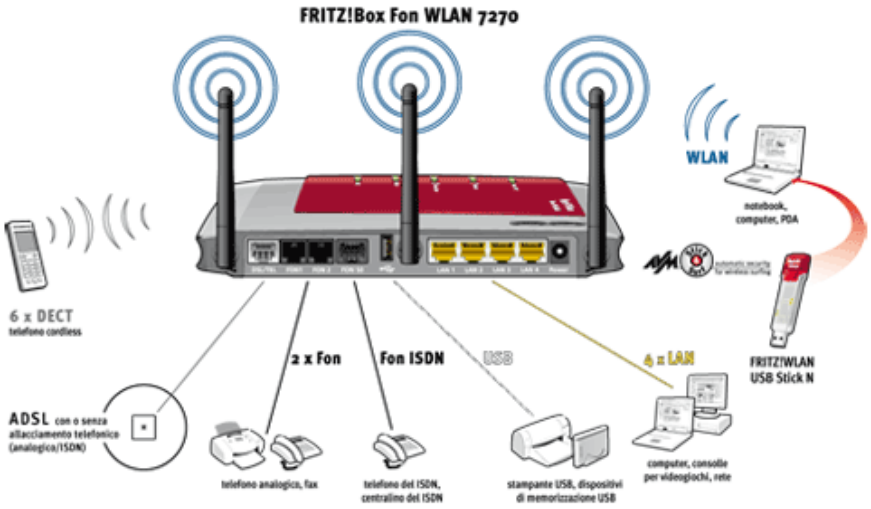


Figura 2.1: Retro del Fritz!Box (immagine fornita dal sito: http://www.avm.de)

Le funzionalitá del Fritz!Box non si limitano alla normale amministrazione ed offrono, per il comparto fonia in particolare, una serie di opzioni altamente personalizzabili supportate da un firmware sempre all'altezza.

Di seguito è proposta una panoramica sulle principali caratteristiche di questo router:

- router WLAN con server DHCP e IP masquerading/NAT (Network Address Translation)
- modem ADSL integrato con supporto ADSL e ADSL 2+
- PBX per telefonia via Internet e su linea fissa (linee ISDN e POTS)
- stazione base DECT integrata (GAP) per collegare fino a 6 telefoni cordless; predisposto per il nuovo standard DECT con migliore qualità voce (CAT-iq)
- posta vocale integrata

- reti wireless WLAN conformi a 802.11n (bozza 2.0, fino a 300 Mbit/s totali), WLAN 802.11g, b, e
- supporto WLAN per connessioni a 2,4 GHz e 5 GHz
- crittografia WLAN con modalitá mista WPA/WPA2, WPA2, WPA, WEP-64 o WEP-128
- modalità risparmio energetico WLAN
- porta USB 2.0 per dispositive quali stampanti e supporti di archiviazione per un utilizzo combinato nella rete; memoria disponibile in LAN, WLAN e Internet
- funzione FRITZ!Musikbox compatibile con ricevitori UPnP-AV
- firewall con ispezione pacchetti stateful e inoltro sicuro alle porte per i server locali
- dimensioni: 210 x 155 x 25 mm; installazione su un piano o a parete
- consumo medio 6-8 Watt

2.2 Le modalitá di accesso al Fritz!Box tramite FTP e Telnet

Il Fritz!Box consente l'accesso tramite tre tipi di interfaccia:

1. WEB
2. FTP
3. TELNET

La prima: WEB, cioè tramite interfaccia web all'indirizzo http://fritz.box oppure http://192.168.178.1. Grazie a questa interfaccia è possibile modificare loe impostazioni di connessione, le varie uscite telefoniche voip FXS (variano da 2 a 3 a seconda dei modelli), le impostazioni della LAN e tanto altro ancora.. la seconda: FTP, è una modalità di accesso al Fritz!Box considerata per esperti, infatti tramite questa modalità è possibile modificare le variabili di sistema e ripristinare il Fritz!Box nei casi di necessità più gravi. La terza: TELNET, è una modalità d'accesso più semplice da raggiungere, consente molte modifiche alle impostazione del Fritz!Box, molte delle quali possibili anche tramite FTP.

2.3 L'architettura del Fritz!Box

Il router possiede un architettura di tipo mipsel con chipset UR8 che integra un processore risc Mips 4Kec da 32 bit della famiglia dei processori 4k, C62x DSP (per DSL) e C55x DSP (per VoIP). La cpu e il c62x dsp operano a 360MHz mentre il C55x DSP a 180MHz. I processori 4Kec sono progettati per sistemi on chip (Soc) ossia per sistemi economici ed efficienti che in un solo chip contengono un intero sistema, o meglio, oltre al processore centrale, integrano anche un chipset ed eventualmente altri controller come quello per la memoria RAM, la circuiteria input/output o il sotto sistema video. Un singolo chip puó contenere componenti digitali, analogici e circuiti in radiofrequenza in un unico chipset integrato. Questa tipologia di integrati viene utilizzata comunemente nelle applicazioni embedded, date le dimensioni ridotte che essi raggiungono con l'integrazione di tutti i componenti.

2.4 La memoria Flash del Fritz!Box

La memoria flash é partizionata in 6 parti, come si può vedere dal file /proc/mtd:

Dev	Dimensione	Erase-size
mtd0	00300000	00010000
mtd1	000b0000	00010000
mtd2	00010000	00010000
mtd3	00020000	00010000
mtd4	00020000	00010000
mtd5	00023000	00010000

Queste partizioni vengono mappate direttamente nella directory virtuale /dev/mtdblock/: $ls /dev/mtdblock/ 0 1 2 3 4 6 in maniera tale da renderne

facile l'accesso e la modifica (ad esempio per aggiornare il firmware). Vediamo ora nel dettaglio a cosa servono le varie partizioni:

mtd0 La partizione mtd0 contiene la root di sistema ed é di tipo squashfs, un filesystem altamente compresso che permette accessi in sola lettura, ideale per essere utilizzato appunto con memorie flash su sistemi embedded.

mtd1 La partizione mtd1 contiene il kernel, compresso con algoritmo LZMA, che verrá utilizzato dal bootloader per avviare il sistema.

mtd2 La partizione mtd2 contiene il bootloader.

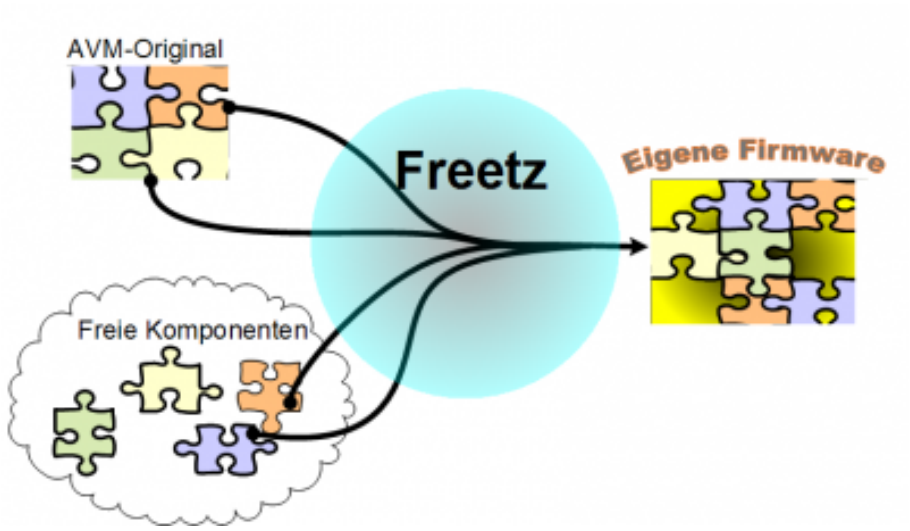
mtd3 e mtd4 Le partizioni mtd3 e mtd4 sono in formato tffs; contengono le variabili d'ambiente condivise tra il bootloader e il systema linux e la configurazione del bootloader. Tali partizioni vengono gestite in double buffering, in quanto si tratta di informazioni molto delicate e una corruzione di tale spazio puó causare un congelamento completo della procedura di boot, nel qual caso occorre intervenire con una JTAG per ripristinare le partizioni del bootloader.

mtd5 La partizione mtd5 é anch'essa di tipo squashfs; viene montata durante il processo di boot e contiene una serie di file necessari durante l'avvio.

2.5 Busybox e linux

Il Fritz!Box integra una particolare distribuzione di linux (versione kernel : 2.6.19.2)realizzata per far fronte alle forti limitazioni hardware che il dispositivo possiede. La shell di sistema è fornita da Busybox. Esso é un software libero, rilasciato sotto la GNU General Public License, che combina diverse applicazioni standard di Unix in un piccolo eseguibile. BusyBox puó fornire la maggior parte delle utility menzionate nel Single Unix Specification ed in aggiunta altre che un utente si aspetterebbe di vedere su un sistema GNU/Linux. Il programma viene solitamente utilizzato in sistemi Linux embedded, anche se comunque viene utilizzato anche in alcune distribuzioni Linux per lo Sharp Zaurus e il Nokia 770. Originariamente scritto da Bruce Perens, nel 1996, lo scopo di BusyBox era di mettere un sistema completo su un solo floppy che sarebbe stato sia un rescue disk sia un installer per la distribuzione Debian GNU/Linux. È divenuto poi uno standard de facto per i dispositivi Linux embedded e nelle installazione delle distribuzioni. Prima di questo ogni eseguibile Linux richiedeva diversi KB, ora invece con BusyBox, che combina più di duecento programmi insieme, viene utilizzato poco spazio. BusyBox é estremamente modulare, consente di includere o escludere i comandi (o funzioni) in fase di compilazione. Ció rende più semplice la personalizzazione dei sistemi embedded.

2.6 Freetz e le modifiche al firmware



Il firmware é il software che permette ad un apparecchio come il Fritz!Box di poter funzionare regolarmente, di svolgere le operazioni per cui esso é stato progettato, e ovviamente, di essere riconosciuto dalle periferiche o pc a cui viene collegato. Ogni modello di Fritz!Box ha una sua versione specifica del firmware che può essere modificato attraverso Freetz.

Freetz é un toolbox per gli sviluppatori e utenti esperti utilizzato per generare un firmware modificato sulla base del firmware originale per il Fritz!Box e di trasferirlo nel dispositivo in questione utilizzando la normale funzione di aggiornamento del firmware.Ci sono molti pacchetti di estensione disponibili, inoltre offre la possibilitá di rimuovere le funzionalitá indesiderate dal firmware originale per crearne uno nuovo che soddisfi le proprie esigenze. Il firmware é costituito da molti singoli componenti. Essi sono stati sviluppati da varie persone e organizzazioni. Freetz é in grado di rimuove i singoli componenti, cambiare la configurazione dei componenti esistenti o inserirne dei nuovi. Una gran parte dei componenti del firmware sono software Open Source. I loro autori hanno espressamente autorizzato il software per essere utilizzato da chiunque, modificato e distribuito. Altre parti del firmware sono

sviluppate da AVM. Questi ultimi componenti sono protetti e non accessibili al pubblico. Una funzionalitá puó essere:

AGGIUNTA Ció potrebbe essere, per esempio, un server web o VPN speciali. L'elenco parziale dei Pacchetti offre una panoramica di ció che é possibile installare:

- Apache + PHP
- Bash
- cifsmount
- httptunnel
- CTorrent
- CURL
- dropbear
- FUSE
- httptunnel
- lighttpd
- samba/nmbd
- usbutils
- webdav

CAMBIATA A volte, la funzionalitá esiste, ma nella sua forma originale non puó essere configurata o vi sono limiti a ció che può essere configurato. (Ad esempio il firewall integrato).

RIMOSSA Questo puó essere utile quando una particolare funzionalità non é necessaria o quando è necessario dello spazio supplementare per far posto a pacchetti aggiuntivi (lo spazio di memoria é piuttosto limitato).

2.6.1 Il processo di modifica e compilazione del firmware

Questa fase puó essere divisa in tre sottofasi:

Organizzazione e selezione dei pacchetti Il processo é basato su menu simile alla selezione dei pacchetti e dei moduli nel procedimento di compilazione del kernel linux. È possibile definire esattamente ció che verrá installato nel firmware e che caratteristiche avrá . Freetz gestisce le dipendenze tra i componenti ed assicura il corretto funzionamento del prodotto finito

Compilazione Questa parte richiede molto tempo, a seconda delle prestazioni del PC ma sará normale procedere senza richiedere alcuna interazione. In conformità con i componenti selezionati, i file di origine verranno scaricati da Internet. Alla fine, verrá generato un file contentente il firmware, quindi l'immagine del kernel e del filesystem.

Installazione del firmware Viene effettuata utilizzando il normale aggiornamento Firmware attraverso l'interfaccia web del Fritz!Box

Configurazione Si procede con la configurazione e l'utilizzo delle nuove funzionalitá del Fritz!Box

Capitolo 3

FritzBee

Avendo la possibilitá di collegare un dispositivo usb al Fritz!Box,si é pensato di trovare un modo per permettere al dispositivo Telegesis Etrx2usb di comunicare con il router attraverso una comunicazione di tipo seriale. Si é quindi studiata la fattibilitá attraverso la raccolta e lo studio di documentazioni che permettessero di assicurare un corretto funzionamento hardware. Questa sicurezza non veniva garantita in quanto, il funzionamento di questo dispositivo non é stato testato su una macchina che possedesse un architettura mipsel, ma solo con PC intel.

3.1 La porta seriale

I computer trasferiscono le informazioni un bit alla volta. Le comunicazioni seriali includono dispositivi quali tastiere, modem e terminali. Quando avviene una comunicazione seriale, ogni byte o carattere d'informazione viene spedito uno alla volta. Il trasferimento di questi bytes sono solitamente espressi in bps oppure in bauds. Questo numero rappresenta il numero di bit che vengono spediti al secondo. Lo standard utilizzato per la comunicazione seriale è L'RS-232, questa può essere di 3 tipi diversi(A, B, C) che si differenziano per i diversi voltaggi che posseggono. Questa interfaccia oltre a pin di

entrata e uscita ne possiede altri che determinano lo stato e l'handshaking. Di seguito ci sono i segnali che vengono gestiti dagli altri pin:

- GND - Logic Ground
- TXD - Transmitted Data
- RXD - Received Data
- DCD - Data Carrier Detect
- DTR - Data Terminal Ready
- CTS - Clear To Send
- RTS - Request To Send

Il control flow

Spesso è necessario gestire il flusso dei dati durante il loro trasferimento tra due interfacce seriali che non comunicano in modo sincrono. Due metodi sono comunemente utilizzati per i dati asincroni. Il primo metodo è spesso chiamato controllo di flusso software e utilizza caratteri speciali per avviare (o XON DC1, 021 ottale) o fermare (XOFF o DC3, 023 ottale) il flusso di dati. Questi caratteri sono definiti nello standard ASCII . Questo metodo è utile quando l'informazione è di tipo testuale. Il secondo metodo si chiama controllo di flusso hardware e utilizza la RS-232 CTS e RTS, invece dei caratteri speciali. Poiché il controllo di flusso hardware utilizza un altro insieme di segnali, è molto più veloce del controllo di flusso software che ha bisogno di inviare o ricevere più bit di informazioni per fare la stessa cosa.

3.2 Le UART

Le UART (Universal Asynchronous Receiver Transmitter) - ricevitore/-trasmettitore asincrono universale - sono chip seriali posti sulla scheda madre del PC oppure su una scheda modem interna. La funzione della UART puó anche essere svolta da un chip che fa anche altre cose. Sui computer piú vecchi come la maggior parte dei 486, i chip erano sulla scheda del controller I/O. Alcuni computer piú vecchi hanno schede seriali dedicate. Scopo della UART é convertire i byte dal bus parallelo del PC in un flusso seriale di bit. Il cavo che esce dalla porta seriale ha solo un cavo per ogni direzione di flusso. La porta seriale invia un flusso di bit, un bit alla volta. Al contrario, il flusso di bit che entra dalla porta seriale via cavo esterno viene convertito in byte paralleli che il computer puó ricevere. Le UART trattano dati divisi in porzioni della dimensione di un byte, che per convenienza é anche la dimensione dei caratteri ASCII. Oltre alla conversione tra seriale e parallelo, la UART esegue alcune altre cose come derivazione (effetto collaterale) dei suoi compiti primari. Il voltaggio usato per rappresentare i bit viene anch'esso convertito (cambiato). I bit extra (chiamati bit di start e stop) sono aggiunti ad ogni byte prima che venga trasmesso. Inoltre, mentre il rapporto di flusso (in byte per secondo) sul bus parallelo all'interno del computer é molto alto, il rapporto di flusso in uscita dalla UART dalla parte della porta seriale é molto piú basso. L'UART ha un elenco fisso di rapporti (velocitá) che puó usare come interfaccia per la sua porta seriale.

3.3 CP2101: Il bridge controller

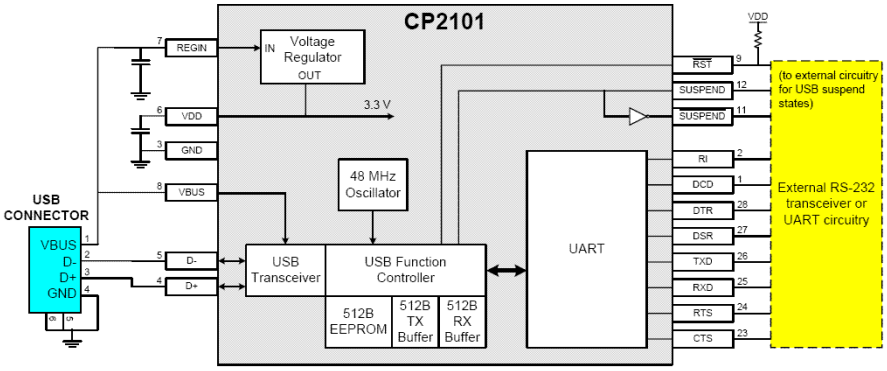


Figura 3.1: Il chip cp2101 (immagine fornita da http://www.silabs.com)

Il chip CP2101 USB-UART converte il traffico di dati tra USB e UART. Questo chip puó essere collegato a qualsiasi UART per fornire immediata connettivitá USB. Esso fornisce una soluzione semplice per l'aggiornamento di dispositivi che utilizzano l'interfaccia RS-232 utilizzando un numero minimo di componenti elettronici e implementando tutti i suoi segnali, compreso il controllo e la gestione dell' handshaking. I driver software sono forniti gratuitamente dalla Silicon Labs. In linux viene utilizzato il modulo cp2101.o oppure cp201x che rappresenta una versione aggiornata del driver.

3.4 I moduli del kernel

3.4.1 Cosa sono

I moduli sono porzioni di codice che possono essere caricati e scaricati nel kernel su richiesta. Si estendono le funzionalitá del kernel senza avere la necessitá di riavviare il sistema. Ad esempio, un tipo di modulo é il driver di periferica, che permette al kernel di accedere all' hardware collegato al sistema. Senza moduli, avremmo dovuto compilare un kernel monolitico e aggiungere nuove funzionalitá direttamente nel kernel.

Per riuscire ad instaurare una connessione tra il FritzBox! e la chiavetta telegesis é stato necessario caricare 2 moduli fondamentali:usbserial e cp2101 ed apportare una modifica ad uno di questi.

3.4.2 Il modulo usb-serial

Questo modulo permette di interagire con il dispositivo in Linux come se fosse un modem seriale analogico. Esso permette inoltre di usare le funzioni write() e read() per mandare o ricevere dati ad un dispositivo via USB. Se il disposivo permette queste 2 funzioni allora verrá usato questo modulo, altrimenti viene rilasciato un modulo dal produttore che implementerá a suo modo queste funzioni. Attualmente il modulo supporta molti convertitori seriali e riesce a gestire ben 256 dispositivi diversi allo stesso tempo. Quando il dispositivo é connesso e riconosciuto dal driver, quest'ultimo provvederá a stampare un messaggio nel log di sistema.

3.4.3 Modifica al modulo cp2101

Recuperati i sorgenti del kernel (versione 2.6.19.2) si é apportata una modifica fondamentale in modo che il modulo riconoscesse l'hardware che doveva essere utilizzato aggiungendo la seguente definizione:

USB_DEVICE(0x10c4, 0x8293)

Dove 0x10c4 e 0x8293 sono cifre in esadecimale che identificano prodotto e produttore del dispositivo.

Una volta apportata la modifica é stato necessario compilare entrambi i moduli utilizzando un cross-compiler fornito da Freetz.

3.4.4 Cos'è un cross-compiler

Un cross-compiler é un compilatore in grado di creare codice eseguibile per una piattaforma diversa da quella su cui viene eseguito il compilatore. Gli strumenti del cross-compiler sono utilizzati per generare eseguibili per sistemi embedded o piú piattaforme. È usato per compilare su una piattaforma su cui non é possibile compilare, come microcontrollori che non supportano un sistema operativo o come in questo caso: un sistema embedded, in cui il dispositivo ha delle risorse estremamente limitate. Il Fritz!Box, ad esempio non é abbastanza potente per eseguire un compilatore o un ambiente di sviluppo. Dal momento che il debug e test possono anche richiedere piú risorse di quelle disponibili su un sistema embedded, la cross-compilazione risulta quindi molto utile allo scopo di ottenere i file compilati per la macchina di destinazione.

GCC, puó essere configurato per compiere una cross-compilazione. Esso supporta varie piattaforme e linguaggi. GCC richiede che una copia compilata del binutils sia disponibile per ogni piattaforma di destinazione.
Sfruttando lo strumento Freetz é stato necessario selezionare i moduli richiesti dal dispositivo, modificare il modulo Cp2101 e infine compilarli.Infatti Freetz automaticamente scarica i sorgenti del kernel per il FreetzBox e una volta apportate le modifiche necessarie provvede alla sua compilazione generando un firmware per il router con le modifiche apportate. Non é stato necessario fare una modifica al firmware, ma si é sfruttata la facilitá con cui vengono compilati i moduli necessari per caricarli successivamente a mano all'interno del Fritz!Box.

3.4.5 Il caricamento dei moduli

Una volta ottenuti i moduli compilati per mipsel si é provveduto a caricarli nel seguente ordine nel sistema attraverso 2 comandi:

insmod usbserial.ko vendor=0x10c4 product=0x8293 Questo comando carica il modulo usbserial registrando un nuovo dispositivo specificando le sue caratteristiche

insmod cp2101.ko s Questo invece carica il modulo cp2101 che gestirà il dispositivo usb

Caricati i moduli viene aggiunto in /dev/ il device file ttyUSB0 con la quale il sistema inizierá ad interfacciarsi. Il device file creato sará quindi un char device ossia un tipo speciale di file che rappresenta una periferica o un dispositivo virtuale su cui é possibile effettuare operazioni di input/output per singoli byte, o comunque per quantitá di dati non rigidamente predeterminate. I dispositivi a caratteri sono caratterizzati da due numeri, detti major number e minor number, che li identificano internamente al kernel, e che sono specifici per la particolare implementazione. I dispositivi a caratteri, pur potendo esistere in qualsiasi punto del file system, sono tipicamente raccolti all'interno della directory /dev; essi presentano nomi e comportamenti che sono specifici per la particolare implementazione. Per ragioni di sicurezza (dato che provvedono accesso diretto all'hardware) possono essere creati solo dal superuser (root). Ottenuto così il collegamento con il dispositivo USB si è realizzato un semplice programma in C che potesse interfacciarsi e quindi spedire e ricevere informazioni da esso.

3.4.6 Mipsel-serial

Le seguenti righe di codice mostrano come avviene l'apertura del device file e quindi le funzioni utilizzate per spedire i comandi e ricevere le risposte:

Listing 3.1: Codice mipsel-serial.c

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <unistd.h>
5  #include <fcntl.h>
6  #include <errno.h>
7  #include <termios.h>
8
9  int fd;
10 /*Questa funzione si occupa di aprire la porta*/
11 int open_port()
12 {
13         FILE *port;
14         char *addr;
15         struct termios options;
16         /*Lettura indirizzo device file da portAddress*/
17         port = fopen("portAddress","r");
18         if (port == NULL)
19         {
20                 return -1;
21         }
22         fscanf(port,"%s", addr);
23         fclose(port);
24         int fd;
25         /*Apertura della porta
26     *Il flag O\_NOCTTY comunica al sistema che questa porta aperta
27     *non deve essere un terminale di controllo,in questo modo nessun segnale
28     *di errore o quant'altro verrà inviato a questo device file.
29     *Con il flag O\_NDELAY si comunica che il segnale DCD verrà ignorato
30         */
31     fd = open( addr , O_RDWR);
32         if (fd == -1)
33         {
34                 perror("open_port error");
35                 exit(-1);
36         }
37         else {
38                 fcntl(fd, F_SETFL, 0);
39                 /*Settaggio delle impostazioni della porta*/
40
41                 tcgetattr(fd, &options);
42                 options.c_cflag &= ~PARENB;
43                 options.c_cflag &= ~CSIZE;
44                 options.c_cflag |= CS8;
45                 cfsetispeed(&options, B19200);
46                 cfsetospeed(&options, B19200);
47                 tcsetattr(fd, TCSANOW, &options);
48
49         }
50         return fd;
51 }
52
53 /*Questa funzione si occupa di leggere la risposta del dispositivo USB
```

```
54  *subito dopo aver inviato il comando
55  */
56  int read_response()
57  {
58          char buffer[1024];
59          int byte=0;
60          char cr;
61          char *ptr;
62          memset(&buffer,0,sizeof(buffer));
63          /*Lettura dal dispositivo USB*/
64          while(read(fd, &cr, 1))
65          {
66                  buffer[byte]=cr;
67                  byte++;
68
69          }
70          /*Se il comando e'andato a buon fine viene
71          *stampato il buffer dell'output cosi' com'e
72          *alrimenti viene indicato nella prima riga un messaggio di errore
73          */
74          ptr = strstr(buffer,"OK\n");
75          if(ptr!='\0')
76          {
77                  printf("%s",buffer);
78                  return 1;
79          }
80          else
81          {
82                  ptr = strstr(buffer,"ERROR");
83                  if (ptr == '\0')perror ("Connection error\n");
84                  else{
85                          fprintf(stdout,"ERROR");
86                          fprintf(stdout,"%s\n",buffer);
87                  }return -1;
88          }
89
90  }
91  /*Invio del comando AT*/
92  int write_command(char *cmd){
93          char c = '\r';
94          int ris;
95          ris = write(fd,cmd,strlen(cmd));
96          if (!ris) {perror("Impossible write\n"); return -1;}
97          ris = write(fd, &c,1);
98          if (!ris) {perror("Impossible write\n"); return -1;}
99          return 1;
100
101 }
102 /*Il primo argomento del main indica il comando AT da eseguire
103 *il secondo indica il tempo d'attesa previsto per la risposta
104 */
105 int main(int argc,char *argv[])
106 {       int sleepT;
107         char *cmd;
108         char cr ='\r';
109         cmd = argv[1];
110         if (argv[2] == '\0')
111                 sleepT = 0;
112         else {
113                 sleepT = atoi(argv[2]);
114
115         }
116         fd = open_port();
117         if(!fd){perror("No port active\n"); exit(-1);}
```

```
118         if (write_command(cmd))
119         {
120                 sleep(sleepT);
121                 read_response();
122         }
123         close(fd);
124         return 0;
125 }
```

Il codice soprariportato è parte del file mipsel-serial.c. Il programma compilato accetta 2 parametri in input.Ad esempio eseguendo: ./mipsel-serial ATI 10 , La stringa ATI verrà spedita direttamente al dispositivo usb sotto forma di comando AT. Il secondo parametro indica il tempo di attesa che intercorre tra l'invio del comando e la lettura del messaggio. Questo è utile perchè vi sono alcuni comandi che necessitano di vari secondi per essere completati (ad esempio : la creazione di una nuova rete Zigbee). Questo file è stato compilato per un architterura mipsel utilizzando sempre uno strumento fornito da Freetz: uClibc.

uClibC

uClibc è una libreria standard C scritta per I sistemi linux embedded. Essa è stata creata per supportare uClinux, una versione di linux che non richiede una memory managment unit. uClibc è un cross-compiler ad hoc che permette facilmente di compilare qualsiasi file sorgente per diversi tipi di architetture tra cui mipsel.

Capitolo 4

Il Webserver

Con delle semplici operazioni é stato possibile creare una pagina web personale all'interno del Fritz!Box, ovviamente visibile da chiunque su internet in grado di gestire il dispositivo Etrx2usb da remoto. Nel file system del Fritz!Box sono disponibili vari file, spesso in modalitá di sola lettura (read-only). Le porzioni scrivibili sono in realtá mappate direttamente nella meomoria RAM o Flash. E' quindi possibile trasferirvi del software e aggiungere delle nuove funzionalitá. Tra i vari pacchetti utilizzabili, vi é websrv: questo pacchetto é un server web minimale utilizzato per l'interfaccia web di configurazione del router. Non potendo essere eseguito con porta di ascolto diversa dalla 80 (http) non é possibile riutilizzarlo per servire ulteriori contenuti se non modificando l'interfaccia di configurazione stessa, quindi per aggiungere una pagina personale all'interno del router si é dovuto aggiungere nel Fritz!Box un altro server web: Apache. Essendo dotato di una memoria flash, non é possibile mantenere il server e i relativi file all'interno del Fritz!Box se non modificando il firmware, si é dovuto quindi realizzare uno script che copiasse il contenuto del server da un supporto usb all'interno del Fritz!Box e procedere con l'avvio.

4.1 Apache

Apache è uno dei webserver modulare più diffusi. E' stato scelto in questo progetto per l'incredibile efficienza di cui dispone. Esso infatti supporta numerose funzioni implementate attraverso dei moduli che ne estendono le funzionalità di base tra cui il supporto di molti linguaggi di programmazione server-side (PHP,Python). Tra le caratteristiche più importanti vi sono:

- Disponibilità del codice sorgente (progetto open-source)
- Portabilità: supporto dei SO Linux, Unix, Windows.
- Efficienza, flessibilità
- Supporto dei più recenti protocolli (compatibilità HTTP/1.1)
- Stabilità ed affidabilità
- Processo di sviluppo open
- Modularità
- Nucleo (core) molto piccolo che realizza le funzionalità di base
- Possibilità di estendere le funzionalità mediante moduli (scritti usando l'Apache module API) compilati staticamente nel nucleo oppure caricati dinamicamente a tempo di esecuzione

I moduli permettono di gestire particolari funzionalità che vengono controllate dal core, un programma principale composto da un ciclo sequenziale. La modulazione viene scomposta nel seguente modo:

Translation traduce la richiesta del client

Acces Control controlla eventuali richieste dannose

MIME Type verifica il tipo di contenuto

Response invia la risposta al client e attiva eventuali procedure

Logging tiene traccia di tutto ciò che è stato fatto

4.2 Telegesis panel control

Affinché la comunicazione con il dispositivo USB risultasse piú user-friendly sono stati realizzati alcuni script in php che permettono di visualizzare i comandi disponibili e metodi in grado di realizzare i propri comandi personalizzati lasciando all'utente finale la possibilitá di decidere quali utilizzare. Questa scelta é stata fatta proprio perché i comandi AT permettono di interfacciarsi con il dispositivo USB a livello piú basso e il set di comandi risulta molto vasto. E' possibile modificare persino i registri del dispositivo per gestire una rete Zigbee, ma molti comandi potrebbero non venire mai utilizzati, quindi si é pensato di realizzare un'applicazione in grado di tenere traccia dei comandi che possono differenziarsi per ogni utente in modo tale da fornire gli strumenti generali per creare un set di comandi ad hoc. Se FritzBee venisse utilizzato ad esempio per tener traccia solamente dei consumi elettrodomestici, potrebbero essere scritti e memorizzati i comandi che permetterebbero, dato un comando, di interrogare i vari sensori disposti all'interno o all'esterno della propria abitazione. Oltre alla possibilità di eseguire comandi predefiniti, l'interfaccia web(Control.php) fornisce la possibilità di aggiungere nuovi comandi che verranno memorizzati in un file Xml (commandSet.xml) all'interno della directory del server. In qualsiasi momento è possibile effettuare un backup del file in modo tale da riutilizzare gli stessi comandi ogni qualvolta si decide di spegnere il Fritz!Box.

Tra le scelte implementative si é voluto distinguere l'applicazione scritta in Php da quella scritta in C (mipsel-serial) che comunica direttamente il comando al dispositivo USB, questo perché l'idea é stata di creare uno strumento che agisca su piú livelli in modo tale da non rendere possilibile la comunicazione con il dispositivo solo via web, ma dare la possibilitá di implementare altri modi di comunicare con il dispositivo usb . Inoltre l'interfaccia con la quale inviare, scrivere e realizzare i propri comandi é lasciata direttamente allo sviluppatore del client che attraverso una richiesta di tipo post comunica al server la creazione o l'esecuzione di un comando.

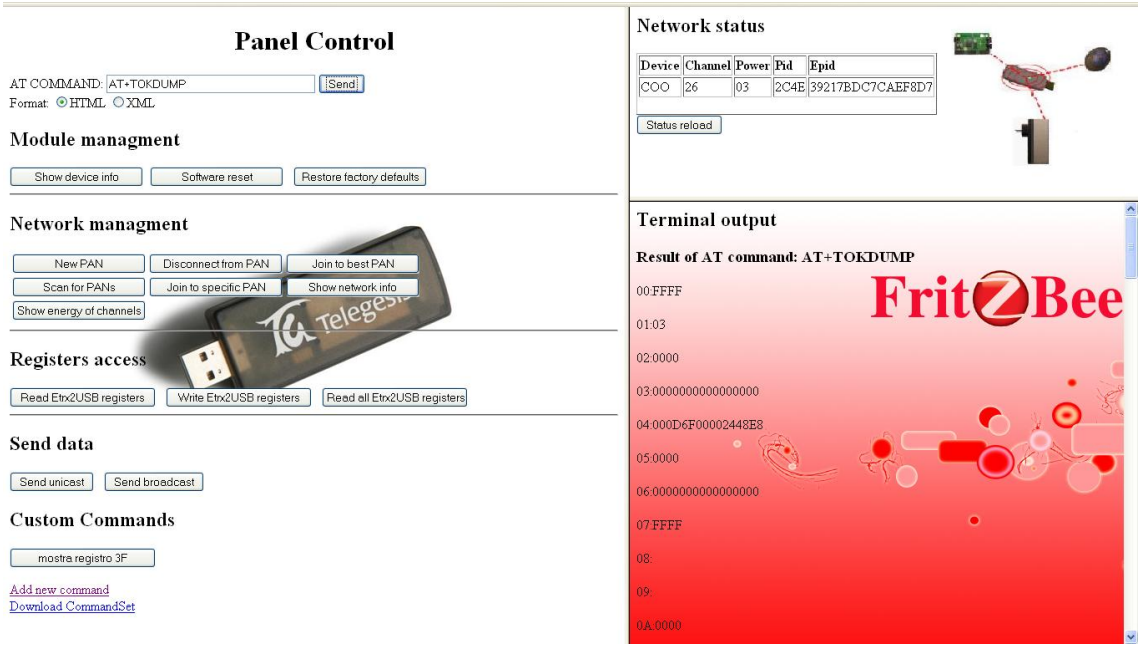


Figura 4.1: L'interfaccia web di FritzBee

Lo script control.php mostra una pagina web formata da 3 frame. Nel primo (a sinistra) vi sono i comandi disponibili di default più quelli inseriti dall'utente. Nel secondo (in alto a destra) viene visualizzato lo stato della rete, mentre nel terzo (in basso a destra) viene visualizzato l'output del dispositivo USB.

Lo script input.php

Lo script crea una pagina web con la lista dei comandi disponibili e viene visualizzata nella parte sinistra della pagina di controllo; prima di inviare il comando è possibile scegliere il formato dell'output richiesto:

1. Html
2. Xml

Alcuni comandi hanno la possibilità di specificare dei parametri aggiuntivi, quindi alla pressione del comando richiesto, la text-area in cima alla pagina viene riempita con il comando AT, in questo modo si lascia la possibilità all'utente di specificare i parametri e di inviare il comando attraverso il tasto: Send. Di seguito è possibile visualizzare il codice dello script che si occupa di mostrare i comandi di default e quelli definiti dall'utente. In fondo alla pagina vi sono due link. Cliccando su: Download CommandSet sarà possibile salvare il command Set creato per riutilizzarlo ogni volta che si vorranno riutilizzare i comandi specifici definiti dall'utente. Il link: Add new command apre la pagina di creazione di un nuovo comando.Questo codice è contenuto nel file: input.php.

Listing 4.1: Codice input.php

```
<html>
<head>

<!-- Lo script serve a riempire l'area testo
     posta in cima alla pagina,per poter modificare
     o aggiungere parametri al comando AT
     prima di inviarlo -->
<script language="javascript">

function showCommand(command){
    document.insertCommand.cmd.value = command;
    }
</script>
<style type="text/css">
body{
      background-image : url('public/etrx2usb.jpg');
      background-position: center;
      background-repeat: no-repeat}
</style>
</head>

<body>
    <div>
        <h1 style="text-align:center;">Telegesis Terminal Control</h1>
        <form name="insertCommand" action="output.php" target="output"  method="post">
            AT COMMAND: <input style="width:250px;" type="text" name="cmd" id="cmd"
                value="" >
            <input type="submit" value="Send"></input><br>
            Format: <input type="radio" name="mode" value="html" checked>HTML</input>
            <input type="radio" name="mode" value="xml" >XML</input>
            </form>
        <form name='control'>

        <h2>Module managment</h2>

            <input type='button' style="width:160px;" value="Show device info" onclick="
                javascript:showCommand('ATI')"/> 
            <input type='button' style="width:160px;" value="Software reset" onclick="
                javascript:showCommand('ATZ')"/>  
            <input type='button' style="width:160px;" value="Restore factory defaults"
                onclick="javascript:showCommand('ATF')"/> </br>
<hr/>
        <h2>Network managment</h2>

        <table>
            <tr>
            <td><input type='button' style="width:160px;" value="New PAN" onclick="
                javascript:showCommand('AT+EN')"/></td>
            <td><input type='button' style="width:160px;" value="Disconnect from PAN"
                onclick="javascript:showCommand('AT+DASSL')"/></td>
            <td><input type='button' style="width:160px;" value="Join to best PAN"
                onclick="javascript:showCommand('AT+JN')"/></td>
            </tr>
            <tr>
            <td><input type='button' style="width:160px;" value="Scan for PANs" onclick
                ="javascript:showCommand('AT+PANSCAN')"/></td>
            <td><input type='button' style="width:160px;" value="Join to specific PAN"
                onclick="javascript:showCommand('AT+JPAN:<PID or EPID>')"/></td>
            <td><input type='button' style="width:160px;" value="Show network info"
                onclick="javascript:showCommand('AT+N')"/></td>
            </tr>
            <tr>
```

```
53              <td><input type='button' style="width:160px;" value="Show energy of channels
                    " onclick="javascript:showCommand('AT+ESCAN')"/></td>
54              <td></td>
55              <td></td>
56              </tr>
57
58          </table>
59
60  <hr/>
61          <h2>Registers access</h2>
62              <input type='button' style="width:175px;" value="Read Etrx2USB registers"
                    onclick="javascript:showCommand('ATS<REG>[bit]')"/>  
63              <input type='button' style="width:175px;" value="Write Etrx2USB registers"
                    onclick="javascript:showCommand('ATS<REG>[bit]=<data>')"/>  
64              <input type='button' style="width:175px;" value="Read all Etrx2USB registers
                    " onclick="javascript:showCommand('AT+TOKDUMP')"/>  
65  <hr/>
66          <h2>Send data</h2>
67              <input type='button' value="Send unicast" onclick="javascript:showCommand('
                    AT+UCAST:<Enter Parameter here>')"/>  
68              <input type='button' value="Send broadcast" onclick="javascript:showCommand
                    ('AT+BCAST:<Enter Parameter here>')"/>  
69          <h2>Custom Commands</h2>
70
71
72  <?php
73          /*Recupero delle informazioni contenute nel file XML
74          * Il ciclo leggera i comandi memorizzati nel file,
75          * producendo un pulsante che servirà a lanciarme
76          * l'esecuzione del comando AT corrispondente
77          */
78          $xml = simplexml_load_file("public/commandSet.xml");
79          $array = $xml->children();
80          foreach($xml->children() as $child)
81          {
82              $attrib = $child->attributes();
83
84             ?>
85              <input type='button'style="width:175px;"  value="<?php echo $child; ?>"
                    onclick="javascript:showCommand('<?php echo $attrib; ?>')"/> 
86    <?php
87     }
88  ?>
89  </form>
90
91
92  <a href="create.php" target="_self"> Add new command</a><br>
93  <a href="public/commandSet.xml"> Download CommandSet </a>
94
95  </body>
96  </html>
```

Lo script create.php

Questo script crea una pagina fornita di un form, con la quale è possibile aggiungere un nuovo comando specificandone il nome e il relativo comando AT. Dopo che il nuovo comando è stato creato è possibile testarlo prima di apportare le modifiche al file commandSet.xml. Una volta che il comando è stato salvato è possibile vederlo fin da subito nella lista dei comandi disponibili cliccando su: Get Back. Di seguito è mostrato il form con la quale è possibile creare un nuovo comando e il relativo codice:

Figura 4.2: La creazione di un nuovo comando

Listing 4.2: Codice create.php

```
<html>
<head>
<script language="javascript">

function showCommand(command){
    document.insertCommand.cmd.value = command;
    }
</script>
<style type="text/css">
body{ background-image : url('etrx2usb.jpg');
      background-position: center;
      background-repeat: no-repeat}

</style>
</head>
<body>

<h1>Create new command</h1>
<form action="<?php $_SERVER['PHP_SELF'];?>" method="post" >
<table>
<tr>
<td>Name: </td>
<td><input type="text" name="name"></input></td>
</tr>
<tr>
<td>AT Command:</td>
<td><input type="text" name="cmd"></input></td>
</tr>
<tr>
<td></td>
<td><input type="submit" value="Add command"></input></td>
</tr>
</table>

<input type="hidden" name="create" value="1"></input>

</form>

<?php
$cmdName = "";
$cmdAT = "";
/*Viene controllato se è stato
* richiesto di creare un nuovo comando
*/
if(isset($_POST['create']))
{
        $cmdName = $_POST['name'];
        $cmdAT = $_POST['cmd'];

        echo "<h3>Testing</h3>";
        /*Viene fornito un form sul quale testare l'invio
        * dei comandi
        */
        ?>
        <input type='button'style="width:175px;"  value= "<?php echo $cmdName; ?>"
             onclick="javascript:showCommand('<?php echo $cmdAT; ?>')" /> 
        <form name="insertCommand" action="output.php" target="output"  method="post">
            AT COMMAND: <input style="width:250px;" type="text" name="cmd" id="cmd"
                 value="" >
            <input type="radio" name="mode" value="html"checked>HTML</input>
            <input type="radio" name="mode" value="xml" >XML</input>
            <input type="submit" value="Send"></input>

```

```
62          </form>
63          <form action="<?php $_SERVER['PHP_SELF'];?>" method="post" >
64              <input type="hidden" value="1" name="save"></input>
65              <input type="hidden" name="name" value="<?php echo $cmdName ?>"></input>
66              <input type="hidden" name="cmd" value="<?php echo $cmdAT ?>"></input>
67              <input type="submit" value="Save"></input>
68          </form>
69
70  <?php
71
72  }
73      /*quando si è certi che il comando restituisce
74      * l'informazione richiesta,si può proseguire
75      * al suo salvataggio nel file XML
76      */
77      if(isset($_POST['save'])){
78       echo "<h2>New Command added</h2>";
79       $xml = simplexml_load_file("public/commandSet.xml");
80          $child = $xml->addChild("command",$_POST['name']);
81          $child->addAttribute("instruction",$_POST['cmd']);
82
83
84          $xmlHandle = fopen("public/commandSet.xml","w");
85          fwrite($xmlHandle, $xml->asXml());
86          fclose($xmlHandle);
87
88      }
89
90  ?>
91
92  <a href="input.php"> Get back</a>
93
94  </body>
95
96  </html>
```

Lo script status.php

Tramite questo script viene caricato nel frame in alto a destra lo stato della rete. In ogni momento è possibile aggiornare lo stato cliccando sul tasto: Status reload. Dopo la pressione del tasto verrà inviato al dispositivo il comando AT: AT+N. Nel caso in cui si è connessi ad una rete Zigbee verranno visualizzati tutti i dispositivi connessi con le loro relative informazioni. Nella figura sottostante è possibile visualizzarne un esempio:

Figura 4.3: Visualizzazione dello stato della rete Zigbee

Di seguito sono riportate le righe di codice dello script:

Listing 4.3: Codice status.php

```
<html>
<head>
<style type="text/css">

body
{
background-image : url('public/devices.jpg');
background-position: right;
background-repeat:no-repeat;
background-attachment:fixed;
}
</style>

</head>
<body>

<?php
    /*La funzione checkStatus invia il comando
    * AT+N al dispositivo USB, la risposta generata
    * verrà poi gestita per creare una tabella
    * con l'elenco dei dispositivi connessi alla rete
    */
    function checkStatus(){
    $array=explode("\n",shell_exec('/var/tmp/mipsel-serial AT+N'));

    $coo = substr($array[3],3);
    if ($coo == "NoPan" || is_null($coo))
        echo "No network";
    else{
    list($device,$channel,$power,$pid,$epid) = explode(",",$coo);

    echo "
    <table border='1'>
    <tr>
        <td>Device</td>
        <td>Channel</td>
        <td>Power</td>
        <td>Pid</td>
        <td>Epid</td>
    </tr>
    <tr>
        <td>$device</td>
        <td>$channel</td>
        <td>$power</td>
        <td>$pid</td>
        <td>$epid</td>

    </tr>
    ";
    for($i=4;$i<sizeof($array)-3;$i++){
            if (isset($array[$i])){
                list($device,$channel,$power,$pid,$epid) = explode(",",$array[$i]);

            echo "<tr>
                    <td>$device</td>
                    <td>$channel</td>
                    <td>$power</td>
                    <td>$pid</td>
                    <td>$epid</td>

```

```
61                  </tr>";
62              }
63
64      }
65      echo "</table>";
66      }
67  }
68
69  ?>
70
71  <?php
72
73  /*La funzione checkStatus() viene
74  * richiamata ogni volta che la pagina
75  * viene aggiornata, per aggiornare lo stato
76  * è necessario cliccare sul tasto reload
77  */
78
79  checkStatus();
80  if (isset($_POST['reload']))
81      Header ('Location: status.php');
82  ?>
83  <form action="" method="post">
84  <input type="submit" value="Status reload"></input>
85  <input type="hidden" name="reload" value="1"></input>
86  </form>
87
88  </body>
89  </html>
```

Lo script output.php

Questo script carica una pagina con l'output del dispositivo usb. Ogni volta che un comando è stato eseguito, lo script si preoccupa di lanciarlo attraverso la funzione:

```
1 shell_exec("./mipsel-serial $cmd")
```

A seconda di come si è specificata l'opzione dell'output, lo script potrà mostrare la risposta del dispositivo attraverso semplice codice Html, oppure in Xml. Se si decide di mostrare l'output in formato Xml, il codice prodotto verrà salvato nel formato: output.xml. Questo file xml contiene la risposta del dispositivo strutturata nel seguente modo:

```
1 <COMMAND name="name of command">
2         <OUTPUT>
3         output of command
4         </OUTPUT>
5 </COMMAND>"
```

Altrimenti nel frame in basso a sinistra verrà visualizzato il seguente output:

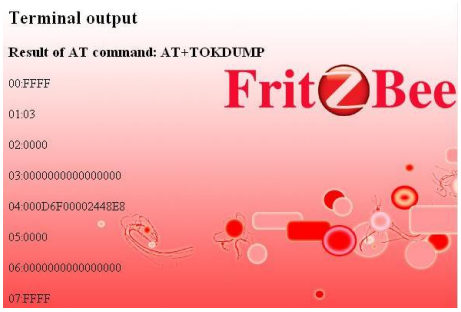


Figura 4.4: Visualizzazione dell'output in html

Nella pagina seguente è riportato il codice dello script che si occupa di inviare i comandi all'applicazione serial-mipsel e di mostrare l'output nel frame destro della pagina di controllo.

Listing 4.4: Codice output.php

```
<html>
<head>
<style type="text/css">
body{
      background-image : url('public/abastract.jpg');
}
</style>

</head>
<body>
<h2>Terminal output</h2>
<?php

/*La funzione seguente produce
  * il file XML leggendo gli elementi
  * dell'array passato come parametro
*/
function toXML($cmd,$array){

   $output="";
   for($i=1;$i<sizeof($array)-1;$i++){
    if(isset($array[$i])){
        $output = "$output
                   $array[$i]";
        }
   }
        $xmlOut = "<?xml version='1.0' encoding='UTF-8'?>
               <COMMAND name='$array[0]'>
                    <OUTPUT>
                         $output
                    </OUTPUT>
                </COMMAND>";

        $xmlFile =  "public/".$cmd.".xml";
        $xmlHandle = fopen($xmlFile,"w");
        fwrite($xmlHandle, $xmlOut);
        fclose($xmlHandle);
        echo "Xml created<br>";
        echo "<br>a href=".$xmlFile.">Save file</a>";

}
/* Viene controllato se è stato inviato
 * un nuovo comando dalla pagina web
 * In caso positivo si procede con l'invio
 * del comando AT al dispositivo USB
*/
if(isset($_POST["cmd"])){
    $cmd = $_POST["cmd"];
        /*La risposta generata dal dispositivo USB
        * viene memorizzata all'interno di un array
        * nella quale,ogni riga è un suo elemento
        */
        $array = array();
        if ($cmd == "AT+EN" || $cmd == "AT+PANSCAN" || $cmd == "AT+ESCAN"){
            echo "Take up 20 seconds";
            $array = explode("\n",shell_exec("/var/tmp/mipsel-serial $cmd 20"));
        }
        else{
            $array = explode("\n",shell_exec("/var/tmp/mipsel-serial $cmd"));
        }
        /* Viene controllato quale formato
        * dell'output è stato richiesto
        */
```

```
64          if($_POST["mode"] == "html")
65              {
66             echo "<h3>Result of AT command: $cmd </h3>";
67                  for($i=1;$i<sizeof($array)-1;$i++){
68                  echo "$array[$i]<br>";
69
70                  }
71              }
72          else toXML($cmd,$array);
73          }
74
75  ?>
76  </body>
77  </html>
```

4.3 La configurazione - Lo script FritzBee.sh

Lo script FritzBee.sh si occupa di copiare e installare i moduli di cui necessita il dispositivo usb. Inoltre essa installa il webserver in una directory temporanea del Fritz!Box e procede con l'avvio. Nella directory di installazione sarà presente anche il file commandSet.xml precedentemente salvato, in modo tale da garantire un ripristino ogni qualvolta che il Fritz!Box verrà ravviato. La directory FritzBee contiene sia il server che i moduli del dispositivo usb, eseguendo lo script sh con gli argomenti 'module' e 'server' verranno installati tutti i file all'interno della directory /var/tmp del Fritz!Box.

Listing 4.5: Codice fritzbee.sh

```
1  #! /bin/sh
2  if [ $1 = "module" ]
3          then
4                  echo "Loading module"
5                  insmod /lib/modules/2.6.19.2/kernel/drivers/usb/serial/usbserial.ko
                       vendor=10C4 product=8293
6                  insmod cp2101.ko
7                  echo "Copyng mipsel-serial"
8                  cp mipsel-serial /var/tmp/
9  fi
10 if [ $2 = "server" ]
11         then
12                 echo "Copyng server"
13                 cp -rf apache-1.3.41 /var/tmp/
14                 echo "Server init"
15                 /var/tmp/apache-1.3.41/apache -f /var/tmp/apache-1.3.41/apache.conf
16                 chmod -R  777 /var/tmp/apache-1.3.41/www/public
17
18 fi
19 echo "Ready for Fritzbee connection"
```

4.4 Possibili sviluppi

Utilizzando Php è possibile implementare facilmente altre funzionalità. Di seguito sono riportate alcune idee e possibili implementazioni future.

4.4.1 Modifica del firmware Fritz!Box

Successivamente si potrà integrare il webserver e i moduli necessari all'interno del firmware in modo tale da garantire il funzionamento di FritzBee anche quando il Fritz!Box verrà ripristinato. Attraverso Freetz verrà generata un'immagine del firmware contentente i files di FritzBee.

4.4.2 Database Sqlite

Installando un database Sqlite sarà possibile memorizzare i dati ricevuti dalla rete Zigbee. Attraverso un'applicazione Php si avrà la possibilità d'interrogare il database oppure salvare impostazioni della rete. Le notevoli prestazioni di Sqlite e la sua compattezza garantirebbero una memorizzazione dei dati molto efficente.

4.4.3 WebDav

Attraverso il support di Fritz!Box al protollo webdav sarà possibile garantire l'accesso ai dati in condivisione da remoto. Verrebbero memorizzati dati come ad esempio log appartenenti alla rete Zigbee e visualizzabili da chiunque abbia accesso alle risorse della rete. Potrebbe essere implementato uno script php che si occupa di memorizzare i dati del dispositivo Etrx2usb e renderli accessibili attraverso webdav. Esiste gia un estensione di php in grado di lavorare con questo protollo.

Conclusioni

Il progetto realizzato in questa tesi si pone come punto d'inizio di uno sviluppo che tenterò di portare avanti in futuro. Molte sono le idee che durante l'implementazione e la progettazione sono venute fuori. L'idea, fin dal principio, è stata quella di avere un unico dispositivo che permettesse di poter gestire più cose possibili riducendo sia nella complessita che nella quantità altri dispositivi gia esistenti. Lo spunto iniziale dato dal mio relatore è stato infatti quello di farmi notare quanti sono gli apparecchi che ci circondano, uno per ogni funzione. Andando avanti negli anni avremo decine di telecomandi sparsi per casa e quando gli elettrodomestici inizieranno a diventare sempre più intelligenti, o sempre più sofisticati, perderemo più tempo per capire come essi funzionino piùttosto che usarli. L'idea di rendere la domotica d'uso comune piùttosto che un privilegio per pochi è l'altra idea che sin dall'inizio ha alimentato questo progetto. Ai giorni d'oggi un controllo domotico per l'illuminazione, ad esempio, viene a costare qualche centinaio di euro. L'acquisto comprende una centralina domotica e qualche dispositivo in grado di regolare l'accensione e lo spegnimento delle luci.FritzBee invece è un esempio concreto di come un router, pensato gia per svolgere numerosissime funzioni legate alla telefonia, possa essere potenziato per controllare e gestire cose che vanno al di fuori degli scopi per cui un router domestico è stato pensato. L'approccio è stato quello di non pensare alla progettazione di un gateway domotico, cosa che gia esiste da tempo, ma di usare un dispositivo gia esistente per far qualcosa di nuovo. La filosofia che sta alla base di questo progetto è quella di saper non reinventare la ruota. La tesi svolta

vuole quindi mostrare come qualcosa di nuovo può essere implementato, partendo semplicemente dalle tecnologie gia esistenti. Putroppo per questioni di tempo non si è potuto testare un funzionamento completo del progetto, ma una tesi nata come studio teorico sulla fattibilità di esso ha reso possibile uno studio approfondito sul come e sulla possibilità che il progetto FritzBee potesse essere realizzato. Sapendo che la chiavetta Etrx2USB funziona perfettamente in un ambiente dove siano gia presenti onde wireless con frequenze utilizzate dal procollo 802.11 si è andato avanti nello sviluppo della tesi.Infatti la parte fondamentale dello studio è stata quella di far comunicare il router dell'AVM con essa. Una volta che si è garantita una corretta comunicazione tra i due disposivi, si è passato ad uno studio di come essa potesse essere gestita da remoto. Durante l'implementazione del progetto, cosa che ha richiesto mesi di lavoro, si pensava a come questo potesse essere utile ad un utente che non ne facesse un uso industriale o commerciale, ma piùttosto uno domestico. Le idee che ne sono venute fuori sono molteplici e mentre a volte si faceva sentire lo sconforto di non poter portare a termire il progettto, le idee e le applicazioni che potevano essere realizzate davano forza per riuscire a concluderlo.

Bibliografia

[1] http://www.zigbee.org
Sito aziendale di Zigbee

[2] http://www.avm.de/de/
Sito casa produttrice del Fritz!Box

[3] http://www.telegesis.com/zigbee_pro
Informazioni sul protocollo Zigbee

[4] http://www.telegesis.com/products/test_page_2.htm
Informazioni sul dispositivo Etrx2USB

[5] http://www.telegesis.com/downloads/general/TG-ETRX2USB-PM-004-105.pdf
Manuale dispositivo Etrx2USB

[6] http://www.telegesis.com/downloads/general/TG-ETRX2-PM-001-109.pdf
Manuale modulo Etrx2

[7] http://www.silabs.com/products/interface/usbtouart/Pages/default.aspx
Sito casa produttrice chip2101

[8] http://tldp.org/HOWTO/Serial-HOWTO.html
Serial HOWTO (David S.Lawyer)

[9] http://cassettone.rossonet.com/Pluto/HOWTO/Modem-HOWTO/Modem-HOWTO-15.html
CHIP-UART

[10] http://wiki.vocesuip.com/index.php?title=AVM
wiki Fritz!Box

[11] http://trac.freetz.org/wiki
wiki Freetz

[12] http://www.apache.org
Sito del webserver Apache

Ringraziamenti

In questi ultimi mesi passati a realizzare la tesi, molte sono le persone che dovrei ringraziare. Inizialmente vorrei ringraziare un mio collega universitario: Valvassori Matteo, con cui da mesi è iniziata una collaborazione per poter sviluppare questo progetto. L'argomento della sua tesi incentrata su Zigbee introduceva le problematiche e le idee che ho successivamente sviluppato in questo progetto. La sua collaborazione è stata di fondamentale importanza, dandomi idee e nuovi spunti per poter andare avanti con la tesi nei momenti in cui la soluzione ai vari problemi sembrava lontana.
Ringrazio tutte le persone che mi sono state vicine sopratutto nelle ultime settimane di lavoro dove l'ansia e la voglia di portare a termine la tesi si facevano sempre più sentire. Ringrazio quindi i miei genitori e la mia ragazza, che mi hanno dato la forza di andare avanti nei momenti più difficili di questi mesi.
Infine ringrazio il mio relatore, il professor Danilo Montesi, che mi ha fornito idee e spunti per poterla realizzare, di conseguenza ringrazio l'università al quale devo le mie conoscenze informatiche acquisite in questi anni di studio.