SAPIENZA
UNIVERSITÀ DI ROMA

Facoltà di Ingegneria

***

Corso di laurea specialistica in
Ingegneria delle telecomunicazioni

Riconoscimento automatico di reti Bluetooth attraverso la piattaforma SDR Universal Software Radio Peripheral

A.A. 2009/2010

Relatore
Prof.ssa
M.-G. Di Benedetto

Candidato
Sergio Benco

Supervisore (CSP)
Ing. Andrea Ghittino

Ringraziamenti

Sono davvero molte le persone che meritano di figurare in questa pagina e dalle quali non posso smettere tutt'ora di trarre insegnamento. Questo traguardo è solo una occasione di poter dire quanto siete importanti.

Una di queste persone è, senza dubbio, la Prof.ssa Maria Gabriella Di Benedetto che ha dimostrato sin da subito di credere in me, nelle mie capacità, ciò di cui le sarò sempre grato. In un momento in cui avevo perso la forza di continuare gli studi, ha accolto la mia insolita richiesta (non avendo ancora terminato gli esami) di partecipare alle attività del suo laboratorio. Nel periodo di circa un anno che ne è seguito ho studiato, condiviso idee, imparato più che in ogni altro momento della mia carriera universitaria.

In ACTS ho avuto modo di confrontarmi con Luca e con il Dome, due assistenti, professori, ingegneri ma soprattutto amici, davvero straordinari. E' stato davvero un piacere imparare ogni giorno dalla vostra esperienza.

Voglio ringraziare e abbracciare tutti gli amici che, da "coinquilini" del lab, hanno condiviso con me i giorni in ACTS: Stefano Boldrini, Jesùs Roldàn Diaz, Carmen J. Martin Martin, Gabriele Tucciarone, Marco Mariani e Ciro Sirignano.

Un'altra persona che ha creduto in me e che ringrazio di cuore, è l'Ing. Andrea Ghittino, per avermi dato l'opportunità di venire quì a Torino in CSP "Innovazione nelle ICT" a svolgere il lavoro oggetto di questa tesi. Ringrazio molto anche l'Ing. Stefano Annese per avermi fornito preziosi suggerimenti tecnici in ogni fase del lavoro. Saluto, inoltre, l'Ing. Roberto Borri, per avermi accolto calorosamente nella sua squadra di ricercatori al CSP di Torino.

In CSP ho potuto lavorare in un ambiente dinamico e produttivo ricco di persone competenti. Tra questi posso menzionare: Floriana "Flo", Francesca, Jingye, Livio, Simone, Nazario, Fabio, Mimmo, Giuliano, Matteo, Sandro, Danilo, Mario, Salvo, e tutti gli altri.

Un sincero grazie e un abbraccio a Marco, Elisa, Gioella, Vincenzo, Francesca, Davide, Stefano, Augusto, Valentina, Laura, Maria Anna, Valentina, Marco, Simone, Anna, Valerio, Sara, Dalila, Claudio, Elisa, Alessandro e tutti gli altri che mi perseguiteranno per non averli inclusi qui, ma purtroppo la pagina è finita! Vi voglio bene!!!

Torino, 24 Maggio 2010

"...nella vita importa non già di essere forti, ma di sentirsi forti;
di essersi misurati almeno una volta,
di essersi trovati
almeno una volta, nella condizione umana più antica,
soli davanti alla pietra cieca e sorda,
senza altri aiuti
che le proprie mani, e la propria testa."

(C. J. McCandless)

Un ricordo commosso
va al mio vicino, collega e amico,

Donato

A mio padre Francesco,

per aver sempre creduto in me,
e per avermi insegnato
anche attraverso il suo lavoro,
a svolgere sempre ogni compito
con impegno ma, soprattutto,
con il cuore.

A mia madre Emilia,

che non ha mai smesso di ascoltarmi,
anche nei momenti più difficili,
riuscendo ogni volta a farmi rialzare
grazie alle mie sole forze.

A mio fratello Mauro,

di avermi sempre fatto sentire importante,
il suo riferimento,
così da incoraggiarmi sempre
a far meglio e a credere
in me stesso.

SAPIENZA
UNIVERSITÀ DI ROMA

Questo lavoro è stato sviluppato nell'ambito dell'accordo trilaterale tra la Facoltà di Ingegneria dell'Università di Roma “Sapienza” (Dip. INFOCOM, lab ACTS), il Politecnico di Torino (Dip. DELEN) e il CSP “Innovazione nelle ICT” di Torino. Nel contesto di questo accordo è stata emessa la borsa di ricerca sul tema “Analisi e classificazione dell'interferenza in reti wireless eterogenee” della durata di sei mesi, da svolgersi tra Roma (due mesi) e Torino (quattro mesi). Nel periodo trascorso nel lab ACTS (Roma), le attività di ricerca si sono svolte sotto la guida della prof.ssa M.-G. Di Benedetto. In CSP (Torino), il resto del lavoro è stato svolto sotto la guida dell'ing. Andrea Ghittino e dell'ing. Stefano Annese.

Indice

1 Introduzione.....1

1.1 La banda ISM.....5

1.2 La Radio Cognitiva e la SDR.....8

2 La tecnologia Bluetooth.....11

2.1 Descrizione generale.....13

2.2 Standard IEEE 802.15.1 e le specifiche del Bluetooth SIG.....14

2.3 Tipologie di pacchetto Bluetooth.....24

2.4 Le specifiche di strato fisico.....35

2.5 Il segnale Bluetooth.....42

3 GNUradio e l'USRP.....50

3.1 Descrizione generale dell'USRP.....51

3.2 Universal Software Radio Peripheral (USRP).....57

3.3 Misure sperimentali sull'USRP.....67

3.4 Misure sperimentali sull'USRP2.....74

3.5 Il tool di sviluppo GNUradio.....78

4 Identificazione del segnale Bluetooth.....89

4.1 Energy detector.....94

4.2 Estrazione delle features.....100

4.3 Durata e tempo di interarrivo dei pacchetti (link ACL).....102

4.4 Durata e tempo di interarrivo dei pacchetti (link SCO).....111

4.5 Recognition time di una trasmissione Bluetooth.....115

4.6 APPENDICE A.....120

4.7 APPENDICE B.....123

4.8 Bibliografia.....129

4.9 Allegato A (paper CogART 2010).....132

Indice delle figure

Fig. 1.1: Il ciclo di operazioni di una radio cognitiva....2
Fig. 1.2: Il modulo AIR-AWARE (ED, Feature Extractor e Classificatore)....3
Fig. 2.1: Andamento mondiale delle vendite di chipset Bluetooth....11
Fig. 2.2: Il logo Bluetooth....12
Fig. 2.2.1: Parallelo tra gli strati ISO/OSI e lo stack Bluetooth....15
Fig. 2.2.2: Protocolli impiegati dai diversi livelli dello stack Bluetooth....16
Fig. 2.2.3: La ripartizione dei canali fisici in Bluetooth e l'AFH....17
Fig. 2.2.4: Topologia di rete di una piconet....18
Fig. 2.2.5: Topologie di rete Bluetooth: Master-Slave, Piconet, Scatternet....18
Fig. 2.2.6: Stati di funzionamento di un dispositivo Bluetooth....19
Fig. 2.2.7: Scambio di messaggi in un paging....21
Fig. 2.2.8: Procedura di paging....22
Fig. 2.2.9: Stato di connessione e suoi sottostati....22
Fig. 2.2.10: Schema di polling per comunicazioni multislave....23
Fig. 2.3.1: Struttura di un pacchetto Bluetooth (Basic Rate, BR)....24
Fig. 2.3.2: Rappresentazione little-endian vs big-endian....25
Fig. 2.3.3: Bluetooth Device Address....25
Fig. 2.3.4: Access Code (AC) dei pacchetti Bluetooth....26
Fig. 2.3.5: Tipologie di Access Code (AC)....27
Fig. 2.3.6: Preambolo dell'Access code....27
Fig. 2.3.7: Trailer dell'Access code....28
Fig. 2.3.8: Campi all'interno dell'Header....28
Fig. 2.3.9: Tipologie di pacchetto (da 1, 3, 5 slot)....29
Fig. 2.3.10: Il pacchetto FHS....30
Fig. 2.3.11: Descrizione dei campi del pacchetto FHS....31
Fig. 2.3.12: Bitstream processing....33
Fig. 2.3.13: Whitening e de-whitening nel bitstream processing....33
Fig. 2.3.14: Schema FEC 1/3....34
Fig. 2.4.1: Lo stack Bluetooth....35
Fig. 2.4.2: Livello Bluetooth Radio e strati superiori nello stack Bluetooth....36
Fig. 2.4.3: Allocazione canali fisici nella banda ISM nei diversi paesi....37
Fig. 2.4.4: Modulazione 8DPSK....38
Fig. 2.4.5: Modulazione DQPSK....38
Fig. 2.4.6: Ricetrasmissione di pacchetti di durata pari a 1 slot nel Master....39
Fig. 2.4.7: Ricetrasmissione di pacchetti di durata pari a 1 slot nello Slave....39
Fig. 2.4.8: Clock e sincronizzazione dello slave al clock della piconet....40
Fig. 2.4.9: Periodi caratteristici del sistema Bluetooth....40
Fig. 2.4.10: Selezione della hopping sequence....41
Fig. 2.4.11: Trasmissione di pacchetti multislot....41
Fig. 2.5.1: Pulse shaping gaussiano nella modulazione GFSK....46
Fig. 2.5.2: Durata dell'impulso gaussiano rispetto al tempo di simbolo Ts....46
Fig. 2.5.3: Shift di frequenza (fd) con shape gaussiano (GFSK)....47
Fig. 2.5.4: Valori di potenza man mano che ci si allontana dalla portante....48
Fig. 2.5.5: Drift in frequenza al variare della lunghezza dei pacchetti....48
Fig. 3.1.1: Spettro del segnale reale passa-banda e campionamento....53
Fig. 3.1.2: Schema di campionamento I&Q di un segnale passa-banda....54
Fig. 3.1.3: Segnale passa-banda moltiplicato per un coseno di frequenza fc....55
Fig. 3.1.4: Segnale passa-banda moltiplicato per un seno di frequenza fc....55
Fig. 3.1.5: Spettro della differenza dei segnali fase e quadratura....56
Fig. 3.1.6: Effetto sullo spettro della conversione Analogico/Digitale (A/D)....56
Fig. 3.2.1: Schema a blocchi della motherboard dell'USRP....58
Fig. 3.2.2: Un'immagine della motherboard dell'USRP....59

Fig. 3.2.3: Schema a blocchi dell'USRP end-to-end....59
Fig. 3.2.4: L'Universal Software Radio Peripherial....60
Fig. 3.2.5: Blocchi costituenti l'Analog to Digital Converter (ADC) dell'USRP....61
Fig. 3.2.6: Ruolo del Digital Down Converter in ricezione....61
Fig. 3.2.7: Struttura interna del DDC all'interno dell'FPGA....62
Fig. 3.2.8: Struttura interna dell'FPGA....62
Fig. 3.2.9: Filtro IIR (integratore)....63
Fig. 3.2.10: Filtro FIR (un "dente" del filtro a pettine)....63
Fig. 3.2.11: Filtro CIC, stadi di integratori (I) e filtri comb (c) in cascata....64
Fig. 3.2.12: Un esempio di filtro CIC interpolatore....65
Fig. 3.2.13: La daughterboard XCVR2450 (dual band 2.4-5.8 GHz)....67
Fig. 3.3.1: Ingresso e uscita dell'USRP: input RF+ADC+FPGA....68
Fig. 3.3.2: Il generatore di segnale utilizzato nelle misure....69
Fig. 3.3.3: Attenuatore Hewelett-Packard (0-110 dB, DC-18 GHz, 1W CW)....69
Fig. 3.3.4: Blocchi dello stadio di ricezione del campionatore I&Q (AD9862)....70
Fig. 3.3.5: FFT plot e valore istantaneo dei campioni per un segnale sinusoidale....71
Fig. 3.3.6: Caratteristica ingresso-uscita dell'USRP....73
Fig. 3.3.7: Caratteristica ingresso-uscita dell'USRP (saturazione)....74
Fig. 3.4.1: Quantizzazione della sinusoide in ingresso all'USRP2....75
Fig. 3.4.2: Caratteristica ingresso-uscita nell'USRP2....76
Fig. 3.4.3: Caratteristica ingresso-uscita nell'USRP2 (gain 0, 20, 40 dB)....77
Fig. 3.4.4: Energy detection con l'USRP2 (25 MHz) e relativo Noise Floor....78
Fig. 3.5.1: Struttura di un flow graph di GNUradio 79
Fig. 3.5.2: Uno screenshot di GNUradio Companion (GRC)....81
Fig. 3.5.3: Esempio di utilizzo di GNUradio companion....82
Fig. 3.5.4: Il blocco e le porte....85
Fig. 3.5.5: Componenti di un blocco GNUradio....86
Fig. 4.1: Posizione di 22 canali Bluetooth catturati in 25 MHz)....94
Fig. 4.1.1: Schema di un energy detector....99
Fig. 4.2.1: Trasmissione voce con pacchetti HV3 (TSCO=)....101
Fig. 4.3.1: Flow graph in GRC del setup per la cattura su file....103
Fig. 4.3.2: Spettro del segnale catturato dall'USRP2 (22 canali in 25 MHz)....103
Fig. 4.3.3: Short-term energy (N=250, overlap 50%, BW=25 MHz)....104
Fig. 4.3.4: Packet diagram relativo allo short-term energy diagram precedente....105
Fig. 4.3.5: Pacchetti dati da 1 slot....106
Fig. 4.3.6: Pacchetti dati da 3 slot....107
Fig. 4.3.7: Pacchetti dati da 5 slot....108
Fig. 4.3.8: Distribuzione statistica delle durate dei pacchetti dati (ACL)....109
Fig. 4.3.9: Distribuzione del tempo di interarrivo dei pacchetti (ACL link)....110
Fig. 4.4.1: Short-term energy in uno scenario di trasmissione voce (SCO)....111
Fig. 4.4.2: Pacchetti HV per link di tipo voce (SCO)....112
Fig. 4.4.3: Distribuzione della durata dei pacchetti per trasmissioni voce....113
Fig. 4.4.4: Distribuzione del tempo di interarrivo dei pacchetti (SCO link)....114
Fig. 4.4.5: Tempi di interarrivocaratteristici per una trasmissione voce (HV3)....115
Fig. 4.5.1: Recognition time nel caso di trasmissione dati simulata (ACL link)....116
Fig. 4.5.2: Short-term energy diagram (sensing time di 400 ms) su diverse bande, dall'alto: 1, 5, 10, 25 MHz....117
Fig. 4.5.3: Recognition time nel caso di trasmissione dati reale (ACL link)....118

Acronimi e abbreviazioni

ACL	Asynchronous Connection Less
ADC	Analog to Digital Converter
AFH	Adaptive Frequency Hopping
BD	Bluetooth Device
BR	Basic Rate
CAC	Channel Access Code
CR	Cognitive Radio
DAC	Digital to Analog Converter
DDC	Digital Down Converter
DIAC	Dedicated Inquiry Access Code
DSP	Digital Signal Processing
DSS	Dynamic Spectrum Sharing
DSSS	Direct Sequence Spread Spectrum
EDR	Enhanced Data Rate
EM	Elettro-Magnetico
ETSI	European Telecommunications Standards Institute
FCC	Federal Communications Commission
FFT	Fast Fourier Transform
FHS	Frequency Hopping Synchronization
FHSS	Frequency Hopping Spread Spectrum
FIR	Finite Impulse Response
FPGA	Field Programmable Gate Array
FSA	Fixed Spectrum Allocation
GFSK	Gaussian Frequency Shift Keying
GIAC	General Inquiry Access Code
GRC	GNU Radio Companion
GSM	Global System for Mobile communications
GU	Gazzetta Ufficiale
IEEE	Institute of Electrical and Electronics Engineers
IIR	Infinite Impulse Response
IP	Internet Protocol
IrDA	Infrared Data Association

ISM	Industrial Scientifical Medical
ITU	International Telecommunication Union
L2CAP	Logical Link Control and Adaptation Protocol
LAP	Lower Address Part
LMP	Link Manager Protocol
LOS	Line Of Sight
LSB/MSB	Least/Most Significant Bit
MAC	Medium Access Control
NAP	Non-significative Address Protocol
PHY	PHYsical layer
PPP	Point-to-Point Protocol
PSK	Phase Shift Keying
QDPSK	Quadrature Differential Phase Shift Keying
QoS	Quality of Service
RFCOMM	Radio Frequency COMMunication
SCO	Synchronous Connection Oriented
SDP	Service Discovery Protocol
SDR	Software Defined Radio
SIG	Special Interest Group
TCP	Transmision Control Protocol
TDD	Time Division Duplexing
UAP	Upper Address Protocol
UDP	User Datagram Protocol
UMTS	Universal Mobile Telecommunications System
UNII	Unlicensed National Information Infrastructure
USRP	Universal Software Radio Peripheral
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Network
WRAN	Wireless Regional Area Network

1 Introduzione

Negli ultimi anni, ovvero dalla pubblicazione del lavoro di Joseph Mitola III nel 1999 [1][1], è emersa sempre di più l'esigenza di sviluppare sistemi di comunicazione wireless in grado di trasmettere informazioni, gestendo l'utilizzo dello spettro radio in maniera intelligente.

Tra i primi a definire il concetto di Radio Cognitiva, si possono menzionare: J. Mitola III, Simon Haykin [3] e Bruce Fette [4]. All'origine di questa nuova tipologia di apparati, vi è la possibilità di sfruttare i recenti avanzamenti nel campo delle radio SDR, per creare un sistema di comunicazione in grado di riconoscere lo stato di occupazione dello spettro radio in tempo reale. Le radio SDR, essendo basate su hardware di tipo *general purpose* (FPGA) e su potenti algoritmi di *Digital Signal Processing* (DSP), hanno permesso di offrire le fondamenta ideali per lo studio e lo sviluppo di apparati di nuova generazione, le cosiddette Radio Cognitive (*Cognitive Radio*, CR).

L'impiego di tali dispositivi, in grado di analizzare l'evoluzione dello scenario radio nel tempo, permetterebbe l'introduzione di una gestione dinamica dello spettro radio (DSS), in grado di migliorare drasticamente l'efficienza di utilizzo dello spettro stesso. L'idea della CR ha accolto anche il favore dei più importanti organi di regolamentazione delle comunicazioni, come ad es. l'FCC [6].

Una prima applicazione della CR è stata studiata dal *working group* IEEE 802.22 e, nel Novembre 2004, ha portato alla proposta di standardizzazione di un sistema di accesso radio denominato Wireless Regional Area Network (WRAN) [8] in grado di garantire accesso wireless a banda larga in regioni estese del territorio (*rural areas*) e in ambito urbano sfruttando le frequenze del segnale TV inutilizzate (*white spaces*) [8]. In questo Standard le funzionalità di CR (*secondary users*) sono in grado di garantire il monitoraggio continuo dello spettro al fine di evitare problemi di interferenza nei confronti delle emittenti televisive (*primary users*).

Attualmente l'allocazione dello spettro radio è basata su una assegnazione delle bande di tipo statico (FSA). Ciascuna banda viene dedicata permanentemente ad uno specifico servizio e quindi agli utenti finali, che

1 La bibliografia è organizzata in capitoli. I richiami bibliografici indicati, ad esempio, con [1] si riferiscono al primo riferimento del corrente capitolo. In caso di richiami del tipo [1, Cap. 3] ci si riferisce al primo riferimento del capitolo 3.

nel gergo della CR vengono detti *primary users*. A seguito della rapidissima crescita del numero e della complessità, di tali servizi, si è verificato un rapido esaurimento dello spettro disponibile per nuove applicazioni. Parallelamente recenti studi hanno dimostrato che l'approccio FSA determina un inefficiente utilizzo dello spettro sia dal punto di vista temporale che spaziale. Basti pensare alle zone rurali, dove molti servizi hanno una copertura radio scarsa o nulla (ampie zone di inattività) e l'utenza è anch'essa ridotta (lunghi periodi di inattività). In queste aree è facile immaginare che le bande rimaste inutilizzate (ma allocate per specifici servizi), possano essere sfruttate per nuovi servizi (*secondary users*). Al fine di garantire il servizio ai *primary users*, si richiede tuttavia che i *secondary users* siano dotati di speciali dispositivi (radio cognitive) in grado di rilevare la presenza di utenti primari attivi nell'area di copertura.

Il funzionamento di un sistema di comunicazione basato sulla CR consiste, nel ripetere ciclicamente l'osservazione dello scenario radio, pianificare una azione, decidere, agire ed acquisire esperienza sugli scenari incontrati.

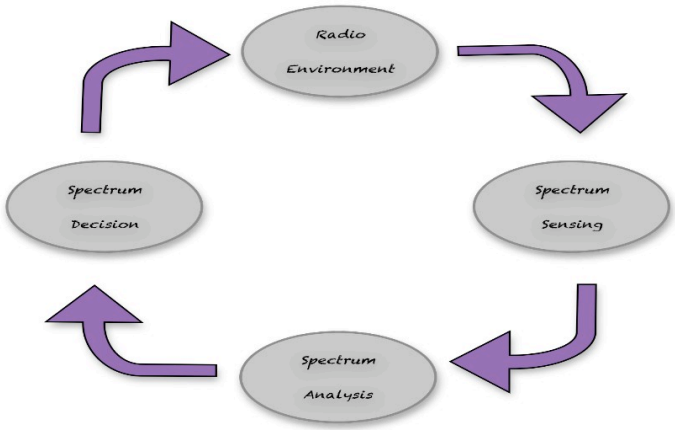


Fig. 1.1: Il ciclo di operazioni di una radio cognitiva

Questo lavoro di ricerca si inserisce nel contesto più ampio degli studi sulla CR con l'obbiettivo di analizzare e proporre soluzioni relative al primo *step* del ciclo cognitivo di una CR: l'osservazione dello scenario radio ovvero lo *spectrum sensing*.

Al fine di realizzare uno studio sperimentale su questo argomento, è stato scelto di studiare una tecnologia di larghissimo impiego, operante nella banda ISM a 2.4 *GHz*, che va sotto il nome di Bluetooth.

La tecnologia Bluetooth è rivolta ad offrire connettività a corto raggio

(*cable replacement*) mediante l'impiego di hardware semplice, ovvero di basso costo, e di ridotte dimensioni. L'analisi dello scenario radio in presenza di trasmissioni Bluetooth, è stata quindi condotta attraverso la radio SDR denominata Universal Software Radio Peripherial (USRP), con l'ausilio di strumenti di analisi sviluppati in MATLAB.

Il Bluetooth è stato studiato con particolare attenzione alle caratteristiche di livello MAC (durata dei pacchetti, *patterns* di scambio dei pacchetti, ecc.), al fine di estrapolare da esse alcune *features* (specifiche della tecnologia Bluetooth) mediante un Energy Detector (ED).

La scelta di impiegare l'ED è seguita a un attento studio di altre tecniche utili all'estrazione di *signal features*, come ad esempio: il riconoscimento della modulazione basato su FFT [6, Cap. 4], basato sul computo degli attraversamenti per lo zero (*zero-crossing*) [5, Cap. 4] e le tecniche basate sulla trasformata congiunta tempo-frequenza [7, 8, Cap. 4]. L'impiego dell'energy detector ha offerto il miglior compromesso tra semplicità realizzativa, flessibilità ed efficacia, come dimostrato in [1, Cap. 4].

Questo lavoro si inquadra in un progetto più ampio, ovvero la creazione di un *framework* per il riconoscimento e la classificazione *multi-standard*, basato su caratteristiche relative alla durata e ai *pattern* di scambio dei pacchetti. L'obiettivo finale è quello di realizzare un sistema di classificazione wireless multi-standard basato su piattaforma radio SDR chiamato AIR-AWARE (Fig. 1.2). In particolare la classificazione è rivolta a diverse tecnologie wireless operanti nella banda di 80 *MHz* centrata attorno a 2.4 *GHz* e detta ISM (*Industrial Scientific Medical*), come: WiFi, Bluetooth, ZigBee.

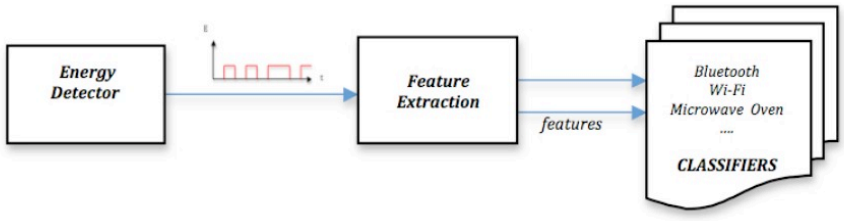


Fig. 1.2: Il modulo AIR-AWARE (ED, Feature Extractor e Classificatore)

La classificazione di tecnologie wireless, basata su *features* di livello MAC, ha già dimostrato [1, Cap. 4] buoni risultati nel caso di coesistenza tra WiFi (traffico reale) e Bluetooth (traffico simulato). Al finedi ottenere dati sul Bluetooth in condizioni di traffico reale, si sonodovute analizza-

re le attuali difficoltà nella cattura di pacchetti Bluetooth attraverso tecniche di *packet sniffing*. Infatti, a differenza che nel WiFi (IEEE 802.11 b/g), in Bluetooth si impiegano 79 canali in modalità FHSS attraverso codici di hopping diversi per ogni rete. Altra caratteristica è che il gli strati di livello 1 e 2 risultano incapsulati da un interfaccia detta HCI (*Host-Controller Interface*). Questa permette un controllo limitato del dispositivo se non attraverso pochi comandi (HCI *commands*) che non includono una modalità "monitor" simile a quella presente in WiFi (vedi anche APPENDICE A). La radio SDR impiegata ha permesso di aggirare questi problemi e di fornire rapidamente tutti i dati necessari all'estrazione di *features* da traffico reale. Una volta esplorate le possibilità offerte dalla radio SDR, è stata scelta una piattaforma valida all'implementazione del modulo AIR-AWARE, la radio Universal Software Radio Peripheral (USRP).

Da uno studio attento della tecnologia Bluetooth, si è deciso di proporre alcune *features* di livello MAC, facilmente estraibili mediante l'ED. Il lavoro che ne è seguito è stato di verifica della validità di tali *features*. Attraverso l'analisi nel dominio del tempo (l'*energy detection* appunto) del segnale ricevuto dall'USRP, si sono potuti ottenere *timestamp* e durata di tutti i pacchetti Bluetooth rilevati in condizioni di traffico reale (*packet diagram*). Attraverso questi dati è stato possibile verificare l'esistenza e la validità delle *features* proposte in condizioni di traffico Bluetooth reale.

I risultati del presente lavoro di ricerca sono anche descritti nel seguente paper:

Sergio Benco, Stefano Boldrini, Andrea Ghittino, Stefano Annese, Maria-Gabriella Di Benedetto, "Identification of packet exchange patterns based on energy detection: the Bluetooth case",

che verrà presentato in occasione del workshop di carattere internazionale CogART 2010 (*3rd International Workshop on Cognitive Radio and Advanced Spectrum Management, November 8-10, 2010, Rome, Italy*).

Questa parte introduttiva descriverà brevemente la banda ISM ovvero la porzione di spettro nella quale operano il Bluetooth e altre importanti tecnologie wireless come il WiFi e ZigBee. Successivamente si introdurranno alcuni concetti tipici del paradigma CR, utili alla comprensione di quanto segue.

1.1 La banda ISM

La banda che va sotto il nome di ISM (Industrial Scientific Medical band) consiste in un insieme di 3 bande (0.9, 2.4, 5.8 *GHz*) definite in ambito ITU e di larghezza rispettivamente 26, 83.5 e 125 *MHz*. Tali bande risultano tra quelle non sottoposte a licenza (*unlicensed bands*) e quindi il loro impiego è libero per tutte le tecnologie radio che rispettino i vincoli di emissione a radiofrequenza.

Il libero accesso alla banda ISM ha determinato la proliferazione di sistemi di comunicazione mobile in grado di connettere laptop, netbook, Personal Digital Assistant (PDA), e dispositivi indossabili (*wearable computers*). Per le applicazioni a corto raggio si sono sviluppate le cosiddette Wireless Personal Area Networks (WPAN), definite dallo Standard IEEE 802.15.1 [1]. Le WPAN permettono la condivisione di informazioni e risorse tra dispositivi a breve distanza (da pochi centimetri a qualche metro). Nel medio-lungo raggio e per bitrate più elevati si sono sviluppate le Wireless Local Area Networks (WLAN, Standard IEEE 802.11) delle quali il WiFi rappresenta l'implementazione più diffusa.

Le sfide maggiori nello sviluppo di tali tecnologie nella banda ISM derivano dalla necessità di sviluppare ricetrasmettitori in grado di operare efficientemente in scenari radio ostili caratterizzati da una notevole interferenza intra-sistema ed inter-sistema. Inoltre, in questa stessa porzione di spettro, spesso vengono a trovarsi altri segnali (provenienti da radiatori intenzionali e non) come ad esempio quelli generati da dispositivi per il comando a distanza (*remote openers* ovvero radiocomandi per cancelli automatici) oppure quelli generati dai comuni forni a microonde.

Lo spettro a radio frequenza nel corrente scenario di allocazione statica risulta essere una risorsa in rapido esaurimento. Il compito di decidere quali bande assegnare ai vari servizi di telecomunicazioni spetta a specifici organi internazionali di controllo. Quest'ultimi approvano disposizioni che devono poi essere recepite dai singoli paesi del mondo. In molti casi il tentativo di armonizzare le esigenze nazionali (spesso riguardanti le applicazioni militari) con le regole internazionali, ha portato a parziali modifiche di quanto deciso in sede internazionale.

Il risultato è che in alcuni paesi del mondo (as es. Francia, Spagna, Giappone) sussistono differenze sostanziali di allocazione dello spettro, le quali a loro volta, hanno l'effetto di ridurre la compatibilità tra dispositivi e infrastrutture (ad es. come per le bande impiegate nei sistemi di telefonia cellulare UMTS e GSM).

Un esempio che evidenzia la rigidità dell'attuale schema di allocazione fissa, è dato dal caso delle bande scelte per le comunicazioni WiMax (IEEE 802.16). Pur risultando allocate diverse bande per tale tecnologia, questa risulta tutt'oggi di scarso impiego sul territorio. Al suo posto si preferisce continuare ad impiegare una tecnologia largamente diffusa come il WiFi. L'effetto è che sebbene le bande WiMax risultino pressoché inutilizzate queste non possono di fatto essere impiegate per altri servizi non autorizzati.

Una possibile soluzione ai problemi di impiego inefficiente dello spettro radio viene dallo sviluppo di radio intelligenti in grado di ascoltare lo spettro radio, evidenziare porzioni inutilizzate e sfruttarle. Tali radio permetterebbero la transizione da uno scenario di allocazione fissa (FSA), a uno più efficiente di gestione dinamica dell'allocazione e dell'accesso (*Dynamic Spectrum Allocation*, DSA e *Dynamic Spectrum Sharing*, DSS). Gli organi di controllo più importanti che si occupano di regolamentare l'accesso alle bande radio sono l'FCC, l'ITU e l'ETSI.

Lo spettro radio oggetto di regolamentazione, ovvero quello impiegabile per comunicazioni wireless, può dirsi compreso tra pochi KHz (Ultra Low Frequencies - ULF) a circa 300 *GHz* (Extremely High Frequencies – EHF). Attualmente l'accesso a tale spettro è regolato concedendo opportune licenze di utilizzo agli operatori privati (anche attraverso gare di assegnazione) o agli enti pubblici (civili o militari). Uno dei compiti dell'organo denominato Unione Internazionale delle Telecomunicazioni (ITU) è proprio quello di regolare l'utilizzo di bande di frequenze nello spettro radio a livello mondiale. Nell'ambito di tale regolamentazione (ITU World Radiocommunication conference), è stato deciso di allocare alcune porzioni dello spettro per applicazioni industriali, scientifiche e mediche (ISM). Tale assegnazione era già presente sotto lo stesso nome negli Stati Uniti presso l'organo di regolamentazione denominato Federal Communication Committee (FCC). L'ITU ha adottato la ripartizione decisa dall'FCC. Le bande ISM definite a livello mondiale (ITU) sono quelle che seguono.

ISM 900 *MHz* (902-928 *MHz*, banda 26 *MHz*)
ISM 2.4 *GHz* (2.400-2.4835 *GHz*, banda 83.5 *MHz*)
ISM 5.8 *GHz* (5.725-5.850 *GHz*, banda 125 *MHz*)

La banda attorno ai 5.8 *GHz*, negli USA, è denominata banda UNII. Le bande ISM sono definite nel documento ITU-T Radio Regulations ai punti S5.138 e S5150. A ciascun singolo paese, per mezzo di organi nazionali, spetta il compito di armonizzare il proprio piano di ripartizione delle frequenze con le suddette linee guida ITU o FCC. In Italia, il Ministero dello Sviluppo Economico (Dipartimento per le Comunicazioni) definisce il Piano Nazionale di Ripartizione delle Frequenze (PNRF), il quale costituisce un vero e proprio piano regolatore dell'utilizzo dello spettro radioelettrico in Italia. Dal sito del Ministero, si legge che lo scopo del Piano è di stabilire, in ambito nazionale:

"...l'attribuzione ai diversi servizi delle bande di frequenze oggetto del Piano, di indicare per ciascun servizio, nell'ambito delle singole bande, l'autorità governativa preposta alla gestione delle frequenze, nonché le principali utilizzazioni civili di verificare l'efficiente utilizzazione dello spettro, al fine di liberare risorse per il settore televisivo e di gestire al meglio gli eventuali contenziosi con i Paesi frontalieri."

Il Piano attualmente in vigore, che concerne le bande di frequenze comprese tra 0 e 1000 *GHz*, è stato approvato con decreto ministeriale del 13 novembre 2008 e pubblicato nella GU n. 273 del 21-11-2008 (Suppl. Ordinario n. 255). Lo scopo di tale decreto è stato quello di recepire nella normativa nazionale alcune decisioni della WRC (*World Radio Conference*), tra cui quelle riguardanti la radiodiffusione televisiva in tecnica digitale, nonché una serie di decisioni della Commissione Europea in merito alla gestione dello spettro radioelettrico. Nel PNRF, in merito alle bande ISM, è possibile leggere quanto segue:

"I servizi di radiocomunicazione operanti in queste bande devono accettare i disturbi pregiudizievoli che possono verificarsi a causa delle citate applicazioni. Ogni misura praticamente possibile deve essere adottata per assicurare che le irradiazioni delle apparecchiature usate per tali applicazioni siano minime e che al di fuori della banda il livello delle irradiazioni sia tale da non causare disturbi pregiudizievoli ai servizi di radiocomunicazione ed in particolare alla radionavigazione e ad ogni altro servizio di sicurezza operante in accordo con le prescrizioni del presente piano."

L'esigenza di dover accettare possibili disturbi nelle bande non sottoposte a licenza è data dal fatto che in tali bande possono essere impiegate contemporaneamente diverse tecnologie (ad es. WiFi, Bluetooth, ZigBee ecc.). Tale scenario è alla base del problema dell'interferenza mutua generata tra questi sistemi di comunicazione.

1.2 La Radio Cognitiva e la SDR

Il termine Radio Cognitiva (Cognitive Radio, CR) è stato introdotto per la prima volta in un articolo di Joseph Mitola III e Gerald Q. Maguire, Jr. pubblicato nel 1999 [1]. Il termine "*Radio*" sta ad indicare un sistema di comunicazione in grado di trasmettere e/o ricevere informazioni per mezzo di apparati elettronici di ricetrasmissione a radiofrequenza (RF) di natura continua (Continuos Wave) o anche impulsiva (impulse radio). Il termine "*Cognitive*", invece, rappresenta un nuovo approccio per la progettazione di sistemi di comunicazione wireless rivolto a introdurre dispositivi cosiddetti "intelligenti", in grado di cambiare i parametri della trasmissione o ricezione in modo dinamico e possibilmente in modo *seamless*. Le nuove radio di tipo cognitive potrebbero consentire di poter comunicare, in modo affidabile, in scenari caratterizzati dalla presenza di più sistemi di comunicazione eterogenei (non interoperanti) mantenendo bassa l'interferenza intersistema [2] [3] [4] [5].

La CR può essere impiegata in 4 scenari [6]: CR per reti primarie in bande con licenza (*Licensed Networks*), CR con reti secondarie in bande con licenza (*Secondary Markets*), CR coordinato tra reti in bande con licenza (*Coordination of Licensed Operation*), CR in bande inutilizzate (con o senza licenza).

Nel primo scenario (*Licensed Networks*) si prevede che il servizio avente diritto di operare in una data banda sottoposta a licenza introduca delle funzionalità di radio cognitiva e, più in generale, di rete cognitiva. Tali tecniche vengono impiegate all'interno della suddetta rete con l'obbiettivo di incrementare l'efficienza nell'utilizzo dello spettro a disposizione. In tal contesto è possibile introdurre, in genere per mezzo di radio SDR, funzionalità di *Adaptive Modulation*, *Dynamic Frequency Selection*, *Transmit Power Control*. Ciascuna di queste o altre tecniche è resa possibile dalla notevole flessibilità offerta dall'impiego di radio SDR.

Nel secondo scenario (*secondary markets*), si prevede una rete primaria con determinate garanzie di servizio e una rete secondaria in grado di impegnare la banda solo dove e quando questa sia inutilizzata dal servizio primario. Ciò implica la necessità di definire un accordo tra i gestori delle reti primaria e secondaria al fine di regolare l'accesso degli utenti afferenti alla rete secondaria. Un esempio di un simile scenario è dato dai servizi di emergenza (pronto intervento, soccorso, ecc.) considerabili come servizi primari e, ad esempio, servizi di trasmissione dati senza requisiti di QoS (*secondary networks*). Tale approccio consente un'elevata

efficienza e flessibilità nell'uso dello spettro radio nonché notevoli possibilità di business per gli operatori di telecomunicazioni.

Nel terzo scenario (*Coordination of Licensed Operation*) si prevedono diverse reti di pari priorità che operino in banda sottoposta a licenza. In tal caso diviene necessaria una qualche forma di coordinamento tra le reti che può essere implementata in diversi modi. L'obbiettivo è quello di garantire *throughput* elevati mantenendo quanto più bassa possibile l'iterferenza mutua tra dispositivi afferenti alle diverse reti.

Un ulteriore scenario (*Non-voluntary third party access*) prevede la creazione di reti cognitive in grado di operare in banda non sottoposta a licenza e lì dove, pur in presenza di banda occupata da altro servizio, si abbia una condizione di inutilizzo (zone non coperte dal servizio primario, ad es. i cosiddetti *white spaces* per il segnale TV). Tale ultima tipologia di scenario è molto interessante se si pensa che da diversi studi risulta che gran parte dello spettro allocato risulta inutilizzato in vaste zone del territorio e per tempi molto lunghi. In tali aree l'impiego di radio capaci di rilevare in modo opportunistico porzioni dello spettro non utilizzate, consentirebbe di creare reti ad elevato *bitrate* senza costi eccessivi per gli operatori.

La Software Defined Radio (SDR) rappresenta una nuova tipologia di radio che presenta gran parte dei suoi blocchi costituenti (idealmente tutti) definiti attraverso moduli software. Tale architettura è resa possibile dall'impiego di hardware riprogrammabile (FPGA), in grado di sostenere elevate velocità di campionamento/ricostruzione ed una buona precisione (*Analog to Digital Converters* e *Digital to Analog Converters*, ADCs e DACs).

Attraverso tali architetture *general purpose* e riprogrammabili, si può ottenere una elevatissima versatilità tale da permettere di modificare i parametri di modulazione, codifica, ecc. in modo dinamico. E' chiaro che la SDR sia considerabile come una tecnologia abilitante per il paradigma Cognitive Radio.

In questa direzione la ricerca sta concentrando da tempo gli sforzi per migliorare ed ampliare le capacità della SDR al fine di realizzare una radio interamente digitale. Un simile oggetto renderà possibile realizzare dispositivi multi-standard (in parte già sviluppati in ambito militare) in grado di scegliere dinamicamente il sistema di trasmissione da impiegare in un dato momento in base alla disponibilità di risorse, al livello di interferenza rilevato oppure al gusto dell'utente.

Ciò consentirà di ridurre i costi per i produttori (tempo di vita maggiore

per i dispositivi grazie ad aggiornamenti software) ed offrire maggiori opportunità di business per gli operatori attraverso applicazioni in grado di sfruttare reti di diversa natura. Infine, per gli utenti finali, l'SDR sarà in grado di garantire un'elevata versatilità d'impiego offrendo dispositivi *all-in-one* (cellulare, cordless, WiFi, Bluetooth in un unico blocco ricevitore) in grado di cambiare caratteristiche operative in modo dinamico e intelligente.

In questo lavoro si è scelto di studiare una possibile applicazione della radio SDR nell'ambito delle tecnologie operanti in banda ISM, in particolare, si è scelto di studiare la tecnologia wireless Bluetooth.

Il problema della coesistenza tra Bluetooth e altri sistemi di largo impiego in banda ISM 2.4 *GHz* (WiFi, ZigBee, ecc.) è ritenuto molto importante per le ripercussioni dirette che ha nella fruizione di tali tecnologie. Tale attenzione è dimostrata anche dal fatto che il *working group IEEE 802.15.2* ha studiato ed emesso un documento dal titolo "Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands " [7] nel quale vengono proposte alcune soluzioni. Tale lavoro spiega quali possano essere le strategie attuabili per permettere la coesistenza del Bluetooth (reti WPAN) con il WiFi (IEEE 802.11b/g, reti WLAN).

Qui vengono distinti 2 scenari principali. Il primo scenario prevede che le radio Bluetooth e WiFi siano nel medesimo *host* e possano comunicare e coordinarsi tramite BUS interno. Un secondo scenario prevede, invece, che le radio siano indipendenti e che debbano essere impiegate specifiche tecniche di spectrum sensing (PHY) oppure di livello MAC.

Il capitolo seguente tratterà in generale le caratteristiche della tecnologia wireless Bluetooth attingendo le informazioni da due importanti documenti: lo Standard IEEE [1, Cap 2] e le specifiche tecniche del Bluetooth SIG [2, Cap 2]. Le indispensabili conoscenze acquisite da questo studio hanno permesso di identificare alcune *features* proprie di questa tecnologia e che verranno poi illustrate e commenatate nell'ultima parte di questo lavoro.

2 La tecnologia Bluetooth

Le specifiche tecniche del sistema Bluetooth sono state ideate e sviluppate da Ericsson e in seguito formalizzate dal Bluetooth Special Interest Group (SIG). Il Bluetooth Special Interest Group (SIG) è un consorzio di aziende operanti in diversi settori dell'industria ICT fondato nel settembre del 1998. Tra queste vi sono: Ericsson, IBM, Intel, Toshiba, Nokia, Lenovo, Microsoft, Motorola e altre società che si sono poi aggiunte all'elenco come associate o come membri aggiunti. I membri del Bluetooth SIG (oggi circa 3000 aziende) si prefiggono di guidare lo sviluppo di applicazioni basate sulla tecnologia wireless Bluetooth e di promuoverne la produzione e il commercio. L'obbiettivo di questo gruppo è essenzialmente quello di pubblicare le specifiche della tecnologia Bluetooth, validare nuove applicazioni, proteggere il marchio Bluetooth e diffondere tale tecnologia. Il quartier generale del Bluetooth SIG è in Kirkland (Washington, USA).

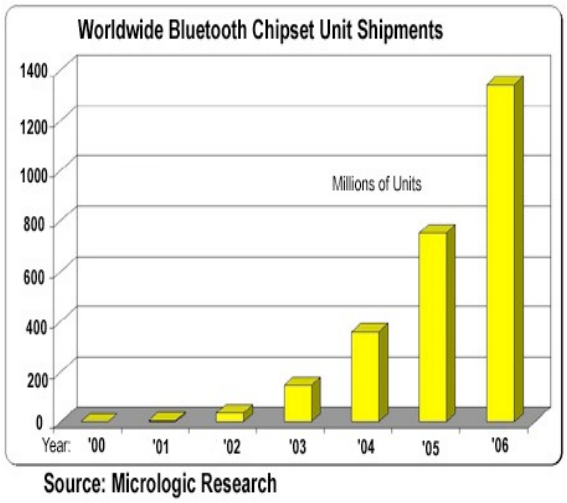


Fig. 2.1: Andamento mondiale delle vendite di chipset Bluetooth

Il sistema Bluetooth è descritto da due principali documenti: lo Standard IEEE 802.15.1 (14 Giugno 2005) per le WPAN [1] e il documento "Bluetooth Core Specification" giunto alla versione 4 (17 Dicembre 2009) pubblicato dal Bluetooth SIG [4]. Come si può notare dalle date di pubblicazione, il documento dello Special Interest Group (SIG), risulta il più aggiornato ed in esso sono descritte le specifiche del Bluetooth Enhanced

Data Rate (EDR) versioni 2.0 e 3.0 (2 Mb/s e 3 Mb/s fino a 24 Mb/s per la versione 3 High Speed che impiega l'802.11 per i dati e l'802.15.1 per il controllo). Inoltre nell'ultimo documento del SIG si introduce una nuova versione del Bluetooth denominata Bluetooth Low Energy Technology.

Fig. 2.2: Il logo Bluetooth

Il nome "Bluetooth" è stato adottato alla costituzione del Bluetooth SIG (1998). Le sue origini derivano dal nome del re danese Harald Blåtand (o Harold Bluetooth in inglese), il quale fu, nel 940-981 d.C., un abile diplomatico che riuscì ad unire, sotto il suo regno, i popoli scandinavi, introducendo nella regione il cristianesimo. Similmente, il Bluetooth avrebbe dovuto permettere la collaborazione tra industrie produttrici provenienti da settori molto diversi tra loro (informatica, telefonia, settore auto, ecc.). Il logo della tecnologia unisce le rune nordiche:

Hagall

Berkanan

corrispondenti alle nostre lettere H e B, le iniziali, appunto, di re Harald Blåtand (Aroldo I di Danimarca). Nelle pagine seguenti si descrivono le principali caratteristiche del sistema Bluetooth. Si è volutamente scelto di focalizzare l'attenzione sulle caratteristiche di strato fisico, al fine di poter approfondire la conoscenza del segnale Bluetooth e poter facilitare la comprensione dell'algoritmo di identificazione.

2.1 Descrizione generale

La tecnologia Bluetooth opera nella banda ISM (Industrial Scientific Medical) intorno a 2.4 *GHz* (2.400 *GHz* – 2.483,5 *GHz*), impiegando gli 80 *MHz* a disposizione, mediante la tecnica a spettro espanso detta Frequency Hopping Spread Spectrum (FHSS). Tale tecnica consiste nel definire una serie di canali (79 canali) a banda stretta (1 *MHz*) e un codice di tipo pseudo-noise (Frequency Hopping Code, FH code) che regola l'accesso ai canali. Una radio in modalità FHSS salta (hop) periodicamente con un ritmo di 1600 hop/sec. Mediamente, in tal modo, il segnale Bluetooth (~1 *MHz*) subisce uno spreading della banda fino a circa 80 *MHz* (79 canali da 1 *MHz*, con bande di guardia di 2 *MHz* per il limite inferiore e 3.5 *MHz* per il limite superiore della ISM).

Lo Standard IEEE 802.15.1 per le WPAN, definisce le specifiche tecniche per gli strati fisico (*physical layer*) e di controllo di accesso al mezzo radio (*MAC layer*). In tale Standard si fa riferimento al Bluetooth ma, in realtà, in esso sono dettate regole valide per qualunque tecnologia wireless dedicata alle *Wireless Personal Area Networks* (WPANs).

L'obbiettivo dello Standard IEEE 802.15.1, inizialmente, era anche quello di trovare una soluzione al problema dell'interoperabilità (scambio di dati tra sistemi basati su tecnologie diverse) tra un dispositivo WPAN e uno con tecnologia IEEE Std 802.11 (*Wireless Local Area Network*, WLAN). Tale obiettivo però si è dimostrato infattibile, per cui ad oggi, nello Standard IEEE per le WPAN, non si prevede ancora l'interoperabilità.

In questa ricerca, si è scelto di concentrare l'attenzione sulla tecnologia Bluetooth. La scelta di un particolare sistema di comunicazione (di largo impiego) consente, da una parte, di evidenziare problematiche reali (intereferenza, inefficienze varie, ecc) di quella specifica tecnologia e, da un'altra, di mostrare come la Software Defined Radio (SDR), e più in generale i princìpi della Radio Cognitiva, possano essere introdotti in tali contesti. Attraverso questi nuovi strumenti, si vuole introdurre una metodologia per la risoluzione di problematiche relative alla coesistenza tra tecnologie diverse come appunto il Bluetooth e il WiFi.

2.2 Standard IEEE 802.15.1 e le specifiche del Bluetooth SIG

Il Bluetooth è un sistema di comunicazione wireless che permette la comunicazione a breve distanza (~0.1m Classe 3, ~10m Classe 2, ~100m Classe 1) tra dispositivi, consentendo quindi di porsi come alternativa all'uso di cavi di connessione per dati (cable replacement). Le caratteristiche che contraddistinguono questo sistema sono: robustezza, potenze ridotte e bassi costi. Il Bluetooth si pone come valida alternativa alla tecnologia IrDA che impiega la trasmissione per mezzo di radiazione EM ad infrarossi. Il punto debole di questo sistema è la necessità di essere il Line Of Sight (LOS) tra ricevitore e trasmettitore. L'impiego nel Bluetooth di onde radio in banda ISM ha superato questo ostacolo mantenendo semplici (ricetrasmettitori a banda stretta) ed economici (~5€) i dispositivi.

La tecnologia Bluetooth è organizzata in diversi sottosistemi. Le parti fondamentali (descritte negli Standard e nei documenti SIG) vanno sotto il nome di core system, per indicare l'insieme di: modulo radio a radiofrequenza (RF), moduli operanti sul segnale in banda base (BB), e stack protocollare.

Il canale di comunicazione instaurato da un device Bluetooth offre in Basic Rate un bitrate lordo di 1Mb/s. Il dettaglio delle possibili configurazioni (valori di bitrate netti) dei link Bluetooth è indicato di seguito.

Configuration	Max. datarate Upstream	Max. datarate Downstream
up to 3 SCO channels	(PCM) 64 kb/s	(PCM) 64 kb/s
Symmetric data link	433.9 kb/s	433.9 kb/s
Asymmetric data link	up to 723.2 kb/s	up to 723.2 kb/s

I canali voce (Synchronous Connection-Oriented, SCO) vengono definiti per mezzo di commutazione a circuito attraverso un meccanismo di slot reservation ad intervalli fissati. I canali dati (Asynchronous Connection-

Less, ACL) sono realizzati attraverso commutazione a pacchetto regolata da uno schema di accesso a polling. Vengono definiti anche link di tipo voce+dati SCO. Tali collegamenti permettono di offrire 64kb/s di dati e 64Kb/s di traffico voce in entrambi le direzioni.

Il Bluetooth è una tecnologia aperta e quindi la relativa documentazione è liberamente accessibile. I documenti principali che la descrivono sono l'IEEE 802.15.1 e il Bluetooth SIG Core Specifications. Alcuni dettagli implementativi vengono lasciati all'Industria consentendo, così, una più rapida penetrazione di questa tecnologia nel mercato consumer al quale, principalmente, si rivolge. In tali documenti è descritto ogni aspetto della tecnologia, dall'architettura, al dettaglio di ogni singolo strato dello stack, ai servizi (profili Bluetooth). Il risultato di ciò è che la documentazione (Standard e SIG Technical Specifications) è eccezionalmente ampia (~600 e ~1600 pagine rispettivamente). Tale aspetto, se da una parte consente una formazione gratuita completa sulla tecnologia, dall'altra prevede una curva di apprendimento piuttosto lenta. Tuttavia, per l'applicazione di interesse in questo lavoro, è bastato concentrarsi sulla descrizione dello strato fisico, sulla struttura dei pacchetti e sulla modulazione del segnale.

L'architettura del sistema Bluetooth prevede una struttura protocollare a stack. Lo stack Bluetooth riprende la suddivisione dei livelli ISO/OSI definendo una serie di moduli che implementano tali livelli dallo strato fisico (livello 1) allo strato di collegamento (livello 2) composto, a sua volta, da MAC (Medium Access Control) e LLC (Logical Link Control).

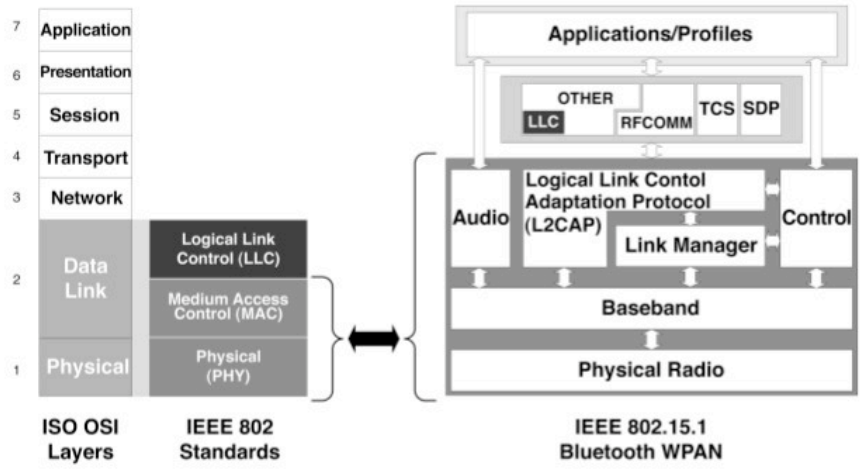


Fig. 2.2.1: Parallelo tra gli strati ISO/OSI e lo stack Bluetooth

Le varie componenti dello stack Bluetooth comunicano per mezzo di protocolli sia di tipo nativo (LMP, L2CAP, RFCOMM, SDP) che preesistenti (PPP, IP, TCP, UDP, ecc.). Lo schema seguente (dal sito del Bluetooth SIG) riprende la suddivisione in moduli dello stack Bluetooth ed indica i bus di segnalazione per i vari protocolli coinvolti. Si evidenzia, inoltre, la presenza di un elemento che racchiude i moduli di strato fisico e di collegamento, che va sotto il nome di Controller Bluetooth. Tale elemento nelle implementazioni è fatto corrispondere, ad esempio, all'adattatore USB Bluetooth. Tali oggetti, i quali permettono di abilitare la ricetrasmissione di segnali Bluetooth su elaboratori sprovvisti di moduli radio integrati, sono molto comuni. In questo tipo di dispositivi, il suddetto Controller può essere realizzato anche in single chip.

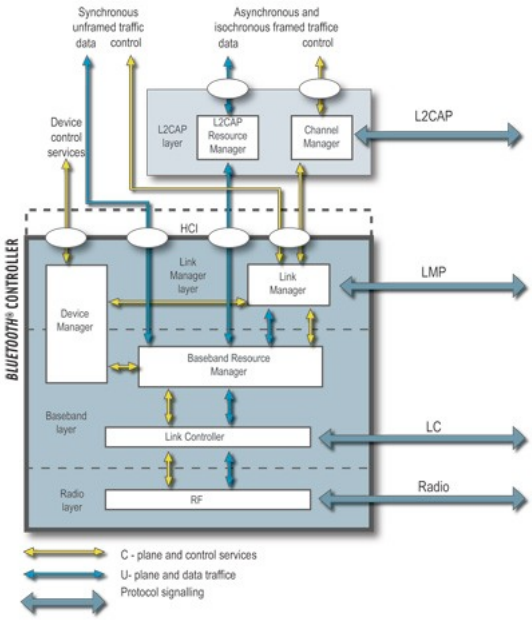


Fig. 2.2.2: Protocolli impiegati dai diversi livelli dello stack Bluetooth

Nel Bluetooth si prevede la suddivisione della banda ISM a 2.4 *GHz* in 79 canali da 1 *MHz*. Il Bluetooth impiega la tecnica di Spread Spectrum denominata Frequency Hopping Spread Spectrum (FHSS) come modulazione secondaria. La modulazione primaria è data da uno schema GFSK (Gaussian Frequency Shift Keying) operante ad una velocità di simbolo (baud rate) pari a 1MS/s il quale, in Basic Rate, supporta un data rate di 1Mb/s. I dispositivi Bluetooth sono in grado di cambiare frequenza portante (frequency hopping) ad un ritmo di 1600 hop/s seguendo un pattern pseudo-random. Tale pattern è generato a partire dal clock e l'indi-

rizzo MAC di un nodo, che assume il ruolo di coordinatore, detto Master.

L'hopping pattern può essere adattato (Adaptive Frequency Hopping, AFH) al particolare scenario di impiego mediante l'esclusione di canali che risultino affetti da elevata interferenza. Tale tecnica è stata sviluppata per permettere una migliore coesistenza (IEEE Std 802.15.2 [7]) del Bluetooth con sistemi wireless come il WiFi (IEEE Std 802.11 b,g) caratterizzati dall'impiego di canali fissi. Nel WiFi, infatti, si possono raggiungere (annex g) elevati bitrate (54 Mb/s) impiegando canali di circa 20MHz con tecnica DSSS (Direct Sequence Spread Spectrum).

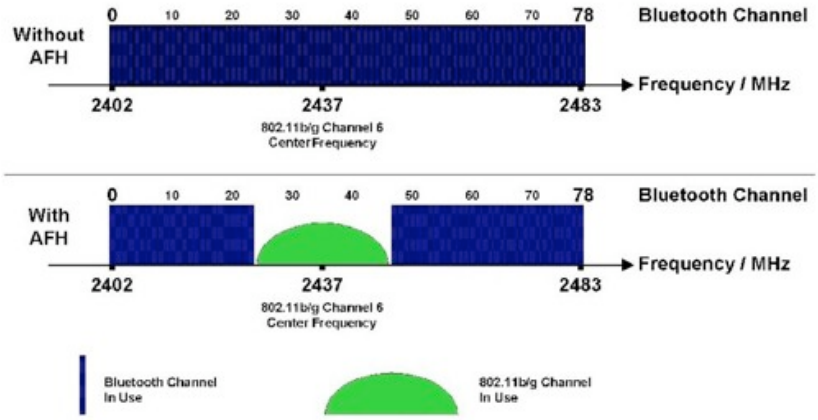


Fig. 2.2.3: La ripartizione dei canali fisici in Bluetooth e l'AFH

I dispositivi connessi in una WPAN Bluetooth sono raggruppati in piconet le quali possono ospitare fino a 8 nodi (compreso il Master). Ogni piconet possiede un nodo con il ruolo di Master che offre il riferimento per la sincronizzazione dei nodi della piconet. Gli altri dispositivi (da 1 a 7 per ciascuna piconet) vengono detti Slave.

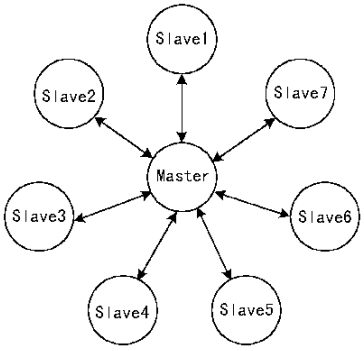


Fig. 2.2.4: Topologia di rete di una piconet

Più in generale le topologie possibili di rete WPAN sono 3. La prima prevede la comunicazione tra un Master e un solo Slave (ad es. applicazioni Client-Server, connessioni tra auricolare e cellulare). La topologia di base è, quindi, la piconet ospitante fino a 7 nodi attivi ma, oltre a questa, si può configurare quella che viene detta Scatternet, ovvero l'interconnessione di più piconet. Tale topologia si realizza attraverso nodi che fungono da Master e Slave, a seconda della Piconet di appartenenza, in modo da offrire funzionalità di bridging tra due WPAN diverse (vedi Fig. 2.2.5, esempio c).

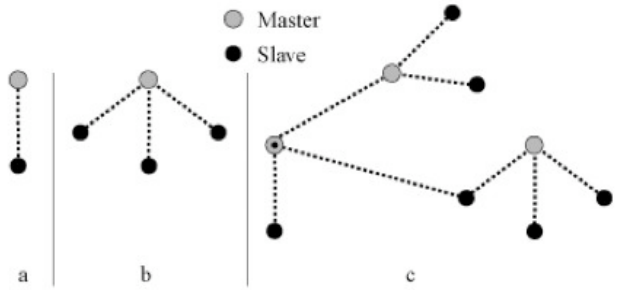


Fig. 2.2.5: Topologie di rete Bluetooth: Master-Slave, Piconet, Scatternet

Il canale fisico Bluetooth viene utilizzato per mezzo di una trasmissione organizzata in slot (Time Division Duplex, TDD). I dati in transito tra Master e Slave, infatti, vengono inviati alternando le comunicazioni da Master a Slave e viceversa. In caso di pacchetti lunghi più di uno slot si consente la trasmissione multislot (pacchetti da 1, 3, o 5 slot consecutivi). I salti da un canale all'altro vengono così a trovarsi sempre tra un invio e una ricezione completa.

Tale caratteristica permette di fare una considerazione. Per quanto detto, se si volessero catturare tutti i pacchetti di una trasmissione dati tra Master e Slave, sarebbe necessario conoscere il Frequency Hopping pattern così da poter seguire l'intera comunicazione. Se, tuttavia, si fosse interessati a catturare un sotto insieme dei pacchetti inviati e ricevuti in quella trasmissione, potrebbe essere sufficiente soffermarsi su un singolo canale Bluetooth (1 di 79 canali, largo $1MHz$) e raccogliere tutti i pacchetti (si ricordi il ritmo di 1600 hop/s) inviati su quel dato canale. Ad esempio, in 1 secondo in quel canale, (connessione con pacchetti da un solo slot) transiterebbero fino a 20 pacchetti (1600/79). Pensando, inoltre, di ascoltare in parallelo più canali si potrebbe riuscire a catturare una serie di pacchetti che possa essere considerata "rappresentativa" della comunicazione in atto e che quindi possa permetterne l'identificazione. Tale approccio è quello seguito in questo lavoro.

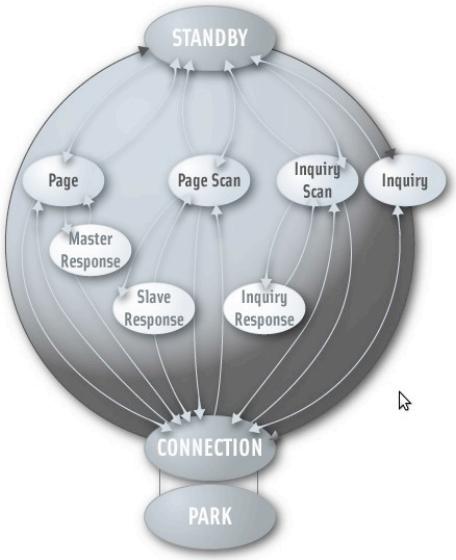


Fig. 2.2.6: Stati di funzionamento di un dispositivo Bluetooth

I device Bluetooth possono trovarsi in 3 stati principali (Connection, Standby e Park) e 7 sottostati: Page, Page scan, Inquiry, Inquiry scan, Master response, Slave response, e Inquiry response (Fig. 2.2.6). Tali stati vengono attivati dal Link Controller al fine di gestire le risorse di una data piconet.

Lo stato Park, ad esempio, permette di avere la possibilità di costituire

piconet con più di 7 slave. Si possono così avere fino ad un massimo di 255 (2^8-1) dispositivi in modalità Park. Da tale stato si può tornare in connected mode sempre rispettando il limite massimo di 7 slave attivi per piconet. Lo stato Standby è lo stato di default per un dispositivo Bluetooth. In tale stato è possibile abilitare una modalità a basso consumo per risparmiare energia in caso di dispositivi alimentati mediante batterie. Dallo stato Standby è possibile passare ai sottostati Page, Page scan, Inquiry, Inquiry scan.

La particolare natura dinamica delle WPAN, che prevede la presenza di device attivi anche per brevissimi periodi di tempo, determina la necessità di poter interrogare gli altri dispositivi in range (Personal Operation Space, POS) per conoscerne lo stato, il BD address e altre informazioni.

A tale scopo, il sottostato detto di Inquiry permette di inviare un messaggio di interrogazione contenuto nell'ID packet (i tipi di pacchetto saranno descritti in dettaglio nel par. 2.3). Questo invio viene effettuato in successione su 32 canali Bluetooth, seguendo una inquiry *hop sequence* derivata dal LAP del General Inquiry Access Code (GIAC). Il ritmo di invio degli ID packet è pari a 3200 Hz (clock rate, durata pari a mezzo slot). I dispositivi in discovery mode (visibili) entrano regolarmente nel sottostato di Inquiry Scan per rispondere ad eventuali interrogazioni. La risposta viene inviata sulla stesso canale nella quale è stata ricevuta dopo un tempo pari a 625 μs ovvero nello slot successivo. Il dispositivo interrogante torna sulla frequenza di invio per ricevere eventuali risposte (*reply*). E' interessante notare che tali risposte non sono forzate e, per questo, esiste un *hidden mode* che impedisce la rilevazione del dispositivo. Si tratta di una forma di sicurezza per quei casi (luoghi pubblici in genere) in cui si scelga di non permettere l'accesso al dispositivo (anche se esistono ben note metodologie che aggirano tale protezione).

Al termine di una interrogazione, il dispositivo interrogante ottiene (tramite il pacchetto FHS) gli indirizzi MAC (BD addr) degli altri dispositivi in range ed i rispettivi clock offset. Tali dati possono essere impiegati nel passaggio ad un successivo sottostato detto di Page. In tale stato il Master (il nodo che inizia la trasmissione) può attivare ed avviare una connessione con uno Slave nello stato di Page Scan. Lo stato Page Scan consiste nell'ascoltare, un canale per volta, in una finestra di 11.25 ms (valore di default) correlando i segnali ricevuti con il proprio DAC (contenente il proprio BDaddr). In tale finestra d'ascolto è possibile ricevere da 16 canali di paging. Non appena il correlatore restituisce un output al di sopra della soglia, lo Slave può passare al sottostato di Slave Response. Lo stato di Page Scan può essere avviato dagli stati di Standby e Connection. In

Page mode il Master tenta di agganciare lo scanning operato dallo Slave trasmettendo un messaggio (su diversi canali) contenente il BDaddr dello Slave (contenuto nel pacchetto detto Dedicated Access Code, DAC). Non essendo i due device sincronizzati a priori, il Master deve attendere il reply dello Slave per ottenere informazioni utili all'instaurazione della connesione. Il Master conosce solo una stima del clock dello Slave e questa può essere stata ottenuta da una precedente connessione o inquiry. Con tali informazioni il Master può stimare l'hopping sequence che lo Slave sta impiegando per effettuare il Page Scan. Al fine di velocizzare lo scanning, l'hopping rate viene incrementato a 3200Hz, di fatto raddoppiando i salti in TX/RX per ciascuno slot (625us). Una volta che il Master riceve la risposta dallo Slave, può passare in nel sottostato Master response. Nel momento in cui lo Slave riceve con successo il page message tra Master e Slave vi è una sincronizzazione poco accurata. I due device si trovano nei sottostati di Slave response e Master response rispettivamente. In questa fase vi è uno scambio reciproco di messaggi al fine di sincronizzarsi ed impiegare il medesimo Channel Access Code (CAC) caratteristico di quella data piconet. Il pacchetto CAC e la hopping sequence sono ricavate dal clock e dal BDaddr del Master.

Step	Message	Packet type	Direction	Hopping sequence	Access code and clock
1	Page	ID	Master to slave	Page	Slave
2	First slave page response	ID	Slave to master	Page response	Slave
3	Master page response	FHS	Master to slave	Page	Slave
4	Second slave page response	ID	Slave to master	Page response	Slave
5	1st packet master	POLL	Master to slave	Channel	Master
6	1st packet slave	Any type	Slave to master	Channel	Master

Fig. 2.2.7: Scambio di messaggi in un paging

Nella Fig. 2.2.8, il Master è inizialmente nel sottostato di Page e lo Slave in Page Scan. Gli scambi successivi riassumono quanto detto. Un'altra utile immagine mostra questa prima fase di instaurazione della connessione.

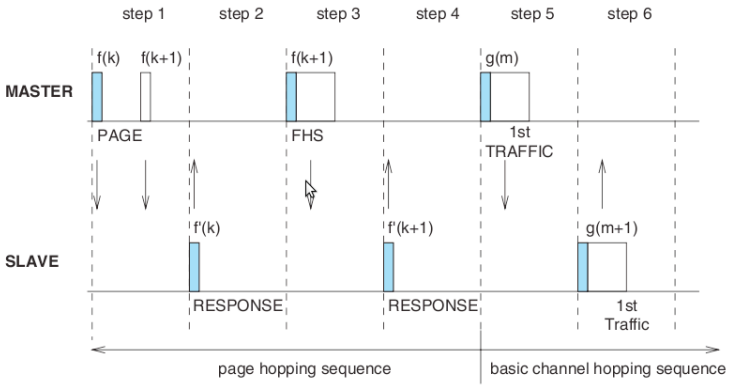


Fig. 2.2.8: Procedura di paging

Nella figura lo Slave risponde al primo messaggio di page da parte del Master su f(k) e quindi nel successivo slot risponde su quella stessa frequenza. Si vede, chiaramente, il passaggio da un rate di 3200 hop/s per la fase di Page e Page Scan al rate di 1600 hop/s negli step 5 e 6. Nello stato Connection, si ha una connessione instaurata tra due dispositivi in una piconet quindi i pacchetti contenenti user data possono transitare nelle due direzioni. La connessione è iniziata per mezzo del pacchetto POLL inviato dal Master per verificare la sincronizzazione dello Slave. Se il Master non riceve risposta entro un dato timeout, entrambi i device ritornano nei sottostati Page e Page Scan. Il primo scambio di pacchetti in connected mode sono scambiati a livello di Link Management al fine di negoziare parametri di livello datalink e quindi il tipo di canali logici di trasporto da instaurare (ACL, SCO, eSCO, ecc.).

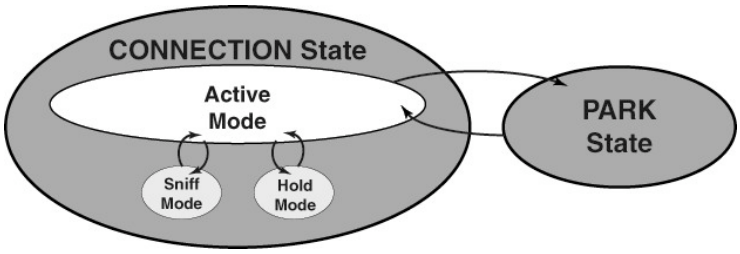


Fig. 2.2.9: Stato di connessione e suoi sottostati

In Active mode sia Master che Slave partecipano attivamente alla trasmissione. I device in tale modalità possono raggiungere un numero massimo di 7 per piconet. Gli Slave ascoltano sugli slot master-to-slave l'arrivo di pacchetti (active slaves).

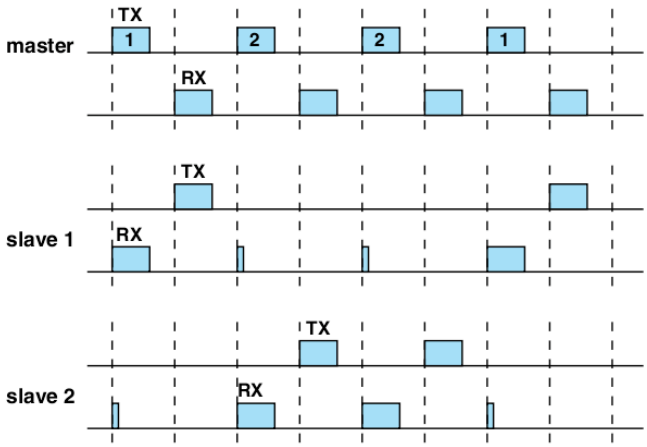


Fig. 2.2.10: Schema di polling per comunicazioni multislave

Un active slave riceve dal Master della piconet un indirizzo temporaneo detto Active Member address di lunghezza 3 bit. L'indirizzo "000" è invece impiegato per trasmissioni broadcast. Un dispositivo Slave può conoscere il numero di slot che il Master gli riserva attraverso il campo TYPE contenuto nell'Header dei pacchetti. Il periodo durante il quale si richiede attività ad uno slave di una piconet è indicato come Tpoll (deciso dal Link Manager) e misurato in slot. Il Master trasmette periodicamente agli slave pacchetti brevi (DM1 ad esempio) in modo da permettere la sincronizzazione degli slave (pacchetti CAC). Gli Slave che risultano forniti di indirizzo Logical Transport Address (LTaddr) associato quindi ad un link ACL o SCO, possono inviare pacchetti negli slot Slave-to-Master.

Il pattern che determina la successione di salti (hops) in frequenza è calcolato a partire dal MAC address (nel BT anche detto BDAddr, o Bluetooth Device Address) e dal clock del Master. Tutti i device afferenti ad una data piconet devono mantenersi sincronizzati con il relativo Master clock al fine di poter comunicare in frequency hopping con gli altri nodi.

Il Master di una piconet possiede sempre il completo controllo delle comunicazioni tra gli Slave. A causa dell'impiego del TDD gli Slave possono comunicare solo con il Master. Ciascuna comunicazione tra Slave deve quindi passare prima per il Master della piconet. Al fine di evitare collisioni sul canale di trasporto ACL, si adotta la seguente soluzione. Uno slave può trasmettere su un dato slot Slave-to-Master solo quando ha ricevuto il proprio LTaddr dall'Header del pacchetto ricevuto nel precedente slot Master-to-Slave. Se ciò non avviene, lo Slave non ha il per-

messo di trasmettere ad eccezione del caso in cui abbia un canale di trasporto sincrono dedicato (SCO).

Le specifiche (release 4 delle Core Specifications) proposte dal Bluetooth SIG [4] prevedono la nuova versione Bluetooth Low Power. Questa introduce una serie di nuove imporatanti funzionalità. Si prevede un incremento del numero di dispositivi ammissibili in active mode (fino a parecchie migliaia). La topologia di rete resta essenzialmente molto semplice (piconet, topologia a stella) ma vengono migliorate le funzionalità di routing tra più piconet (scatternet networking). Altra novità è data dall'integrazione di funzionalità di supporto all'impiego di web services. L'instaurazione delle connessioni è divenuta più veloce. I nuovi sistemi low power si pongono come nuovo Standard per i sensori dedicati ad applicazioni consumer come, ad esempio, orologi da polso, elettronica per il fitness e, più in generale, nella domotica.

Nel seguito si scenderà più in dettaglio nella descrizione degli "oggetti" di interesse per l'identificazione del segnale Bluetooth ovvero i pacchetti Bluetooth e le caratteristiche dei relativi segnali che li trasportano. In questo percorso verrà inoltre ripreso lo strato Bluetooth Radio nel quale vengono mo-demodulati i segnali Bluetooth.

2.3 Tipologie di pacchetto Bluetooth

I dispositivi Bluetooth si scambiano dati attraverso il canale radio per mezzo di pacchetti. Vengono definiti due possibili rate di trasmissione: il Basic Rate (BR) e l'Enhanced Data Rate (EDR). Nel Basic Rate il pacchetto prevede un primo campo detto Access Code di 72 bit, seguito da un campo contenente l'Header di 54 bit ed infine un campo variabile da 0 a 2745 bit in cui incapsulare il payload.

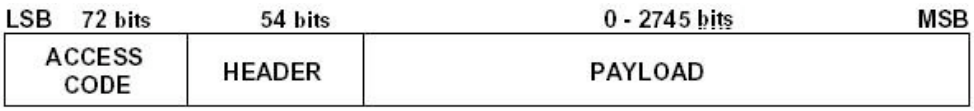


Fig. 2.3.1: Struttura di un pacchetto Bluetooth (Basic Rate, BR)

I pacchetti inviati presentano una inversione di bit in modo da risultare big-endian ovvero il bit più significativo è l'ultimo bit del payload (in

aria viene inviato prima il LSB). Questa convenzione è impiegata nei protocolli di comunicazione (ma anche dai processori SUN) e spesso prende il nome di network order. Tale scelta ha origini storiche, risalenti alle prime applicazioni nelle reti telefoniche ed è stata poi adottata anche per l'Internet Protocol (IP) divenendo così molto comune nelle reti. L'altra possibilità di rappresentazione è data dall'ordine detto little-endian, che corrisponde allo schema adottato da molte comuni architetture per calcolatori (processori Intel) e comunemente detta host order. Per fare un esempio, nel caso di una Word (ovvero 2 byte, 16 bit), il numero esadecimale 0x0123 (il suffisso 0x è usato per indicare un numero in base 16) verrà immagazzinato come:

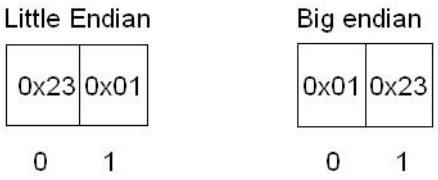


Fig. 2.3.2: Rappresentazione little-endian vs big-endian

E' molto importante aver chiari questi semplici schemi perché regolano il passaggio dall'interfaccia di rete agli strati superiori. Nello Standard IEEE, nel definire il modulo Baseband del Bluetooth, si include la definizione del BDaddr (Bluetooth Device Address) ovvero il campo di 48 bit che descrive, in modo univoco, l'indirizzo Bluetooth di un device. Il BDaddr corrisponde di fatto all'indirizzo MAC dell'interfaccia radio Bluetooth (assegnato dalla IEEE Registration Authority) ed è scomposto nelle seguenti parti: LAP (24 bit, Lower Address Part), UAP (8 bit, Upper Address Part) e NAP (16 bit, Non-significant Address Part).

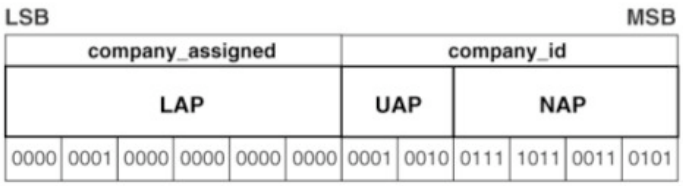


Fig. 2.3.3: Bluetooth Device Address

Come si può vedere, il LAP è la parte corrispondente all'indirizzo di una data azienda produttrice (Nokia, Samsung, Fujitsu, ecc.). Il BDaddr può assumere qualsiasi valore ad eccezione di un insieme di 64 LAP riservati (da 0x9E8B00 a 0x9E8B3F) che vengono impiegati per inquiry di tipo generale (GIAC) o dedicato (DIAC).

Non appena un dispositivo entra a far parte di una piconet gli viene assegnato un indirizzo detto AMaddr (Active Member Address) di lunghezza 3 bit. In questo schema si ritrova il numero massimo di nodi per una piconet pari a 23 = 8 dispositivi. Se la piconet è già al suo numero massimo di nodi, i successivi device che richiedono l'accesso vengono posti in Park mode. In tale stato, questi nodi ottengono un indirizzo detto PMaddr (Passive Member Address) di lunghezza pari a 8 bit. In tal caso si possono accettare 28=256 dispositivi. Non appena un indirizzo di tipo AMaddr torna libero è possibile assegnarlo a un device posto in Park mode.

Tutti i pacchetti Bluetooth iniziano con un campo detto Access Code. Se dopo di esso è previsto un Header allora la lunghezza del campo Access Code è di 72 bit, altrimenti è pari a 68 bit (shortened access code). La differenza tra i due formati è dovuta alla presenza o meno di un campo detto trailer (4 bit). L'Access Code permette la sincronizzazione tra due nodi, la compensazione del DC offset e l'identificazione dei pacchetti. Infatti tutti i pacchetti inviati sullo stesso canale fisico sono preceduti dal medesimo Access Code. Al ricevitore un correlatore a scorrimento rileva la ricezione di un pacchetto e innescando le successive elaborazioni in banda base (modulo Baseband). Lo shortened Access Code è presente nelle fasi di inquiry, paging e nello stato Park. In tali situazioni l'Access Code funge da messaggio di segnalazione. L'Access Code è composto dai campi Preamble (4 bit), *sync word* (64 bit) ed il trailer (4 bit) presente, come già detto, solo in caso di successivo Header.

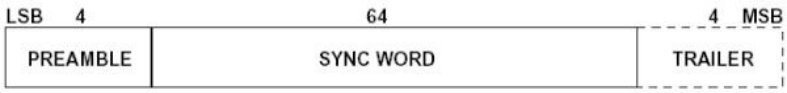


Fig. 2.3.4: Access Code (AC) dei pacchetti Bluetooth

Sono previsti 4 diversi tipi di Access Code: CAC (Channel Access Code), DAC (Dedicated Access Code), GIAC (General Inquiry Access Code), DIAC (Dedicated Inquiry Access Code). Il CAC è posto all'inizio di tutti i pacchetti scambiati nello stato di connessione e contiene la LAP del Ma-

ster della piconet. Il DAC invece è posto all'inizio dei pacchetti scambiati nei sottostati di page, page scan e i corrispondenti stati di response e viene calcolato a partire dal LAP del dispositivo al quale si rivolge (paged device). Nello stato di inquiry viene invece impiegato un GIAC (impiegando un LAP dedicato, 0x9E8B33) per rilevare tutti i dispositivi raggiungibili oppure un DIAC (63 possibili indirizzi, 0x9E8B00 to 0x9E8B3F) per rilevare quelli attivi.

Code type	LAP	Code length	Comments
CAC	Master	72	See also 8.1.3
DAC	Paged device	68/72[a]	
GIAC	Reserved	68/72[a]	
DIAC	Dedicated	68/72[a]	

[a]Length 72 is used only in combination with FHS packets.

Fig. 2.3.5: Tipologie di Access Code (AC)

Il Preamble è formato da una sequenza fissa di zero e uno lunga 4 bit impiegata per compensare la componente continua (DC compensation). La sequenza di preamble assume la forma "1010" se il LSB della *sync word* seguente è pari a "1", altrimenti è "0101".

Fig. 2.3.6: Preambolo dell'Access code

La *sync word* invece contiene una code word di 64 bit calcolata a partire dal LAP (Lower Address Part del BDaddr, 24 bit). Nel caso di pacchetti CAC, per la *sync word* viene usata la LAP del BDaddr del Master. Nel caso di GIAC e DIAC vengono invece impiegate LAP riservate (dette anche dedicate). Nel caso di pacchetti DAC viene invece impiegato il LAP del BDaddr dello Slave.

L'algoritmo che definisce la *sync word* garantisce che, tra quelle basate su differenti LAP, vi sia una elevata distanza di Hamming (bassa cross-correlazione). Inoltre ciascuna *sync word* deve risultare PN al fine di garanti-

re ottime proprietà di sincronizzazione basate sul calcolo dell'autocorrelazione. Più in dettaglio, le *sync word* sono basate su un codice (64, 30) che viene sommato bit a bit (XOR) con un codice PN di pari lunghezza (64 bit). La distanza di Hamming tra *sync word* basate su LAP diverse è pari a $d_{min} = 14$.

Al fine di permettere una corretta sincronizzazione, la funzione di autocorrelazione delle *sync word* deve presentare un picco concentrato nell'origine. Tale comportamento viene realizzato per mezzo di una sequenza di Barker di lunghezza 7 bit, costruita aggiungendo 6 bit (“110010” o “001101”) a seconda, rispettivamente, che il MSB del LAP sia pari a “1” o “0”.

Un Trailer di 4 bit è, infine, aggiunto dopo la *sync word* se l'Access Code precede un Header. Tali 4 bit risultano essere pari a “1010” o “0101” se, rispettivamente il MSB della *sync word* risulta pari a “0” o “1”. Il Trailer e gli ultimi 3 MSBs della *sync word* formano una sequenza di 7 bit di zero e uno alternati che viene impiegata per compensare più efficacemente la componente continua. Il Trailer è presente nel Channel Access Code ma viene anche impiegato nel DAC e nell'IAC nel caso di page response ed inquiry response (pacchetti FHS).

Fig. 2.3.7: Trailer dell'Access code

L'Header dei pacchetti Bluetooth è lungo 18 bit e trasporta informazioni di strato 2 (Link Control). In esso sono presenti 6 campi: LT_ADDR (3 bit) con il logical transport address, TYPE (4 bit) codice per indicare il tipo di pacchetto, FLOW (1 bit) controllo di flusso, ARQN (1 bit) indicatore di acknoledge, SEQN (1 bit) sequence number, HEC (8 bit) per il controllo di errore sull'Header.

Fig. 2.3.8: Campi all'interno dell'Header

Nel campo LT_ADDR è indicato l'indirizzo logico di trasporto ovvero indica lo Slave destinatario nel caso di trasmissioni in slot Master-to-Slave e lo Slave di origine nel caso di trasmissioni in slot Slave-to-Master.

Il campo TYPE permette di avere informazioni sul tipo di pacchetto ovvero se si tratta di pacchetto SCO, eSCO o ACL. Esso da, inoltre, informazioni sull'attivazione o meno della modalità EDR e informazioni sulla durata (in slot) del pacchetto. La conoscenza della durata del pacchetto può essere d'ausilio per evitare che un ricevitore (non destinatario) resti su un dato canale per più di 1 slot se non necessario.

Il campo FLOW è impiegato per il controllo di flusso nel caso di link di trasporto ACL. In caso di buffer di ricezione pieno, il campo FLOW viene posto a "0" (STOP), così che venga inibito l'invio di nuovi pacchetti ACL al trasmettitore. Il campo ARQN permette di indicare con un ACK o un NACK, rispettivamente, una corretta ricezione del payload (CRC corretto) o una ricezione errata. Il campo SEQN è usato per mantenere un ordinamento nello stream di dati inviati. Il campo HEC consiste in un Header Error Check in grado di restituire un controllo sui bit errati. Gli 8 bit dell'HEC proteggono i precedenti 10 bit dell'Header.

Segment	TYPE code $b_3b_2b_1b_0$	Slot occupancy	SCO logical transport (1 Mbps)	eSCO logical transport (1 Mbps)	eSCO logical transport (2-3 Mbps)	ACL logical transport (1 Mbps) ptt=0	ACL logical transport (2-3 Mbps) ptt=1
1	0000	1	NULL	NULL	NULL	NULL	NULL
	0001	1	POLL	POLL	POLL	POLL	POLL
	0010	1	FHS	reserved	reserved	FHS	FHS
	0011	1	DM1	reserved	reserved	DM1	DM1
2	0100	1	undefined	undefined	undefined	DH1	2-DH1
	0101	1	HV1	undefined	undefined	undefined	undefined
	0110	1	HV2	undefined	2-EV3	undefined	undefined
	0111	1	HV3	EV3	3-EV3	undefined	undefined
	1000	1	DV	undefined	undefined	undefined	3-DH1
	1001	1	undefined	undefined	undefined	AUX1	AUX1
3	1010	3	undefined	undefined	undefined	DM3	2-DH3
	1011	3	undefined	undefined	undefined	DH3	3-DH3
	1100	3	undefined	EV4	2-EV5	undefined	undefined
	1101	3	undefined	EV5	3-EV5	undefined	undefined
4	1110	5	undefined	undefined	undefined	DM5	2-DH5
	1111	5	undefined	undefined	undefined	DH5	3-DH5

Fig. 2.3.9: Tipologie di pacchetto (da 1, 3, 5 slot)

I tipi di pacchetto elencati nel segment 1 in tabella, ovvero quelli che presentano una lunghezza pari ad 1 slot, sono molto comuni in una trasmissione Bluetooth. In totale i tipi di pacchetti più comuni sono: NULL, POLL, FSH, DM1 (usato sia per i dati che per la segnalazione) e l'ID.

L'ID packet non compare nello schema Bluetooth SIG precedente, per il fatto che non presenta il campo Header. Tale pacchetto ID (anche detto identity packet) può contenere l'IAC oppure il DAC e presenta una lunghezza fissa di 68 bit. La sua ricetrasmissione è alquanto robusta per via dell'impiego, in ricezione, di un correlatore bit a bit.

Il pacchetto NULL non presenta payload ma contiene il CAC e l'Header per una lunghezza fissata a 126 bit. Esso può essere impiegato per rispondere a seguito di una ricezione avvenuta con successo ARQN o per inviare informazioni sullo stato del buffer in ricezione. Il pacchetto di NULL non necessita di ACK in risposta.

Il pacchetto POLL, allo stesso modo del NULL, non presenta payload ma necessita di ACK. Viene impiegato solo dal Master per fare polling sugli Slave della piconet. Per tale pacchetto non viene previsto un incremento del SEQN. Ad ogni POLL del Master gli Slave devono rispondere anche se non hanno dati da trasmettere.

Il pacchetto FHS è uno speciale pacchetto di controllo che contiene una serie di informazioni tra cui il BDaddr e il Clock del device sorgente.

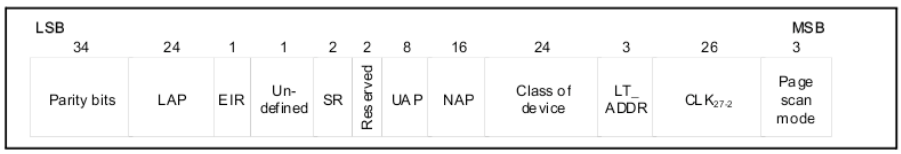


Fig. 2.3.10: Il pacchetto FHS

Il payload contiene 144 bit protetti da 16 bit di CRC. Il payload è, inoltre, codificato con un FEC 2/3 che produce una stringa di 240 bit. Tale pacchetto consta di 11 campi (riassunti nella figura seguente dal Bluetooth SIG). Esso viene impiegato in fase di Page Master response, Inquiry response e role switch (in caso di cambiamento del ruolo Master-Slave in una data trasmissione).

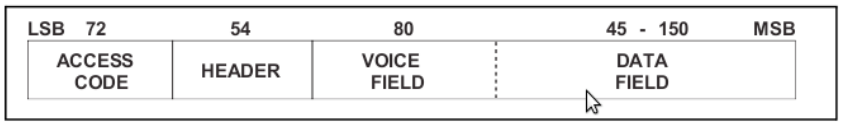
Parity bits	This 34-bit field contains the parity bits that form the first part of the sync word of the access code of the device that sends the FHS packet. These bits are derived from the LAP as described in Section 1.2 on page 68.
LAP	This 24-bit field shall contain the lower address part of the device that sends the FHS packet.
EIR	This bit shall indicate that an extended inquiry response packet may follow. See Section 8.4.3 on page 163.
Undefined	This 1-bit field is reserved for future use and shall be set to zero.
SR	This 2-bit field is the scan repetition field and indicates the interval between two consecutive page scan windows, see also Table 6.4 and Table 8.1 on page 152
Reserved	This 2-bit field shall be set to 10.
UAP	This 8-bit field shall contain the upper address part of the device that sends the FHS packet.
NAP	This 16-bit field shall contain the non-significant address part of the device that sends the FHS packet (see also Section 1.2 on page 68 for LAP, UAP, and NAP).
Class of device	This 24-bit field shall contain the class of device of the device that sends the FHS packet. The field is defined in Bluetooth Assigned Numbers.
LT_ADDR	This 3-bit field shall contain the logical transport address the recipient shall use if the FHS packet is used at connection setup or role switch. A slave responding to a master or a device responding to an inquiry request message shall include an all-zero LT_ADDR field if it sends the FHS packet.
CLK_{27-2}	This 26-bit field shall contain the value of the native clock of the device that sends the FHS packet, sampled at the beginning of the transmission of the access code of this FHS packet. This clock value has a resolution of 1.25ms (two-slot interval). For each new transmission, this field is updated so that it accurately reflects the real-time clock value.
Page scan mode	This 3-bit field shall indicate which scan mode is used by default by the sender of the FHS packet. The interpretation of the page scan mode is illustrated in Table 6.5.

Fig. 2.3.11: Descrizione dei campi del pacchetto FHS

Nel FHS è possibile scorgere LAP, UAP, e NAP ovvero il BDaddr del sender, inoltre, attraverso la conoscenza dei parity bits e del LAP è possibile ottenere il CAC del device (Master) che ha inviato il pacchetto FHS. I pacchetti dati possono essere trasportati dai canali logici di trasporto chiamati SCO, eSCO oppure ACL. I pacchetti Synchronous Connection Oriented o SCO sono suddivisi nei seguenti formati:

```
HV1: 10 byte di dati, 1/3 FEC, senza CRC, 240 bit di payload
HV2: 20 byte di dati, 2/3 FEC, senza CRC, 240 bit di payload
HV3: 30 byte, nessun FEC né CRC, 240 bit di payload
```

Il formato DV trasporta voce (80 bit) e dati (150 bit). Il traffico voce non presenta FEC. Il campo dati può contenere da 1 a 10 bytes, compreso 1 byte di Header per il payload e 16 bit di CRC. Il campo dati (assieme al CRC) è, inoltre, protetto da un FEC 2/3.



Il link di trasporto di tipo eSCO (enhanced SCO) fornisce una modalità di trasmissione audio ad alta qualità ed, in caso di perdita dei dati, questi vengono ritrasmessi per migliorare la qualità audio. I formati previsti in BR per i link eSCO sono:

EV3: da 1 a 30 byte (negoziato in fase di eSCO setup dal LMP), 16 bit di CRC, nessun FEC, 1 slot

EV4: da 1 a 120 byte (negoziato in fase di eSCO setup dal LMP), 16 bit di CRC, FEC 2/3, occupa fino a 3 slot

EV5: da 1 a 180 byte (negoziato in fase di eSCO setup dal LMP), 16 bit di CRC, nessun FEC, occupa fino a 3 slot

Nel caso di Enhanced Data Rate (EDR) vengono definiti 4 altri formati di pacchetti eSCO: 2-EV3, 3-EV3, 2-EV5, 3-EV5. In questi pacchetti vengono impiagati formati di modulazione diversi dal GFSK ovvero il formato di modulazione digitale π/4-DQPSK (2-EV3, 2-EV5) che permette di raggiungere 2Mb/s e l'8DQPSK (3-EV3, 3-EV5) che può offrire un bitrate lordo di 3Mb/s. I link di tipo ACL in BR sono invece basati sui seguenti tipi di formati: DM1, DH1, DM3, DH3, DM5, DH5, AUX1.

DM1: da 1 a 18 byte, occupa 1 slot, CRC di 16 bit, FEC 2/3, payload header di 1 byte.

DM3: da 2 a 123 byte, occupa 3 slot, CRC di 16 bit, FEC 2/3, payload header di 2 byte.

DM5: da 2 a 226 byte, occupa 5 slot, CRC di 16 bit, FEC 2/3, payload header di 2 byte.

DH1, DH3, DH5: lo stesso dei pacchetti DM1, DM3, DM5 rispettivamente, ma senza l'impiego di una codifica FEC.

Il pacchetto AUX1 è di tipo DH1 ma senza CRC. Esso presenta da 1 a 30 byte di payload ed occupa 1 solo slot.

Nel caso di modalità EDR i pacchetti di livello di trasporto ACL sono: 2-DH1, 2-DH3, 2-DH5 con schema di modulazione π/4-DQPSK, e 3-DH1, 3-DH3, 3-DH5 con schema di modulazione 8DQPSK.

Il *bitstream processing* è dato da quell'insieme di operazioni che vengono svolte prima di inviare i pacchetti sull'interfaccia radio. Tali operazioni sono a livello di bitstream e hanno lo scopo di aumentare l'affidabilità e la confidenzialità della trasmissione.

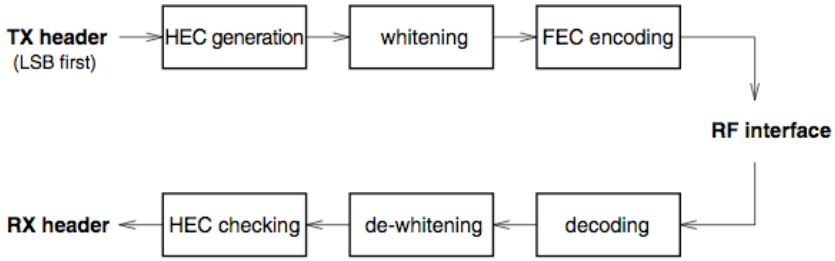


Fig. 2.3.12: Bitstream processing

Nel caso dei bit che compongono l'Header, il bitstream processing prevede l'applicazione di un HEC (Header Error Check) di 8 bit, uno XOR dell'intero Header con una sequenza PN (whitening) e una codifica FEC. Al ricevitore l'intero processo è svolto inversamente. Dallo Standard si ha che il bitstream processing sull'Header ha carattere obbligatorio.

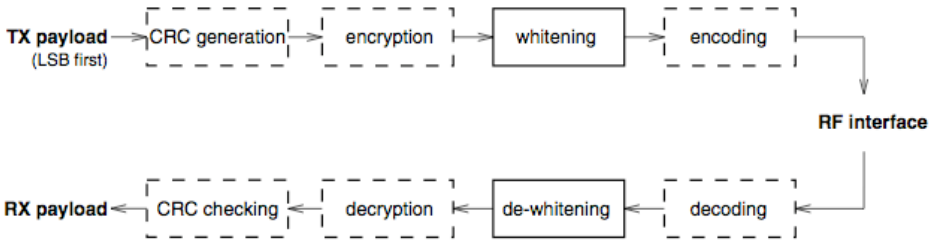


Fig. 2.3.13: Whitening e de-whitening nel bitstream processing

Nel caso del payload la situazione è leggermente diversa. Si prevede infatti la possibilità di criptare i dati (encryption) ed, inoltre, tutto il processamento non ha più carattere obbligatorio ad eccezione del whitedewhitening. In test mode invece è possibile disabilitare anche questa funzione, di fatto escludendo ogni operazione di bitstream processing.

Il controllo degli errori (*Error Checking*) è una funzionalità ottenuta attraverso i pacchetti CAC, l'impiego di somme di controllo CRC (Cyclic Redundancy Check) o attraverso l'HEC nell'Header. Alla ricezione di un

pacchetto viene dapprima controllata l'integrità del Access Code ed in particolare dei 64 bit della *sync word* (derivabili attraverso la LAP del Master). Questo check previene, nei dispositivi commerciali, la ricezione di pacchetti provenienti da piconet diverse inibendo di fatto qualunque possibilità di cattura di tali pacchetti (sniffing). I controlli basati sul calcolo dell'HEC e del CRC invece fanno riferimento all'UAP del Master.

Il *data whitening* è l'operazione che permette di rendere pseudo-noise il flusso di dati. Questa operazione viene applicata sia all'Header che al Payload dei pacchetti con lo scopo di minimizzare la componente continua (DC bias) presente nel bitstream. Il data whitening avviene prima della codifica FEC. Al ricevitore l'operazione inversa viene ottenuta attraverso lo XOR con la medesima sequenza di scrambling (ottenuta per mezzo di un registro a scorrimento, LFSR) usata in trasmissione. La correzione degli errori (*Error Correction*) è svolta mediante uno tra questi tre schemi: 1/3 rate FEC, 2/3 rate FEC, oppure uno schema di Automatic Repeat reQuest o ARQ. Lo schema FEC 1/3 consiste in un semplice codice a ripetizione di lunghezza 3 bit.

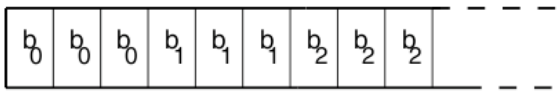


Fig. 2.3.14: Schema FEC 1/3

Lo schema FEC 2/3 consiste in un codice di Hamming (15, 10), con il quale ogni blocco di 10 bit è codificato da una codeword di 15 bit. Il codice è ottenuto attraverso un LFSR. Tale codice è in grado di correggere tutti gli errori sul singolo bit e di rilevare errori di 2 bit in ogni codeword.

Lo schema di Automatic Repeat reQuest adottato dal Bluetooth prevede la presenza di ACK nell'Header dei pacchetti di ritorno ed eventuale ritrasmissione dei pacchetti non ricevuti correttamente. Tale schema viene impiegato solo con i pacchetti che presentano il campo CRC.

I pacchetti ora descritti nell'attraversare il canale radio wireless passano dapprima attraverso lo strato detto Bluetooth Radio. In tale strato si svolgono le operazioni di mo-demodulazione necessarie per ottenere il segnale GFSK ed il FHSS nella banda ISM 2.4GHz caratteristici di questa tecnologia.

2.4 Le specifiche di strato fisico

Come già accennato, al di sotto dell'interfaccia HCI, che separa l'host dal controlle Bluetooth, vi sono tre moduli che mappano gli strati ISO/OSI di livello 1 e 2 e sono: il Link Management Protocol (LMP), il Baseband e il Bluetooth Radio. Quest'ultimo non rappresenta un vero e proprio strato quanto piuttosto l'insieme di caratteristiche fisiche del segnale emesso dal device Bluetooth.

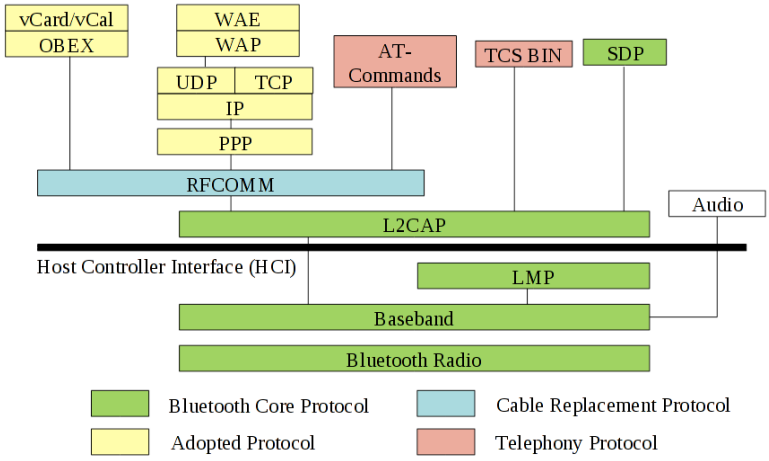


Fig. 2.4.1: Lo stack Bluetooth

In LMP risiedono tutte le funzionalità di instaurazione, configurazione e mantenimento del collegamento, nonché di autenticazione. Tale modulo provvede a gestire il discovery di altri LM remoti e a comunicare con essi attraverso il Link Management Protocol (LMP). Il LMP opera per mezzo del Link Controller (LC). Il protocollo LMP gestisce lo scambio di PDU (protocol data units) che vengono spedite dall'LM di un dispositivo all'altro per via di un indirizzo AMaddr nell'Header. Tali PDU occupano sempre un solo slot (pacchetti DM1 per l'ACL e HV1 per l'SCO).

Le PDU generabili dallo strato LMP sono: General Response, Authentication, Pairing, Change Link Key, Change the Current Link Key, Encryption, Slot Offset Request, Clock Offset Request, Timing Accuracy Information Request, LMP Version, Supported Features, Switch of Master-Slave Role, Name Request, Detach, Hold Mode, Sniff Mode, Park Mode, Power Control, Channel Quality-Driven Change, Quality of Service, SCO Links, Control of Multi-Slot Packets, Paging Scheme, Link Supervision,

Connection Establishment, Test Mode, Error Handling.

Nel modulo Baseband si realizzano gran parte delle funzioni di strato fisico. In esso vengono gestiti i canali fisici ed, inoltre, vengono offerti servizi come la correzione di errore (link error correction), il data whitedewhitening, la selezione del next hop, e la sicurezza. Il layer Baseband è implementato per mezzo del Link Controller che lavora in cooperazione con il LM per realizzare le funzionalità di gestione del collegamento (link management) e della potenza (power control). In Baseband vengono gestiti i collegamenti sincroni (SCO, eSCO) e asincroni (ACL), vengono gestiti i pacchetti e effettuate funzioni di paging e inquiry. In questo modulo viene anche gestito il TDD ovvero l'utilizzo degli slot in ricetrasmissione e il Frequency Hopping.

I requisiti di strato fisico (physical layer) del sistema Bluetooth, nello Standard IEEE 802.15.1 vengono indicati sotto il nome di Bluetooth Radio.

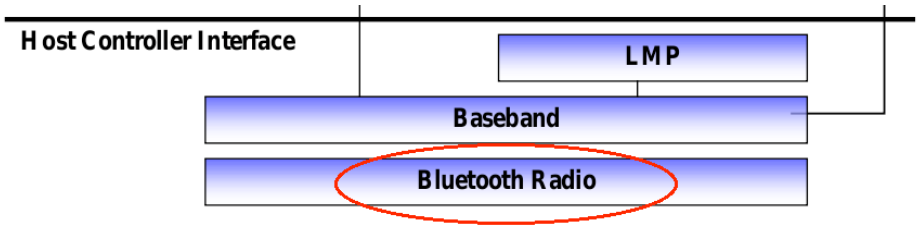


Fig. 2.4.2: Livello Bluetooth Radio e strati superiori nello stack Bluetooth

Si analizzano ora in dettaglio le caratteristiche dello strato fisico (Baseband e Bluetooth Radio) al fine di introdurre la successiva analisi del segnale.

La radio Bluetooth, come già accennato, utilizza la tecnica di modulazione a spettro espanso (Spread Spectrum, SS) denominata Frequency Hopping Spread Spectrum (FHSS). La tecnica FHSS consiste nell'impiegare ricetrasmettitori a banda stretta (1 *MHz*, nel caso del Bluetooth) in grado di saltare (hop) su diverse frequenze portanti (canali) definite nella banda ISM (la banda ISM a 2.4 *GHz* presenta una larghezza di 80 *MHz*). Nella banda ISM per il BT vengono così definite 79 possibili frequenze di hop (23 in alcuni paesi). Il salto tra le 79 possibili frequenze è regolato da un codice Pseudo-Noise (PN) detto anche frequency hopping (FH) code.

Nei diversi paesi del mondo esistono, come anticipato, differenze in merito alla scelta e al numero dei canali a RF per il Bluetooth. La figura se-

guente mostra le differenze tra i diversi paesi nel mondo. Va notato, inoltre, che esistono delle bande di guardia al'inizio ed alla fine della banda ISM, così che in 80 *MHz* sono contenuti 79 canali da 1 *MHz*.

Country	Frequency Range	RF Channels	
Europe* & USA	2400 – 2483.5MHz	f = 2402 + k MHz	k = 0, ..,78
Japan	2471 – 2497 MHz	f = 2473 + k MHz	k = 0, ..,22
Spain	2445 – 2475 MHz	f = 2449 + k MHz	k = 0, ..,22
France	2446.5 – 2483 MHz	f = 2454 + k MHz	k = 0, ..,22

* Except Spain and France

Fig. 2.4.3: Allocazione canali fisici nella banda ISM nei diversi paesi

La velocità di salto (hopping rate) è di 1600 hop/sec (valore nominale) ma può raggiungere il valore di 3200 hop/s (clock BT, 3200 Hz) nella fase di discovery di altri dispositivi. Il valore di 1600 hop/s è raggiunto nel caso di trasmissioni con pacchetti lunghi non più di 1 slot (ad es. pacchetti DM1). La tecnica FHSS permette di impiegare ricetrasmettitori molto semplici (banda stretta, 1 *MHz*) ma, allo stesso tempo, di combattere il fading sfruttando i vantaggi di una trasmissione a spettro espanso (80 *MHz*).

I trasmettitori Bluetooth possono essere racchiusi in 3 classi in base alla massima potenza trasmessa. I dispositivi di classe 1 sono designati per applicazioni a lungo raggio (~100 m) e presentano una massima potenza trasmessa di 20 dBm. I dispositivi di classe 2 vengono designati per impieghi comuni (~10 m) e rappresentano la maggior parte dei dispositivi in commercio. La potenza massima in trasmissione di un dispositivo di classe 2 è di 4 dBm. I dispositivi di classe 3 hanno una potenza massima in trasmissione di 0 dBm e quindi una portata molto ridotta (~10 cm) utile per il cable replacement ed applicazioni simili.

Ciascuno di questi tipi di dispositivo possiede una potenza in trasmissione nominale di 0 dBmW che poi può essere variata in base ad algoritmi di controllo di potenza gestiti dallo strato LMP (Link Manager Protocol). Tali algoritmi si basano su misure di potenza ricevuta (RSSI, Radio Signal Strength Indicator) e su comandi LMP in grado di innescare un aumento o una diminuizione della potenza trasmessa a seconda dello scenario.

Le specifiche dello strato Radio del Bluetooth prevedono 2 tipi di modulazioni possibili. Una prima tipologia di modulatore è ritenuta fondamentale ed è alla base della modalità denominata Basic Rate (BR). In modalità Basic Rate viene impiegato un modulatore GFSK in grado di garantire semplicità di implementazione. Una seconda modalità, considerata opzionale, è detta Enhanced Data Rate (EDR). La modalità EDR è caratterizzata da una modulazione di tipo PSK (Phase Shift Keying) e presenta 2 varianti: la π/4-DQPSK e la modulazione 8DPSK. Per tutte le tipologie di modulazione previste, il symbol rate è di 1MS/s. Il bit rate presente in aria è di 1 Mbps per il BR, 2 Mb/s per l'EDR che impiega il π/4-DQPSK e di 3 Mb/s per l'EDR che impiega la modulazione 8DPSK.

La modulazione π/4-DQPSK è una modulazione differenziale ovvero ciascun simbolo è codificato a partire dallo differenza di fase con il simbolo precedente. Nella modulazione QPSK sono previsti 22 diversi shift di fase: 0, +π/2, +π, -π/2 che permettono di codificare 2 bit/simbolo (1 Mbaud/s → 2Mb/s). Nel caso di modulazione 8DPSK si impiegano 3 bit per codificare 23 possibili shift di fase tra il simbolo trasmesso e quello precedente (1Mbaud/s → 3 Mb/s).

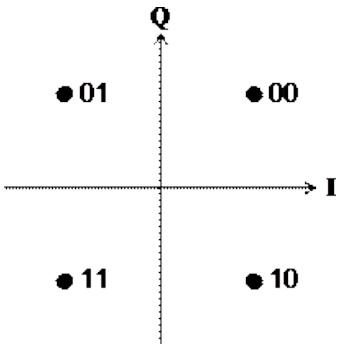


Fig. 2.4.5: Modulazione DQPSK

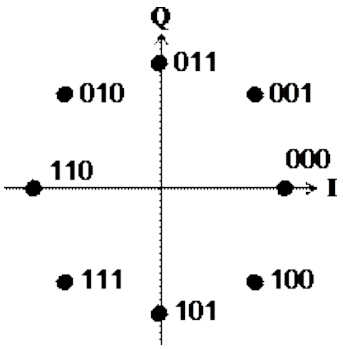


Fig. 2.4.4: Modulazione 8DPSK

Le trasmissioni full-duplex in Bluetooth sono ottenute per mezzo di uno schema a divisione di tempo (Time Division Duplex, TDD) tra Master e Slave attraverso l'impiego di slot da 625us. Il Master trasmette ogni 1250us.

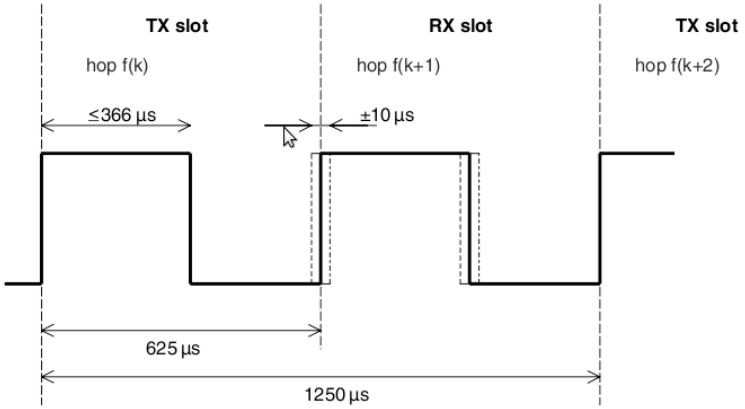


Fig. 2.4.6: Ricetrasmissione di pacchetti di durata pari a 1 slot nel Master

In uno slot da 625 *µs*, la tramissione dati può occupare al massimo 366 *µs*. Tale periodo corrisponde alla durata della trasmissione di un pacchetto DM1 o DH1 (ad es. DH1: 72 access + 54 head + 224 payload + 16 CRC = 366 *bit* → GFSK binario 1 *bit/µs* → 366 *µs*). Attorno all'istante esatto in cui è prevista la ricezione di un pacchetto si ammette una finestra di incertezza di larghezza pari a 20us (+/- 10us attorno al valore esatto). Il Master trasmette negli slot pari (0, 2, 4, …) mentre lo Slave trasmette negli slot dispari. La sincronizzazione dello slave è mantenuta per mezzo di CAC nei pacchetti diretti dal Master allo Slave.

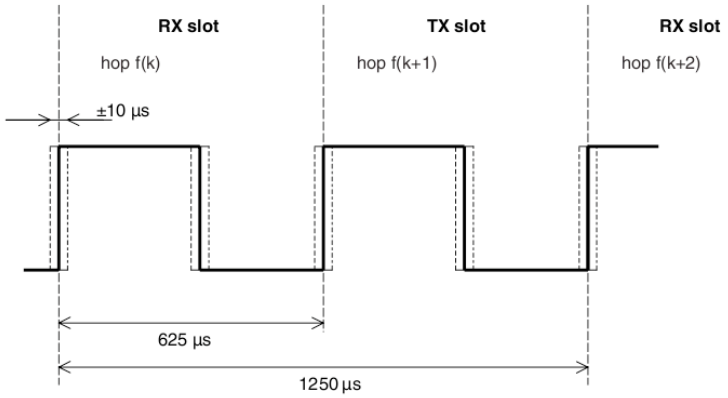


Fig. 2.4.7: Ricetrasmissione di pacchetti di durata pari a 1 slot nello Slave

Il sistema di accesso al mezzo adottato è di tipo TDMA (Time Division Multiple Access). Il sistema risultante TDD/TDMA sarà qui di seguito descritto, dopo aver definito meglio il clock del sistema Bluetooth. Il

clock nel Bluetooth è implementato per mezzo di un contatore di 28 bit che torna a zero dopo 2^{28}-1 battiti di clock.

Un ciclo di clock (*clock tick*) avviene ogni 312.5 *µs*, ovvero con una frequenza di 3200 *Hz*. Il contatore del clock impiega quindi poco più di 23 ore ovvero circa un giorno ad azzerarsi. Ciascun dispositivo Bluetooth possiede il suo clock (CLKN) derivato da un oscillatore al quarzo interno. La sincronizzazione del clock di un device con quello del Master (CLK) avviene sommando l'offset tra i due clock al valore del CLKN.

Fig. 2.4.8: Clock e sincronizzazione dello slave al clock della piconet

Vi sono 4 periodi fondamentali derivati dal battito di clock: 312.5 *µs* (CLK0), 625 *µs* (CLK1), 1.25 *ms* (CLK2) e 1.28 *s* (CLK12).

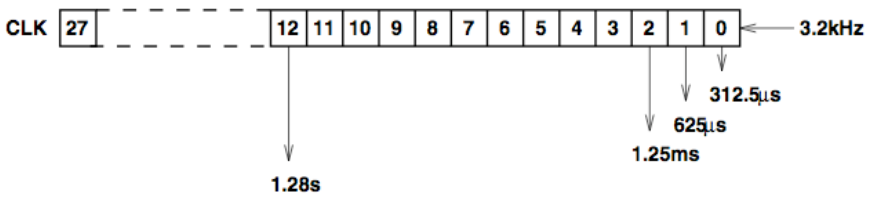


Fig. 2.4.9: Periodi caratteristici del sistema Bluetooth

I diversi clock possono quindi essere: CLKN (clock nativo), CLK (clock Master), CLKE (clock stimato). L'accuratezza richiesta al CLKN (di riferimento anche per gli altri due) è di ±20 *ppm* (in Hold e Sniff mode è sufficiente ±250 *ppm*). Uno slot è ricavato da 2 battiti di clock e quindi presenta una durata di 625 *µs*. La comunicazione avviene sempre tra 2 soli dispositivi, uno detto Master e l'altro Slave. Il ruolo del Master è quello di determinare il clock della trasmissione e di offrire allo Slave il riferimento per la sincronizzazione. Il Frequency Hopping code (FH code) come già accennato è dato da una sequenza Pseudo-Noise (PN) che può essere lunga 79 valori (numero dei canali Bluetooth in banda ISM) in connected mode oppure di 32 valori per gli stati di inquiry e paging.

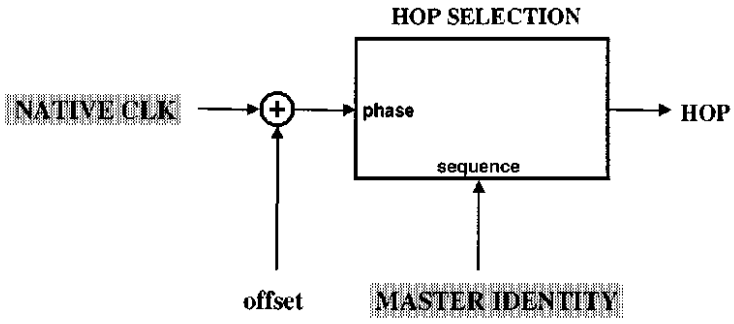


Fig. 2.4.10: Selezione della hopping sequence

Conoscendo il BDaddr del Master e l'offset tra il proprio clock (native clock) e il clock del Master, lo Slave è in grado di calcolare la hopping sequence. La sequenza vera e propria è calcolata a partire dal LAP dell'indirizzo del Master, mentre il punto di inizio della sequenza è dato dallo sfasamento tra i clock dei 2 dispositivi. Una volta che Master e Slave condividono il Frequency Hopping pattern e sono sincronizzati, si ha che il Master inizia a trasmettere occupando sempre gli slot dispari (1, 3, 5, ...) mentre lo Slave risponde sempre sugli slot pari (2, 4, 6, ...).

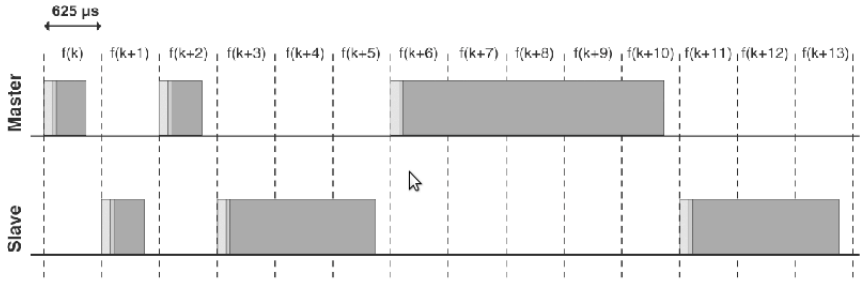


Fig. 2.4.11: Trasmissione di pacchetti multislot

Nel caso di pacchetti di lunghezza maggiore di 1 slot (maggiore di 625 μs) si opera nel seguente modo. Il nodo in trasmissione invia l'intero pacchetto mantenendo il canale di inizio slot (frequency hopping fermo) ma incrementando lo stesso il contatore degli slot attraversati. Una volta trasmesso il pacchetto, si riattiva il frequency hopping trascurando gli hop relativi agli slot attraversati, così che venga mantenuta la sincronizzazione con il codice di hopping degli altri dispositivi.

2.5 Il segnale Bluetooth

A differenza di molti altri sistemi di comunicazione che operano su una singola porzione di spettro in modo permanente (come ad es. la IEEE 802.11 b, g WLAN), il Bluetooth usa i suoi canali radio attraverso la tecnica di Spread Spectrum detta Frequency Hopping Spread Spectrum (FHSS).

Lo Spread Spectrum (SS) è una tecnica utilizzata nei sistemi di comunicazione, in cui il segnale viene trasmesso su una banda di frequenze che è considerevolmente più ampia di quella richiesta dal segnale modulato. Ciò ha lo scopo di migliorare l'SNR aumentando la robustezza della comunicazione nei confronti di interferenti a banda stretta. Attraverso lo Spread Spectrum è, inoltre, possibile permettere l'accesso contemporaneo, alla stesse banda di frequenze, a più utenti (schemi di accesso multiplo). Un altro impiego (di origine militare) è anche quello di impiegare tali tecniche al fine di nascondere il segnale radio trasmesso al di sotto della potenza di rumore, in modo da rendere infattibile (se non si conosce il codice di *spreading*) l'intercettazione delle comunicazioni.

La tecnica di Spread Spectrum ottenuta mediante il salto di frequenza (Frequency Hopping Spread Spectrum) è stata inventata nel 1942 dall'attrice Hedy Lamarr e dal musicista George Antheil (brevetto USA N. 2.292.387). La scoperta fondamentale di Lamarr e Antheil, fu che la trasmissione di onde radio poteva essere fatta rimbalzare da un canale all'altro a intervalli di tempo regolari, seguendo una sequenza di salti che fosse nota soltanto alle fonte di trasmissione ed al ricevitore. Antheil suggerì di adottare, come rudimentale codice macchina, un sistema simile a quello dei rotoli di carta perforati usati nei primi pianoforti meccanici. Il progetto fu così presentato al "National Inventors Council" di Washington e brevettato l'11 agosto 1942 come "Sistema di Comunicazione Segreta - n. 2.292.387".

Al fine di fare un'analisi delle caratteristiche peculiari del segnale Bluetooth (modulazione GFSK) si introduce ora un po' di notazione relativa ai segnali modulati. Il segnali impiegati nei sistemi di trasmissione radiomobili sono di tipo passa banda intorno ad una assegnata frequenza portante fc (carrier frequency). Indicato con s(t) un segnale reale, deterministico e ad energia finita, dotato di trasformata di Fourier S(f), s(t) si dice passa-banda intorno alla frequenza f_c se il suo spettro S(f) è concen-

trato intorno a f_c e, inoltre, la banda di S(f) è piccola rispetto a f_c. Tale segnale può quindi essere rappresentato come segue:

$$s(t)=a(t)\cos(2\pi f_c t+\theta(t))$$

dove a(t) ∈ ℝ+ rappresenta l'inviluppo reale di s(t) e θ(t) ∈ ℝ+ è la fase di s(t). La precedente può anche essere posta nella forma:

$$s(t)=s_c(t)\cos(2\pi f_c t)-s_s(t)\sin(2\pi f_c t)$$

$$s_c(t)=a(t)\cos(\theta(t))$$

$$s_s(t)=a(t)\sin(\theta(t))$$

dove le funzioni reali (componenti in fase e quadratura) $s_c(t)$ e $s_s(t)$ sono dette componenti analogiche di bassa frequenza rispetto a f_c. Da quest'ultime è possibile ricavare l'inviluppo reale a(t) e la fase θ(t) mediante le formule seguenti:

$$a(t)=\sqrt{s_c^2(t)+s_s^2(t)} \qquad \theta(t)=\arctan\left(\frac{s_s(t)}{s_c(t)}\right)$$

l'inviluppo complesso s(t) è quindi definito come:

$$\underline{s}(t)=s_c(t)+js_s(t)=a(t)e^{j\theta(t)}$$

così che s(t) sia anche ricavabile come:

$$s(t)=\Re[\underline{s}(t)e^{j2\pi f_c t}]$$

L'inviluppo complesso s(t) può essere sempre posto nella seguente forma, detta forma standard:

$$\underline{s}(t)=\sum_{k}\underline{u}(t-kT_s;\vec{x}(k))$$

$$\vec{x}(k)=[x(k)x(k-1)...x(k-L)]^T$$

dove $\vec{x}(k)$ rappresenta il vettore contenente gli ultimi L+1 simboli emessi dalla sorgente ed L rappresenta la lunghezza di memoria. La funzione u(...) invece rappresenta un segnale di banda base a valori complessi e di durata che può essere anche superiore ad un periodo di segnalazione. Un formato di modulazione digitale in forma standard si dice lineare quando l'impulso base u(...) è fattorizzabile in:

$$\underline{u}(t-kT_s;\vec{x}(k))=\underline{p}(t-kT_s)\psi(\vec{x}(k))$$

dove p(t) è un segnale deterministico ad energia finita che non dipende da $\vec{x}(k)$, mentre $\psi(\vec{x}(k))$ è una funzione scalare complessa del vettore $\vec{x}(k)$ che non dipende da t ed è lineare in $\vec{x}(k)$. Un formato di modulazione digitale è detto privo di memoria se L = 0, quindi:

$$\underline{s}(t)=\sum_{k}\underline{u}(t-kT_s;x(k))$$

In Bluetooth (modalità Basic Rate) viene impiegato un formato di modulazione non lineare e con memoria (L=2, quindi a risposta parziale) detto Gaussian Frequency Shift Keying (GFSK). Tale formato è sostanzialmente una modulazione Frequency Shift Keying (FSK) alla quale viene applicato un filtro di pre-modulazione gaussiano allo scopo di aumentare l'efficienza spettrale.

L'efficienza spettrale η_B di un formato di modulazione digitale, descrive la capacità di uno schema di modulazione di convogliare un flusso (binario) con rate R_b, erogato dalla sorgente, occupando la minore banda possibile. Indicata con B la banda occupata dal segnale modulato passabanda, l'efficienza spettrale di un formato di modulazione è definita

come segue:

$$\eta_B = \frac{R_b}{B}$$

e misura, in (bit/s/Hz), la quantità di informazione trasferita dal segnale digitale in esame nell'unità di tempo e nell'unità di banda. Un altro importante parametro di valutazione delle prestazioni di un formato di modulazione digitale è dato dall'efficienza in potenza. Tale parametro presenta la forma seguente:

$$\eta_P = \frac{1}{SNR}$$

ovvero è dato dall'inverso del rapporto segnale a rumore (SNR) che è necessario avere in ingresso al demodulatore al fine di ottenere un determinato valore della probabilità di errore. Esso quindi offre una misura della capacità del formato di modulazione di preservare la fedeltà del messaggio erogato dalla sorgente.

Il formato di modulazione digitale FSK, o a spostamento di frequenza, è di tipo non lineare, privo di memoria ed a inviluppo costante. In uno schema di modulazione FSK, il bit "0" viene rappresentato da un tono a frequenza f_0 mentre il bit "1" viene rappresentato da un tono a frequenza f_1. Tali armoniche vengono mantenute per un tempo pari a T_s (tempo di segnalazione):

$$T_s \simeq \frac{1}{R_b} \log_2 2$$

I modulatori FSK impiegano un Voltage Controlled Oscillator (VCO). Dapprima i bit di informazione vengono trasformati in uno stream binario antipodale {-1, 1} (No-Return to Zero, NRZ), successivamente, si applica un filtraggio gaussiano con banda normalizzata pari a $B_N = B_G T_s = 0.5$, ovvero di lunghezza pari a 2 periodi di segnalazione T_s. Il VCO traduce poi l'ampiezza della sequenza antipodale filtrata in uno shift di frequenza. L'applicazione di un filtro di premodulazione gaussiano per-

mette di aumentare l'efficienza spettrale (ovvero di ridurre la banda) smussando (*pulse shaping*) le variabilità repentine del segnale rettangolare generato dal Sample-and-Hold in fase di creazione della sequenza antipodale.

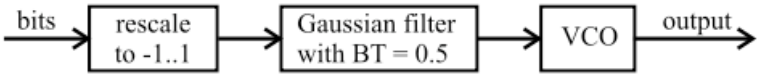


Fig. 2.5.1: Pulse shaping gaussiano nella modulazione GFSK

Questo filtraggio produce un incremento del Bit Error Rate (BER) ma diminuisce anche il rumore per via della riduzione di banda del segnale. L'aumento del BER può essere spiegato pensando al fatto che il filtraggio gaussiano riduce l'energia per bit E_b, essendo:

$$E_b = P_s\, t$$

Riprendendo la rappresentazione relativa alla banda normalizzata B_N del filtro gaussiano, si vede come l'energia per bit E_b nel caso di B_N=0.5 (Bluetooth) sia minore che in sistemi con modulazione MSK (Minimum Shift Keying, MSK) che presentano un filtro di forma rettangolare (definiti a risposta piena) con B_N tendente a infinito.

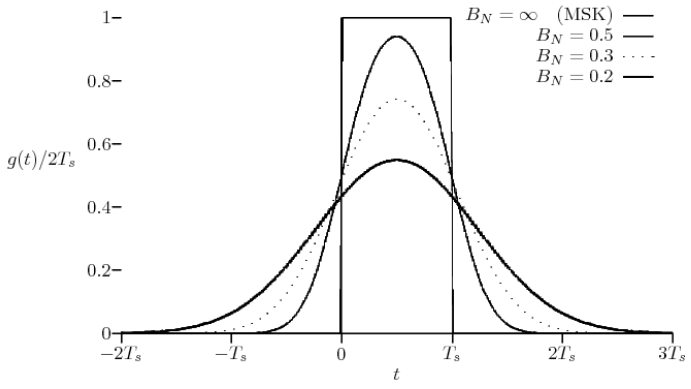


Fig. 2.5.2: Durata dell'impulso gaussiano rispetto al tempo di simbolo Ts

L'indice di modulazione è invece compreso tra 0.28 e 0.35 (attraverso ricevitori SDR è stato trovato sperimentalmente un valore di 0.32, come si vedrà più avanti). L'indice di modulazione h (spesso indicato anche con μ) in un segnale FSK è dato da:

$$h = \frac{2f_d}{R_b} = 2f_d T_s$$

con f_d deviazione di frequenza, R_b bitrate, T_s il tempo di simbolo. In accordo con la formula precedente, lo schema di modulazione GFSK del Bluetooth, presenta uno shift f_d dalla frequenza centrale fc del canale compreso tra 140 KHz e 175 KHz. Il rate di simbolo (*baud rate*) è pari a 1 *MS/s*. Per trasmissioni in Basic Rate (GFSK binario), che sono anche le più comuni, ciò produce un bit rate di 1 Mb/s. In caso di trasmissioni in Enhanced Data Rate (EDR) il bit rate può raggiungere 2 Mb/s (con modulazione π/4-DQPSK) o anche 3 Mb/s (con modulazione 8DPSK).

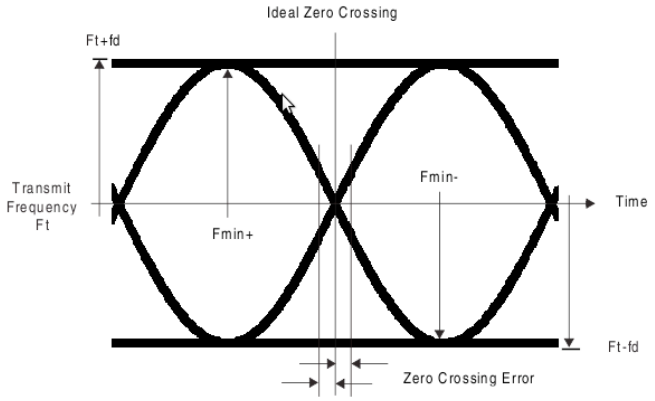


Fig. 2.5.3: Shift di frequenza (fd) con shape gaussiano (GFSK)

Nel GFSK a 2 livelli (binario), come detto, deviazioni positive maggiori o uguali a +115 *KHz* dalla frequenza centrale f_c del canale, rappresentano il bit "1" mentre deviazioni negative (< -115 *KHz*) rappresentano il bit "0" (Fig. 2.5.3). Lo Standard IEEE specifica che, per ogni trasmissione, la minima deviazione ammessa dalla frequenza centrale $F_{min} = \{|F_{min}-|, F_{min}+\}$ non può essere minore del 80% (in valore assoluto) della deviazione f_d rispetto alla frequenza di trasmissione (f_t è la frequenza centrale del canale). Inoltre, il valore minimo per f_d non deve mai scendere al di sotto di

115 *KHz*.

Le emissioni a radiofrequenza in banda ISM 2.4 *GHz* ammesse dallo standard IEEE 802.15.1 devono rispettare i limiti imposti dal FCC (Federal Communications Commission) statunitense (FCC Rules Part 15 relative to "Unlicensed Spread Spectrum radio systems", FCC Part 15.247).

Scelto un canale Bluetooth 'M' sul quale trasmettere, devono essere rispettati alcuni limiti sulla potenza emessa nei canali adiacenti. Chiamato 'N' il canale adiacente ad 'M', si ha il seguente insieme di limitazioni.

Frequency offset	Transmit power
± 500 kHz	–20 dBc
2MHz (|M – N| = 2)	–20 dBm
3MHz or greater (|M – N| ≥ 3)	–40 dBm
NOTE—If the output power is less than 0 dBm, then, wherever appropriate, the FCC's 20 dB relative requirement overrules the absolute adjacent channel power requirement stated in this table.	

Fig. 2.5.4: Valori di potenza man mano che ci si allontana dalla portante

La tolleranza relativa al valore della frequenza centrale è fissato a ±75 kHz prima di trasmettere. Il *drift* ammissibile in frequenza durante la trasmissione segue, invece il seguente schema, variabile a seconda della lunghezza del pacchetto (1, 3, 5 slot).

Duration of packet	Frequency drift
Maximum length 1-slot packet	± 25 kHz
Maximum length 3-slot packet	± 40 kHz
Maximum length 5-slot packet	± 40 kHz
Maximum drift rate[a]	400 Hz/µs

[a]The maximum drift rate is allowed anywhere in a packet.

Fig. 2.5.5: Drift in frequenza al variare della lunghezza dei pacchetti

Per il ricevitore, invece, si prevede una sensibilità di -70 dBm. La potenza ricevuta deve comunque garantire garantire un Bit Error Rate (BER) di 0.1% (10^{-3}). Per segnali GFSK con indice di modulazione h = 0.32 in un canale radio di tipo AWGN (Additive White Gaussian Noise), il rapporto

segnale-rumore (SNR) richiesto per ottenere un BER dello 0.1% è circa 12.5 dB.

L'approccio seguito in questo lavoro è stato quello di cercare tra le diverse tecniche di analisi dei segnali quella che risultasse più semplice, ma anche più flessibile. La flessibilità richiesta è essenzialmente rappresentata dalla capacità di poter adattare il medesimo approccio al sensing di diverse tecnologie radio (WiFi, ZigBee, Bluetooth). Sono state studiate le peculiarità di numerose tecniche di analisi operanti sul segnale nel dominio del tempo, della frequenza e le tecniche di analisi congiunta tempo-frequenza (*TF analysis*) (vedi [3] [4] [5] [6] [7] [8] Cap. 4). Un interessante esempio di analisi congiunta tempo-frequenza è dato dalla trasformata di Wigner-Ville (WV). Tale trasformazione permette di ottenere un'ottima risoluzione nei due domini (tempo e frequenza) al costo di una maggiore complessità rispetto alle tecniche tradizionali di analisi in frequenza (basate sulla FFT) o in tempo (ad es. energy detection).

A partire da studi precedenti [1, Cap4], è emerso che allo scopo di determinare la presenza di *primary users* può essere sufficiente e vantaggioso dal punto di vista della complessità elaborativa, analizzare l'output di un semplice Energy Detector (ED). In questo modo, estraendo alcune caratteristiche del pattern di invio e ricezione dei pacchetti o della loro durata, si possono ottenere *features* specifiche di quel protocollo MAC ovvero della tecnologia radio impiegata.

Il limite, infatti, delle altre tecniche studiate è risultato essere quello di offrire buone prestazioni solo nel riconoscimento di un ristretto numero di modulazioni ed, inoltre, alle spese di una elevata complessità elaborativa. L'Energy Detector, invece, si è dimostrato molto più flessibile e di facile implementazione.

Questo approccio verrà meglio illustrato dopo aver approfondito le caratteristiche della radio SDR impiegata in questo lavoro: la radio Universal Software Radio Peripherial (USRP), oggetto del prossimo capitolo.

3 GNUradio e l'USRP

Il crescente interesse verso le Software Defined Radio (SDR) ha prodotto, negli ultimi anni, la necessità di sviluppare soluzioni hardware in grado di garantire ottime prestazioni nell'elaborazione (DSP) e in particolare nella conversione analogico/digitale. Una radio SDR è caratterizzata da una elevata flessibilità di configurazione, alte velocità di calcolo, elevata larghezza di banda (capacità di trattare segnali a banda larga) e precisione (rappresentare fedelmente i campioni del segnale) ovvero un buon *dynamic range* (rapporto tra il massimo e il minimo valore rappresentabili).

I più recenti sviluppi nell'elettronica digitale (ADC, DAC, FPGA), hanno permesso di produrre microprocessori in grado di garantire le prestazioni richieste dalle radio SDR. I nuovi ADC/DAC consentono, infatti, di campionare segnali a frequenze sempre più elevate. Ciò rende possibile, in alcuni casi, la conversione diretta di segnali passa banda nel dominio numerico (ad es. per i segnali radio FM). Attraverso veloci ADC/DAC diviene possibile “avvicinare” il dominio numerico a quello analogico (antenna e front-end TX/RX) con il vantaggio di sfruttare l'efficienza dei potenti algoritmi di elaborazione dei segnali (Digital Signal Processing, DSP) in grado di semplificare e velocizzare il trattamento dei segnali stessi. La radio Software Defined si prefigge lo scopo di sfruttare al massimo le possibilità offerte dagli ADC/DAC e gli algoritmi di DSP per ottenere radio configurabili via software altamente efficienti e versatili. La valutazione di un sistema SDR può risultare davvero complessa ed, in generale, dipende fortemente dall'applicazione di interesse.

Come già accennato, si tratta sistemi hardware *general purpose* in grado di simulare qualsiasi sistema di comunicazione per mezzo di moduli software in grado di modificare le operazioni svolte dall'hardware stesso. Alcuni validi parametri, anche se non esaustivi, per una prima valutazione di un sistema SDR sono il clock degli ADC/DAC presenti, la precisione della rappresentazione dei campioni o anche il dynamic range e, ovviamente, il costo. Infatti l'impiego di veloci ADC/DAC e di FPGA in grado di eseguire rapidamente complessi calcoli di DSP, si ripercuote direttamente sul costo del sistema. Nella rapida rassegna che segue, si da un'idea della varietà dei sistemi hardware attualmente impiegati per applicazioni SDR.

In ambito militare il progetto della Difesa statunitense Joint Tactical Radio System (JTRS, pronunciato “jitters”) è un sistema SDR in grado di interoperare con molteplici sistemi radio preesistenti sia militari che civili. La sua adozione come mezzo di comunicazione standard per le forze armate USA è prevista a partire dall'anno 2010. Tale sistema si basa sull'architettura Software Communications Architecture (SCA) che rappresenta un framework voluto dal Dipartimento della Difesa degli Stati Uniti (DoD) per la progettazione di software e hardware SDR.

In ambito civile, sono noti i sistemi: FlexRadio Systems SDR-1000, FLEX-5000, SoftRock RXTX, HPSDR, USRP. Le migliori performance sono ottenibili mediante l'hardware FlexRadio FLEX-5000 (il quale dispone di convertitori ADC e DAC a 24 bit operanti a 192 *MS/s*) per un costo totale che si aggira intorno ai 3000 dollari (USD). Allo stesso modo, si sono sviluppati anche prodotti di fascia più bassa, come ad esempio l'SDR SoftRock v9.0 (intorno ai 50$USD). Un altro esempio è dato dal progetto HPSDR (High Performance SDR) il cui costo per un sistema completo si aggira intorno a 500$USD ed è in grado di offrire buone prestazioni per la maggior parte delle applicazioni (ad esempio il modulo Mercury presenta un ADC a 16 bit, 122.88MS/s).

Un altro valido prodotto è dato dall'hardware denominato USRP (Universal Software Radio Peripheral). L'USRP è stato ideato da Matt Ettus e successivamente sviluppato da un gruppo di collaboratori (Ettus Research LLC, fondata nel 2004). L'obbiettivo di questo progetto è stato sin da subito quello di offrire un prodotto di basso costo pur mantenendo buone prestazioni (ADC a 64MS/s, DAC a 128 *MS/s*). L'USRP viene attualmente impiegato da numerosi gruppi di ricerca ed appassionati in tutto il mondo. Tale successo ha portato all'acquisizione della società Ettus Research LLC da parte di National Instruments nel Febbraio 2010. La radio USRP è strettamente legata al progetto GNUradio ideato da Eric Blossom nel 2001. GNUradio è un Software Development Kit (SDK) opensource specificatamente designato per lo sviluppo di sistemi SDR. Esso si compone di diversi moduli in grado di interoperare e produrre complesse elaborazioni di Digital Signal Processing (DSP).

3.1 Descrizione generale dell'USRP

La radio USRP prevede attualmente 2 versioni chiamate: USRP e USRP2. La prima versione dell'USRP è in grado di campionare il segnale ricevu-

to (ADC) ad una velocità di 64MS/s con risoluzione pari a 12 bit (in trasmissione il DAC opera a 128 *MS/s* con 14 bit di precisione). Tuttavia, a causa dell'adozione di una connessione USB 2.0 (con throughput lordo pari a 480 Mbit/s, che si traduce in 32 MB/s effettivi) verso il calcolatore (*host*), l'USRP presenta delle limitazioni nella larghezza di banda che può essere effettivamente analizzata dall'*host* al quale è collegato. Tale collo di bottiglia sarà analizzato più avanti.

La seconda versione dell'USRP, chiamata USRP2, è stata creata con lo scopo di risolvere i problemi evidenziati dalla versione precedente. Ciò è ottenuto adottando, ad esempio, convertitori ADC/DAC più performanti (100 *MS/s* a 14 bit per l'ADC, 400 *MS/s* a 16 bit per il DAC) ed un'interfaccia Giga-Ethernet verso l'host.

USRP	USRP 2
4 ADC, 64 *MS/s*, 12 bit	2 ADC, 100 *MS/s*, 14 bit
4 DAC, 128 *MS/s*, 14 bit	2 DAC, 400 *MS/s*, 16 bit
FPGA Altera Cyclone	FPGA Xilinx Spartan 3-2000
Cypress FX2 USB2 controller	Gigabit Ethernet interface
4 slot (2 RX / 2 TX)	2 slot (1 RX / 1 TX)

Per comprendere meglio il ruolo dell'hardware e il suo funzionamento in una Software Defined Radio è necessario introdurre ora alcune nozioni relative al campionamento di un segnale complesso (*quadrature sampling*). Nella trattazione che segue si farà riferimento al caso della ricezione nell'USRP di un segnale passa banda essendo lo scenario di utilizzo dell'SDR in questo lavoro.

Il campionamento di segnali complessi (rappresentabili mediante componenti in fase e quadratura) è una operazione fondamentale nei moderni sistemi di comunicazione numerica. Tale processo consiste nel campionare il segnale analitico ottenuto generando i rami fase e quadratura mediante trasformazione di Hilbert. Alla conversione del segnale ricevuto passa-banda da analogico a numerico (*analog to digital conversion*, ADC) segue la traslazione del relativo spettro in banda base. Quest'ultima è realizzata mediante mixing con la sinusoide complessa $e^{-j2\pi fct}$, dove $-f_c$ sta ad indicare l'ammontare della traslazione verso l'origine (pari alla frequenza portante f_c). Posta f_s la frequenza di campionamento, si ha che

lo spettro del segnale viene ad essere riportato in banda base assumendo, tuttavia, un andamento periodico di periodo f_s come mostrato dalla figura seguente.

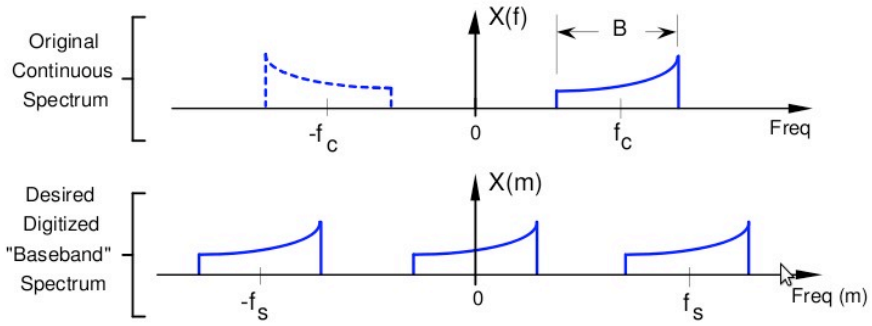


Fig. 3.1.1: Spettro del segnale reale passa-banda e campionamento

Tale fenomeno è spiegato ripercorrendo brevemente i passi dal campionamento allo spettro del segnale campionato. Campionando a passi di T secondi un segnale tempo-continuo x(t) si ha:

$$\hat{x}(t)=x(t)\sum_{m=-\infty}^{\infty}\delta(t-mT)$$

Il prodotto precedente nel dominio della frequenza diviene una convoluzione, ovvero:

$$\hat{X}(j\omega) = \frac{1}{2\pi}[X(j\omega)] * [\frac{2\pi}{T}\sum_{m=-\infty}^{\infty}\delta(\omega-\frac{2\pi}{T}m)]$$

$$\hat{X}(j\omega) = \frac{1}{T}\sum_{m=-\infty}^{\infty}X[j(\omega-\frac{2\pi m}{T})] = X(e^{j\omega T})$$

L'ultima uguaglianza permettere di scrivere una relazione tra il segnale x(t) e il segnale campionato x(kT) nel dominio della frequenza. Tale relazione mostra che lo spettro del segnale campionato x(kT) non è altro che la somma di infinite repliche traslate dello spettro del segnale x(t) di una quantità pari alla frequenza di campionamento.

$$\omega_c = \frac{2\pi}{T} \quad [\text{rad/sec}]$$

Lo schema seguente, che descrive tale operazione, mostra come a partire da un segnale passa-banda $x_{BP}(t)$, si pervenga ad una coppia di segnali in fase e quadratura. Nel diagramma è anche rappresentato il passaggio dal dominio analogico (*continuos*) a quello numerico (*discrete*) realizzato per mezzo dell'ADC.

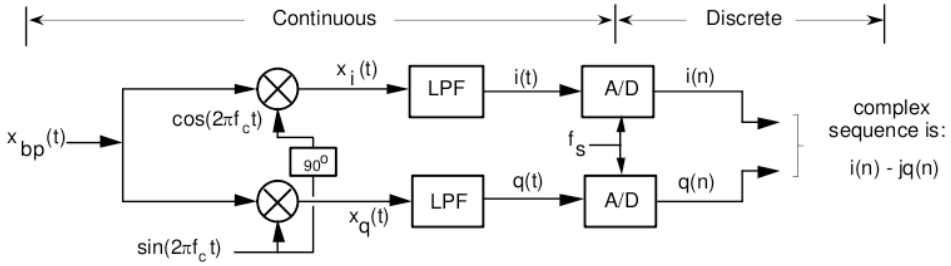


Fig. 3.1.2: Schema di campionamento I&Q di un segnale passa-banda

Dopo il mixing, come intuibile dallo spettro appena visto, è necessario effettuare un filtraggio passa-basso (LPF) per estrarre solo una replica (in banda base) del segnale in fase e di quello in quadratura. Sui rispettivi rami poi è operata la conversione ADC, la quale, nell'USRP avviene ad un ritmo di 64 milioni di campioni complessi al secondo, rappresentati con una precisione di 12 bit.

Più in dettaglio la rappresentazione seguente mostra lo spettro dei segnali sui rami in fase e in quadratura e i successivi passi prima di giungere al segnale numerico in uscita dall'ADC.

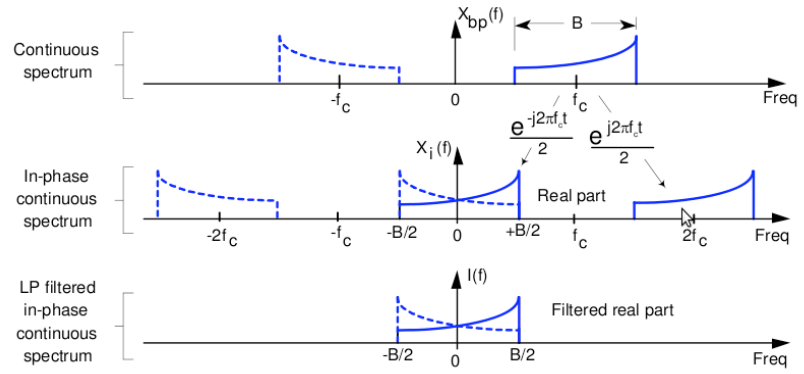


Fig. 3.1.3: Segnale passa-banda moltiplicato per un coseno di frequenza f_c

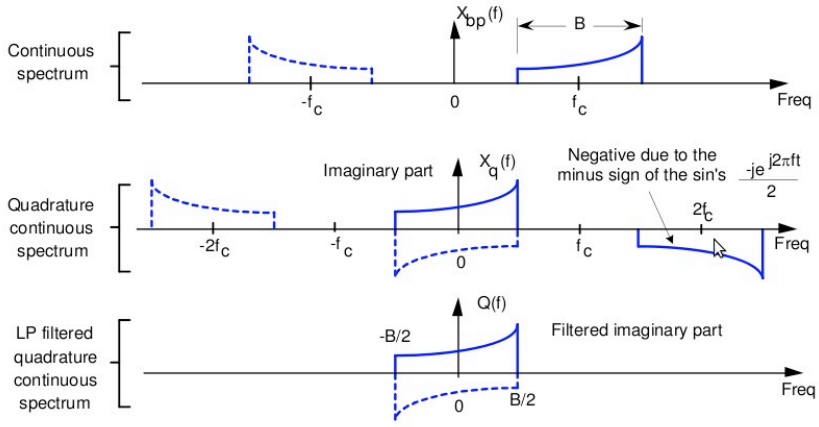


Fig. 3.1.4: Segnale passa-banda moltiplicato per un seno di frequenza f_c

Lo spettro risultante del segnale passa-banda $x_{BP}(t)$ è dato dalla differenza tra lo spettro del segnale sul ramo in fase e quello sul ramo in quadratura.

$$\mathbf{I(f) - j\ Q(f)}$$

dove I(f) e Q(f) sono ottenute mediante filtraggio passa-basso (LP filter) come illustrato nella figura seguente.

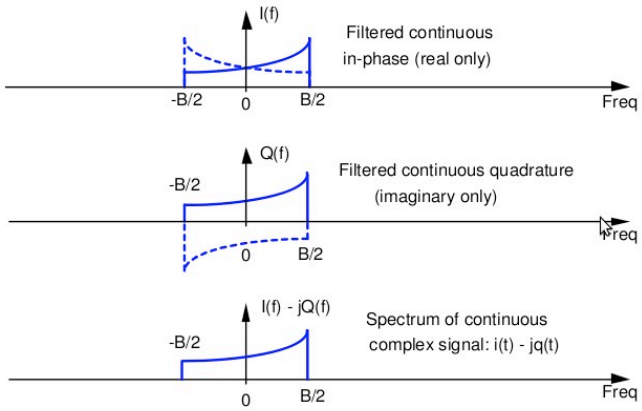


Fig. 3.1.5: Spettro della differenza dei segnali fase e quadratura

L'ultima operazione è quella della conversione ADC. Il risultato è ancora una volta uno spettro periodico ma relativo alla sequenza ottenuta dal campionamento dei segnali sui due rami in fase e quadratura.

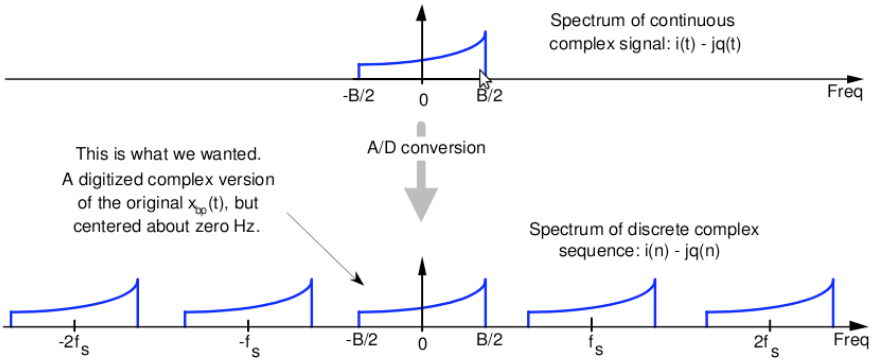


Fig. 3.1.6: Effetto sullo spettro della conversione Analogico/Digitale (A/D)

La figura 3.1.6 rappresenta lo spettro del segnale campionato che, a meno di un filtraggio passa-basso, rappresenta lo spettro voluto ovvero $x_{BP}(t)$ campionato e riportato in banda base.

La scelta di impiegare un campionamento complesso permette di sfruttare una importante peculiarità di questo approccio riguardante la nota formula di Nyquist. Un segnale infatti, secondo Nyquist, è rappresentabile in modo esatto mediante i suoi campioni solo se questi vengono presi con un ritmo f_s maggiore o uguale a due volte la banda del segnale B ($f_s >= 2B$).

Nel caso di impiego di campioni complessi tuttavia, ciascun campione complesso porta con sé due numeri reali (la sua parte reale e la parte immaginaria) che possono essere visti come campioni distinti del segnale stesso. Ciò produce un raddoppio della banda di Nyquist ovvero consente di impiegare ritmi di campionamento pari a f_s per rappresentare una banda B. Quanto detto resta valido in fase di ricezione ma le medesime procedure si applicano anche in trasmissione mediante l'impiego di un DAC in luogo dell'ADC.

Si vedrà ora come i concetti sopra esposti vengano implementati dalla radio USRP. In quello che segue si effettuerà una descrizione mirata delle caratteristiche peculiari dell'USRP (versione 1) con particolare attenzione agli stadi di ricezione e trattamento del segnale. Quanto detto varrà, in larga misura, anche per la radio USRP2 data la struttura del tutto simile dei moduli principali.

3.2 Universal Software Radio Peripheral (USRP)

L'Universal Software Radio Peripherial (USRP) è un dispositivo hardware con interfaccia USB 2.0, in grado di ricevere e trasmettere segnali a radiofrequenza sfruttando elevate capacità di campionamento e ricostruzione dei segnali ricevuti (ADC/DAC) e la capacità di calcolo di un comune PC (host), per realizzare una Software Defined Radio (SDR) [1].

La struttura dell'USRP è essenzialmente divisa in due parti. Un primo stadio si occupa di trattare il segnale analogico ricevuto in antenna (*daughterboard*) e un successivo stadio campiona tale segnale in modo da produrre sull'interfaccia USB una sequenza di campioni complessi ad un fissato rate e con data precisione (*motherboard*). Su tali campioni vengono poi applicati (ad esempio attraverso il tool GNUradio) gli algoritmi di Digital Signal Processing (DSP) necessari all'applicazione di interesse [2].

La motherboard dell'USRP ospita gli elementi principali dell'SDR, come: il processore Field Programmable Gate Array (FPGA) Altera Cyclone EP1C12, il controller USB 2.0 Cypress FX2, 4 convertitori ADC (64MS/sec, 12 bit) e 4 convertitori DAC (128MS/sec, 14 bit), 2 slot per il BUS di trasmissione TX e 2 slot per la ricezione RX. Nella figura seguente si possono identificare i vari blocchi principali che concorrono a formare l'architettura della motherboard dell'USRP. E' evidente che l'FPGA è il processore centrale di questo sistema che ha il compito di gestire il flusso di dati da e verso le daughterboard. In questo processore vengono

svolte importanti pre-elaborazioni (come ad esempio la digital down-conversion, DDC) che rendono possibile il trasporto verso l'host attraverso il canale USB e le successive elaborazioni software (DSP).

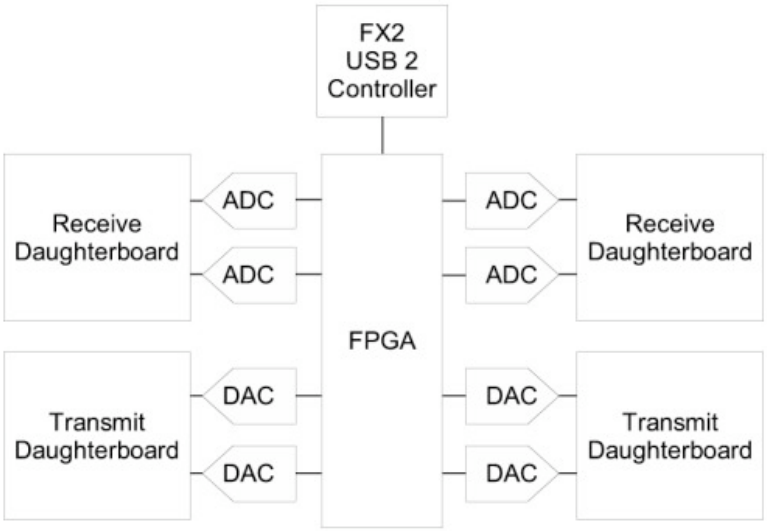


Fig. 3.2.1: Schema a blocchi della motherboard dell'USRP

La disposizione degli slot per le daughterboard prevede una coppia TX/RX per il lato A e un'altra coppia TX/RX per il lato B. Tale motherboard, assieme alla eventuale daughterboard installata, è montata in uno chassis metallico che offre sostegno anche all'antenna.

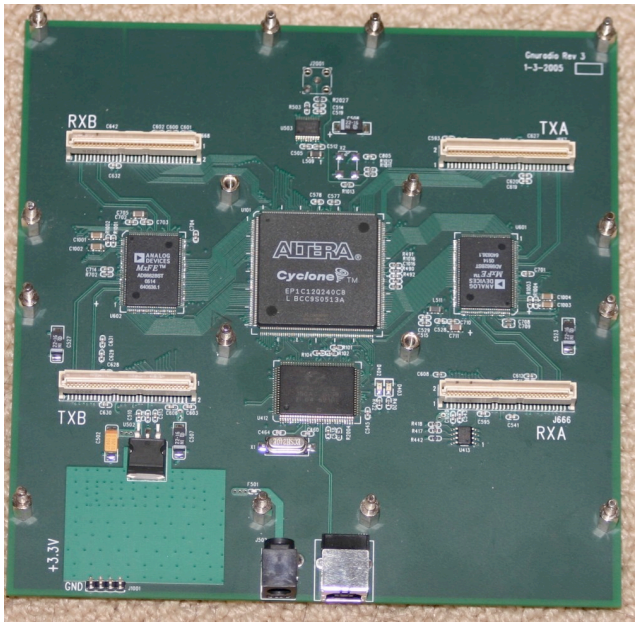


Fig. 3.2.2: Un'immagine della motherboard dell'USRP

Quando sono attivi più canali, ad esempio 4 canali in ricezione, i campioni che giungono all'FPGA dai 4 ADC vengono interallacciati. Ad esempio da 4 canali complessi (I + jQ) si ottiene la sequenza I0 Q0 I1 Q1 I2 Q2 I3 Q3 I0 Q0, ecc. In tal caso il rate da ciascun canale (tasso di decimazione) deve essere lo stesso ed, inoltre, il data rate aggregato dei 4 canali deve risultare inferiore o uguale al *throughput* del canale USB ovvero 32Mb/s.

L'USRP può lavorare in *full-duplex* in quando i bus in trasmissione sono indipendenti da quelli in ricezione. In tal modo i data rate nelle due direzioni (TX/RX) possono essere anche diversi.

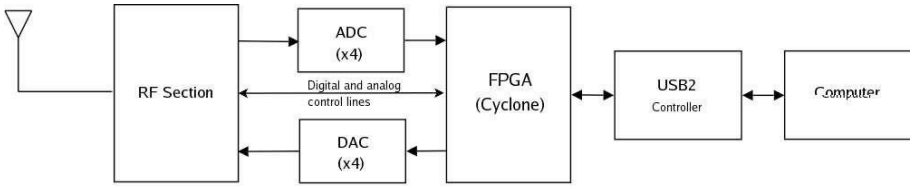


Fig. 3.2.3: Schema a blocchi dell'USRP end-to-end

Una notevole flessibilità di impiego è garantita dal fatto che la *motherboard* non presenta filtri di anti-aliasing o ricostruzione e quindi non fis-

sa ad un particolare valore la frequenza di ricezione. La banda di interesse può così essere centrata e gestita attraverso l'impiego una opportuna *daughterboard* (RF section) che ha il compito di implementare lo stadio RF e quindi i relativi filtraggi antialiasing o di ricostruzione. L'insieme di *chassis*, una o più antenne (per applicazioni MIMO), motherboard e, una o più, *daughterboards* compongono l'URSP completo (Fig. 3.2.4).

Fig. 3.2.4: L'Universal Software Radio Peripherial

Scelta la banda di interesse e la relativa antenna, si invia il segnale ricevuto alla *daughterboard* operante in tale banda (la RFX2400 per la banda ISM). Nella motherboard dell'USRP sono presenti due slot per daughterboards in trasmissione e due slot per la ricezione. Esistono tuttavia daughterboards in grado di trasmettere e ricevere e quindi occupanti le socket TX e RX contemporaneamente (è il caso della scheda RFX2400).

La conversione da segnale analogico a numerico (ADC) e viceversa (DAC) è prodotta da due processori Analog Devices AD9862 (Mixed-Signal Front-End Processor), ciascuno dei quali possiede uno stadio ADC e uno DAC. L'ADC del processore Analog Devices AD9862 è in grado di campionare un segnale ad un rate di 64 *MS/s* con precisione di 12 bit. Lo stadio DAC lavora invece a un ritmo di 128 *MS/s* a 14 bit. Sia in ricezione (ADC) che in trasmissione (DAC), si hanno due canali per poter campionare (o ricostruire) separatamente fase e quadratura del segnale analitico in input oppure due segnali reali o un solo segnale reale. L'ADC lavora sfruttando il campionamento in fase e quadratura ovvero impiegando la traformata di Hilbert per la rappresentazione di segnali reali (segnale analitico). La figura che segue illustra proprio tale approccio.

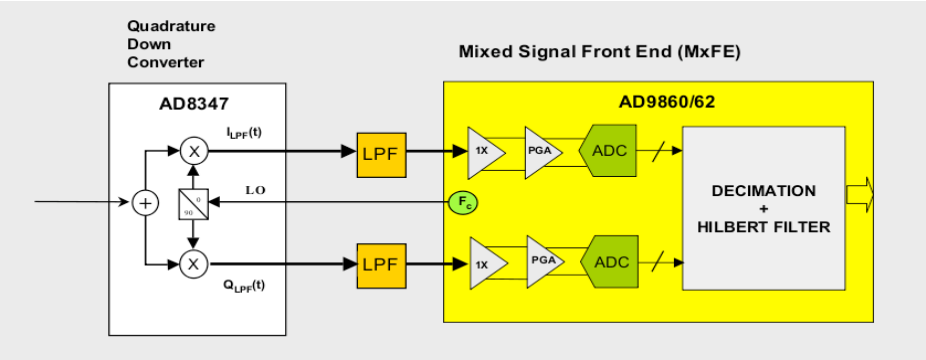


Fig. 3.2.5: Blocchi costituenti l'Analog to Digital Converter (ADC) dell'USRP

Un componente molto importante per l'USRP è l'FPGA Altera Cyclone EP1C12 montata sulla motherboard. In esso infatti avvengono le operazioni più critiche. I DAC e gli ADC sono direttamente connessi con l'FPGA. In ricezione, i campioni (rami in fase e quadratura) provenienti dall'ADC, entrano nell'FPGA per essere elaborati dal primo blocco, il Digital Down Converter (Complex DDC). Il clock di questo processore lavora ad una frequenza di 64 *MHz*. Per chiarire il ruolo del DDC si può fare riferimento alla figura seguente.

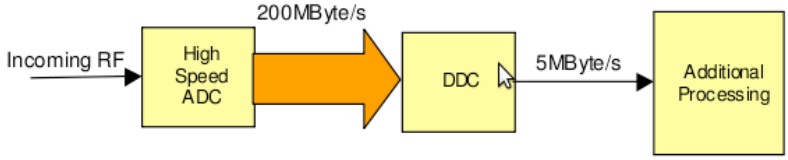


Fig. 3.2.6: Ruolo del Digital Down Converter in ricezione

Per esempio, si assuma in ingresso un segnale ad RF di 1 *MHz* di banda modulato attorno ad una portante di 40 *MHz*. Attraverso l'impiego di un ADC a 100 *MS/s* è possibile campionare in modo esatto tale segnale (frequenza di Nyquist pari a 50 *MHz*). Se si assume una precisione di 16 bit (2 byte/campione) tale operazione produce un flusso di 200 MByte/s all'uscita dell'ADC. Tale flusso, nel caso di canale di trasporto USB, risulta chiaramente troppo elevato. In realtà il valore di 200 MB/s è largamente superiore a quello realmente richiesto per acquisire il segnale di 1 *MHz* di banda e nei sistemi analogici ciò comporta in genere l'impiego di schemi di ricezione ad eterodina. Il DDC,tuttavia, offre un sistema più sem-

plice e flessibile (perché numerico) per risolvere questo problema. Ciò avviene riducendo la banda a quella sufficiente a rappresentare il segnale di interesse (ad es. 2.5 *MHz* per il segnale scelto) portando il flusso in ingresso al DSP da 200 MByte/s a soli 5 Mbyte/s attraverso una operazione detta di decimazione.

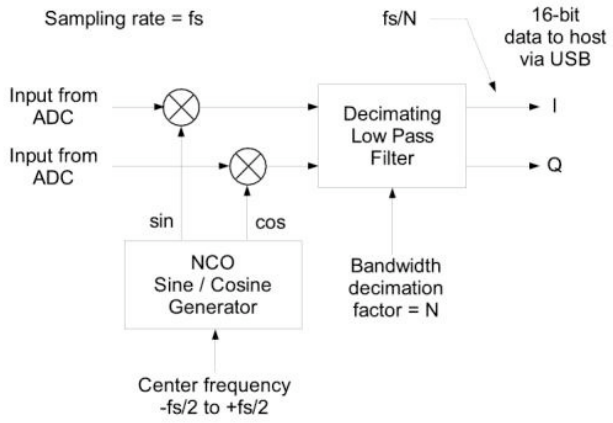


Fig. 3.2.7: Struttura interna del DDC all'interno dell'FPGA

Il DDC permette quindi di portare il segnale passa-banda ricevuto in banda base riducendo il rate di campionamento a quello minimo in grado di consentire una corretta ricostruzione nonché velocizzare l'elaborazione successiva. Il *Numerical Controlled Oscillator* (NCO) permette, attraverso l'algoritmo COordinated Rotation DIgital Computer (CORDIC), di effettuare operazioni trigonometriche senza l'ausilio di moltiplicatori hardware.

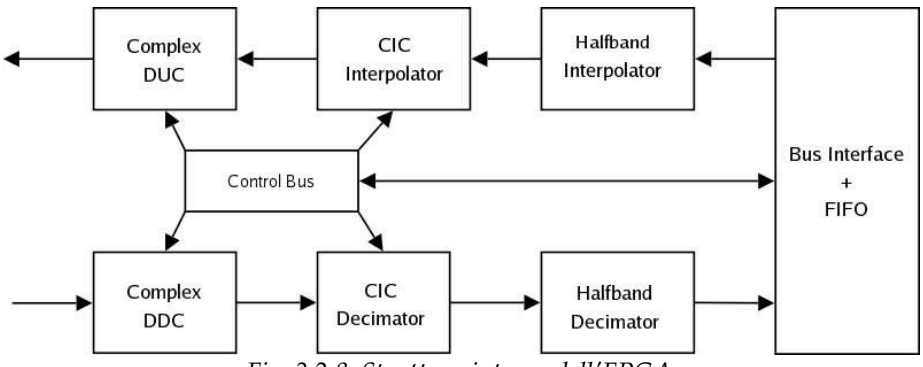


Fig. 3.2.8: Struttura interna dell'FPGA

Si analizza ora più in dettaglio il funzionamento del DDC all'interno dell'FPGA. Il DDC effettua due operazioni fondamentali: un filtraggio passa-basso (*low pass filter*) e un *downsampling*. Entrambi queste operazioni sono realizzate da un filtro CIC (Cascaded Integrator-Comb filter). Il filtro CIC è stato introdotto per la prima volta da Eugene B. Hogenauer nel 1981 [4][5]. Questo genere di filtri si presta molto bene ad una implementazione in hardware ed offre elevate prestazioni nella reiezione del fenomeno dell'aliasing. Il CIC è un filtro FIR (*finite impulse response*) che può integrare un'operazione di decimazione oppure di interpolazione (CIC Decimator e CIC Interpolator). In ricezione, che è quella che si analizzerà ora, si impiega un CIC decimatore. Un filtro CIC è composto da due componenti base: un integratore (passa-basso) e un filtro a pettine (nel quale, in ricezione, avviene la decimazione).

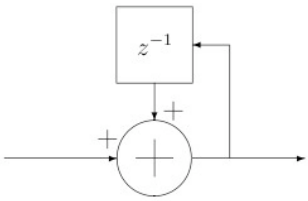


Fig. 3.2.9: Filtro IIR (integratore)

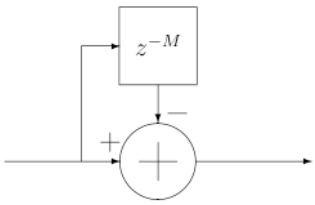


Fig. 3.2.10: Filtro FIR (un "dente" del filtro a pettine)

Chiamati x[n] l'ingresso e y[n] l'uscita, per il filtro integratore, si ha la seguente equazione alle differenze:

$$y[n]=y[n-1]+x[n]$$

che presenta un feedback unitario. Nel dominio della trasformata Z, la precedente assume la forma:

$$H_I(z)=\frac{1}{1-z^{-1}}$$

ovvero la risposta di un filtro accumulatore (filtro IIR a un solo polo). Nel caso dell'elemento base di un filtro comb posto a valle del downsampler, si ha invece:

$$y[n]=x[n]-x[n-RM]$$

dove R è il fattore di down sampling, M è un parametro di progetto chiamato *differential delay*. Nel dominio Z, la precedente diviene:

$$H_C(z)=1-z^{-RM}$$

Per costruire un CIC decimatore si opera combinando questi due elementi base, ottenendo una cascata di N integratori che lavorano alla frequenza di campionamento fs, seguita da un downsampler (con fattore R) e altri N elementi del filtro a pettine operanti ad un rate fs/R.

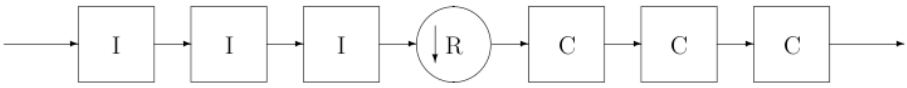


Fig. 3.2.11: Filtro CIC, stadi di integratori (I) e filtri comb (c) in cascata

La funzione di trasferimento di un filtro CIC assume così la forma seguente:

$$H(z)=H_I^N(z)\,H_C^N(z)=\frac{(1-z^{-RM})^N}{(1-z^{-1})^N}=\left(\sum_{k=0}^{RM-1} z^{-k}\right)^N$$

L'ultimo membro dell'espressione precedente mostra che pur essendo presenti N filtri di tipo IIR, il filtro CIC possiede una funzione di traferimento pari a quella di un banco di N filtri FIR. All'aumentare degli N stadi (filtri integratori e comb) migliora la capacità di reiezione dell'aliasing ed aumenta anche l'attenuazione in banda passante (*passband "droop"*). Il guadagno in continua (DC gain) è invece funzione del fattore R di decimazione.

Questa particolare struttura consente di ottenere un rate minore (di un fattore R) su metà del filtro (gli stadi *comb* del filtro CIC) così da migliorarne l'efficienza. Altro risultato è quello di aver ridotto il numero di ele-

menti necessari per il filtro a pettine. Infine, si è ottenuta una struttura flessibile, in quanto indipendente dal particolare fattore di decimazione R (*multirate filtering*). In trasmissione la cascata di elementi componenti il filtro CIC è invertita ponendo i filtri *comb* prima degli integratori.

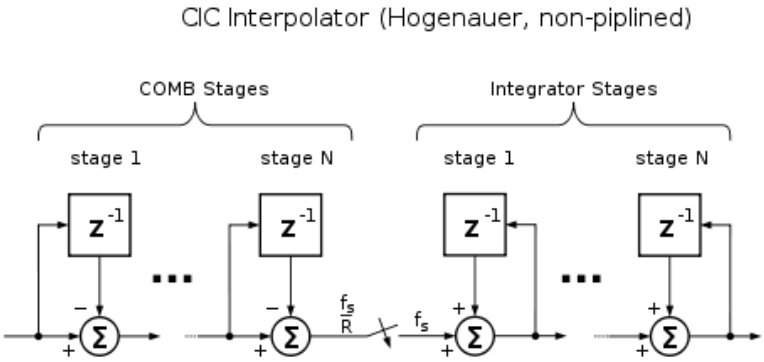


Fig. 3.2.12: Un esempio di filtro CIC interpolatore

In ricezione si ha, quindi, la seguente situazione. Supponendo che il fattore di decimazione sia R, il filtraggio passa basso riduce la banda di un fattore pari ad R ovvero $[-f_c/R, +f_c/R]$. Il sottocampionatore si occupa quindi di eliminare i campioni in eccesso al di fuori della banda filtrata riducendoli dello stesso fattore da $[-f_c, +f_c]$ a $[-f_c/R, +f_c/R]$. In tal modo, dopo il filtraggio, si è ridotta la quantità di campioni a quella sufficiente a rappresentare il segnale (mantenendo nullo l'aliasing). Un minor numero di campioni permette di rendere compatibile il flusso in ingresso verso l'host (in caso di ricezione) con il *throughput* dell'interfaccia USB gestita dal controller USB 2.0 dell'USRP (Cypress FX2) pari a 32MB/s.

I campioni complessi prodotti dall'ADC, come già detto, sono rappresentati mediante 12 bit (rami I e Q). Nell'FPGA è definito un controller del guadagno sull'input analogico (a monte dell'ADC) con scala da 0 a 20 dB. Un gain pari a 0 dB sull'input analogico ammette valore massimo (2^{11} - 1 = 2047, 1 bit per il segno) con segnale in ingresso di 2 Vpp (con il gain a 20 dB il fondo scala dell'ADC è raggiunto con un segnale di 0.2 Vpp).

I campioni complessi generati dall'ADC, prima di venire inviati al bus USB, vengono mappati dall'FPGA a 16 bit (*signed integer*) per il ramo in fase e 16 bit (signed integer) per quello in quadratura. In questa trasformazione si ha uno shift di 3 bit a sinistra che porta il livello massimo da 2047 a 16376 (2047×2^3). All'interno dell'FPGA è implementato uno stadio CORDIC (*COordinated Rotation DIgital Computer*) che modifica ulterior-

mente il livello massimo portandolo a 16376 x 1.647/2 = 13485. Ogni campione complesso che giunge al controller USB è quindi dato da 16+16 = 32 bit ovvero 4 byte per campione. Con questa configurazione, attraverso l'USB possono transitare un numero di campioni complessi pari a:

$$throughput_{USB} = \frac{32\text{MB}/sec}{4\,Bytes/Sample} = 8\,MS/s$$

Essendo i campioni presi dai rami in fase e quadratura indipendenti, la frequenza di Nyquist per il flusso a 8 *MS/s* è direttamente 8 *MHz* (*quadrature sampling*). Questo valore corrisponde alla larghezza di banda con la quale l'USRP può lavorare. In realtà il roll-off del filtro CIC nell'FPGA attenua il 25% della banda passante lasciando a disposizione circa 6 *MHz* utili.

Considerando il rate in input dai campionatori digitali ADC (per fase e quadratura), che è pari a 64 *MS/s* complessi, si ha che il minimo fattore di decimazione per evitare un collo di bottiglia (*USRP overrun*) sul canale USB è pari a:

$$decimazione_{min} = \frac{64\text{MS}/s_{ADC}}{8\text{MS}/s_{USB}} = 8$$

In molte applicazioni che coinvolgono segnali a banda stretta (ad. es. i canali Bluetooth), una banda di 8 *MHz* (6 *MHz* effettivi, per quanto detto sopra) è più che sufficiente. Nel caso del Bluetooth, ad esempio, permette il sensing di 6 canali Bluetooth (1 *MHz* di banda per canale).

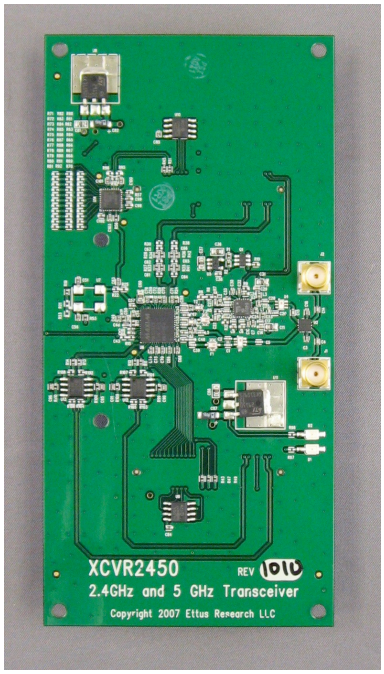


Fig. 3.2.13: La daughterboard XCVR2450 (dual band 2.4-5.8 GHz)

Al fine di poter ricevere i segnali in banda ISM 2.4 *GHz*, è stata impiegata la daughterboard Ettus Research XCVR 2450 (Fig. 3.2.13). Questa scheda è in grado di lavorare nelle bande ISM a 2.45 *GHz* e 5.8 *GHz*. La XCVR2450 presenta una potenza in trasmissione di 100mW (20dBm).

3.3 Misure sperimentali sull'USRP

Al fine di poter caratterizzare il valore dei campioni prodotti dall'USRP, si è scelto di operare una analisi agli estremi del processo di campionamento. Tale calibrazione è stata svolta considerando lineare la trasformazione operata sul segnale ricevuto dall'input RF fino ai campioni raccolti in uscita attraverso GNUradio (3.3.1).

Leggendo il datasheet del campionatore dell'USRP (AD9862) si vede che è realizzato per mezzo di un ADC lineare. Estraendo un set di misure potenza in input / livello ADC, si può risalire al calcolo del coefficiente angolare che descrive il rapporto incrementale della retta di funzionamento dell'ADC stesso. Sono state eseguite diverse misure in condizioni

di comportamento lineare e in condizioni di saturazione della dinamica dell'ADC.

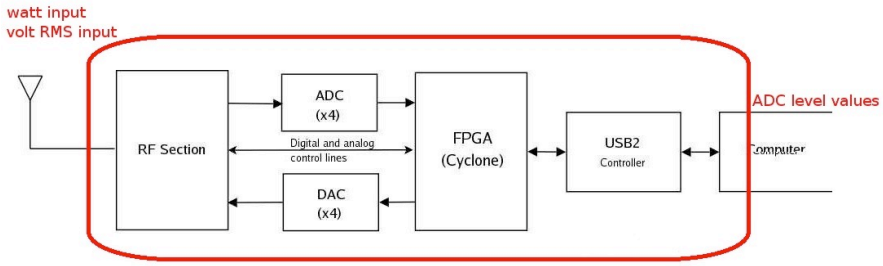


Fig. 3.3.1: Ingresso e uscita dell'USRP: input RF+ADC+FPGA

Da ulteriori indagini sulla struttura interna dell'FPGA, è stato ricavato l'effetto del passaggio dei campioni dell'ADC nei successivi blocchi. Come già detto, i campioni dell'ADC sono rappresentati mediante 12 bit (segno compreso) dai quali si ottengono valori compresi tra [-2047, +2048]. Nell'FPGA i campioni vengono mappati su 16 bit con uno shift di 3 bit che porta il valore massimo a $2047 \times 2^3 = 16376$. Il successivo stadio CORDIC (*Coordinated Rotation Digital Computer*, set di algoritmi per eseguire calcoli trigonometrici mediante sole somme e shift), ha l'effetto di diminuire questo valore di una quantità pari approssimativamente a 1.647 / 2 (16376 → 13485). Tale livello risulta essere quello di saturazione.

Lo stadio CORDIC sfrutta il concetto di rotazione di fase di un numero complesso al fine di ottenere il risultato di operazioni trigonometriche senza effettuare moltiplicazioni hardware. Ciò diviene particolarmente utile, come nel caso nell'NCO dell'USRP, quando si debbano effetuare dette operazioni in una FPGA. In tal caso, infatti, è generalmente utile risparmiare quanti più *gates* possibile evitando le moltiplicazioni in favore di somme e shift.

L'analisi effettuata per risalire alla relazione livelli/tensione di input, è stata condotta inviando mediante cavo in input all'USRP (connettore SMA) un segnale sinusoidale di cui era nota potenza, frequenza centrale e banda. Il segnale sinusoidale è stato generato mediante un generatore di segnale (Fig. 3.3.2) in grado di operare ad una frequenza f_c = 5760.89 *MHz* fissa e caratterizzato da una potenza di uscita pari a 23.2 dBm.

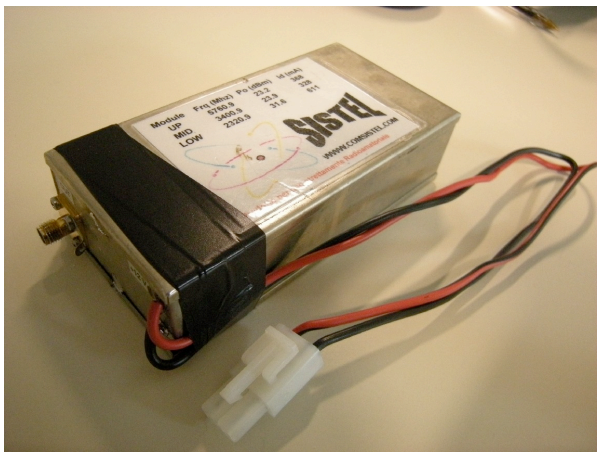


Fig. 3.3.2: Il generatore di segnale utilizzato nelle misure

Il suddetto generatore di segnale è stato collegato al USRP attraverso un attenuatore variabile (Fig. 3.3.3) con range di attenuazione 0 - 110 dB, banda di lavoro DC – 18 *GHz* e potenza massima ammissibile in input di 1W CW (C*ontinuos Wave*).

Fig. 3.3.3: Attenuatore Hewelett-Packard (0-110 dB, DC-18 GHz, 1W CW)

Il segnale prodotto dal generatore di segnale, è stato prima studiato attraverso un analizzatore di spettro per conoscerne con precisione frequenza centrale, banda e potenza a valle dell'attenuatore e dei cavi ad RF impiegati. In questo modo, si è avuta una misura precisa della potenza ricevuta dall'USRP, comprensiva anche delle perdite dovute ai cavi e i connettori. L'analizzatore impiegato per le misure è stato un Rohde&Schwarz (banda 100 kHz – 6 *GHz*, RBW 100 Hz - 1 *MHz*). Le im-

postazioni di misura sono state: *frequency span* di 1 *MHz* (larghezza canale Bluetooth) *e* una *radio bandwidth* (RBW) di 30 *kHz*. In questo modo è stata misurata la frequenza centrale del segnale prodotto dal generatore di segnale sinusoidale che è risultata essere pari a fc = 5760.89 *MHz*.

Impostando questo stesso valore nel blocco USRP source del GRC, è stato possibile mediante FFT sink visualizzare e valutare l'offset dell'oscillatore locale dell'USRP. Tale offset è risultato essere pari a 69.17 *kHz*. Questo valore è stato registrato e sottratto dalle successive misurazioni.

Il valore di input gain dell'USRP è stato posto a 0dB. In questo modo si è evitato di introdurre l'effetto del Programmable Gain Amplifier (PGA in Fig. 3.3.4).

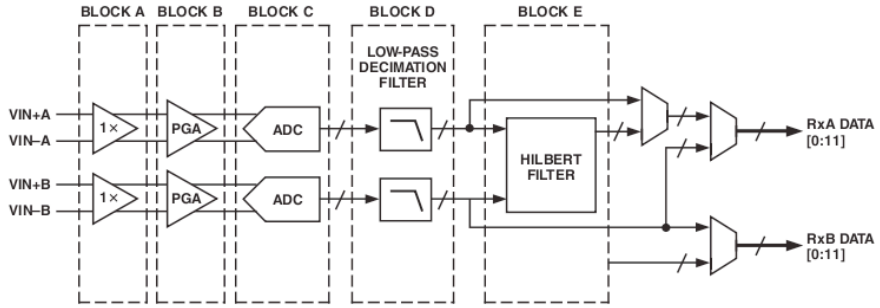


Fig. 3.3.4: Blocchi dello stadio di ricezione del campionatore I&Q (AD9862)

A questo punto è stato possibile registrare i campioni raccolti dall'USRP quando all'ingresso è stato posto il segnale sinusoidale precedentemente caratterizzato. Il blocco USRP *source* è stato impostato con una frequenza centrale pari a:

```
frequenza segnale in input – offset USRP + 150 kHz
```

In tal modo, l'USRP ha restituito in banda base i campioni (nei rami in fase e quadratura) di un tono sinusoidale di frequenza 150 kHz (Fig. 3.3.5), corrispondente allo shift del GFSK Bluetooth. Il valore dei campioni complessi è stato anch'esso stampato in modo da poter attivare la funzione di *peak hold* su di esso. In questo modo è possibile registrare l'ampiezza del segnale in input espressa in livelli del quantizzatore (ADC).

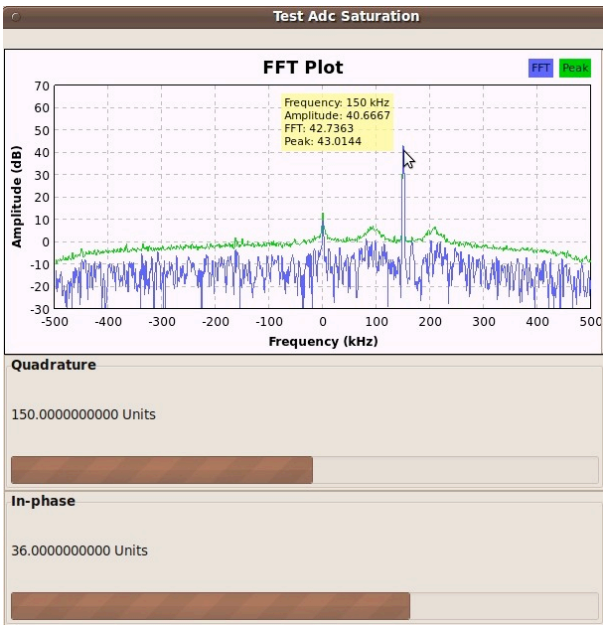


Fig. 3.3.5: FFT plot e valore istantaneo dei campioni per un segnale sinusoidale

Il grafico di figura 3.3.6, è stato ottenuto variando l'attenuazione applicata al segnale sinusoidale (P_o = 23.2 dBm) ottenendo un set di potenze in input all'USRP comprese tra -70 dBm e -25.5 dBm. Assumendo un impedenza di ingresso pari a 50 ohm, è stato possibile calcolare le tensioni di input (nella Fig. 3.3.6 comprese tra 0.1 mV e 11.9 mV RMS).

Attenuation (dB)	Input power (dBm)	Input voltage mV (RMS)	Output level (value)
90	-70,0	0,071	25
85	-65,0	0,126	35
80	-60,5	0,211	50
75	-55,0	0,398	85
70	-50,5	0,668	135
65	-45,1	1,243	255
60	-40,5	2,111	415
55	-35,1	3,931	800
54	-34,2	4,360	900
53	-33,0	5,006	1010
52	-32,1	5,552	1110
51	-31,2	6,159	1215
50	-30,4	6,753	1320
49	-29,3	7,665	1520
48	-28,5	8,404	1710
47	-27,4	9,539	1920
46	-26,5	10,580	2140
45	-25,5	11,871	2420

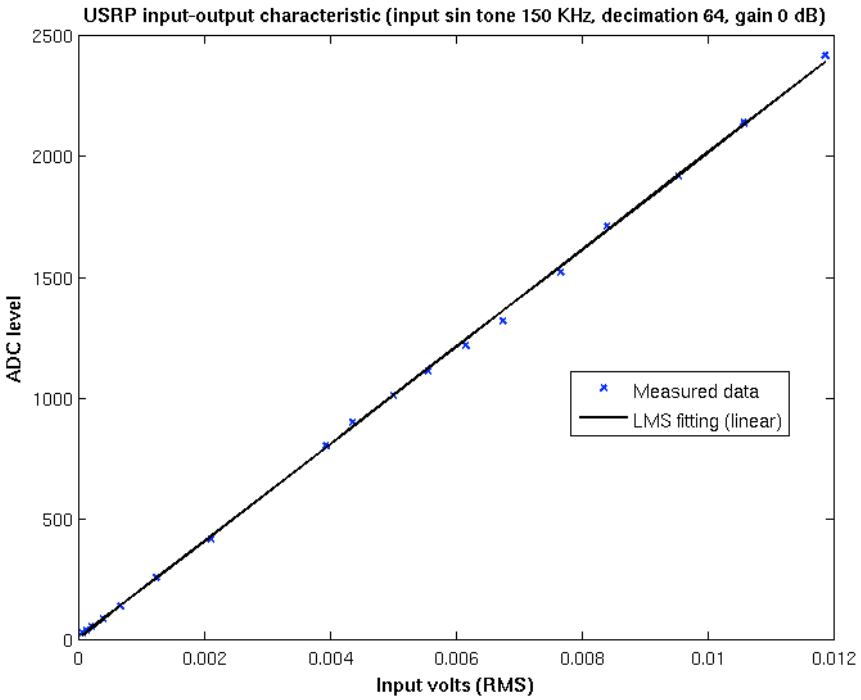


Fig. 3.3.6: Caratteristica ingresso-uscita dell'USRP

Impiegando le funzioni standard di MATLAB *polyfit* e *polyval* si sono potuti calcolare i coefficienti del polinomio di grado N in grado di interpolare le misure minimizzando l'errore quadratico medio (Least Mean Square, LMS). Approssimando quindi le misure di Fig. 3.3.6 con una retta (grado 1, N = 1) si è ottenuto il coefficiente angolare che è risultato pari a 2,012919 x 10^5. Tale valore è stato poi utilizzato per trasformare il valore dei campioni ricevuti da livello ADC a valore RMS di tensione.

Continuando ad aumentare la potenza in input fino ad un valore di -11 dBm si è studiato il comportamento dell'USRP in condizioni di saturazione.

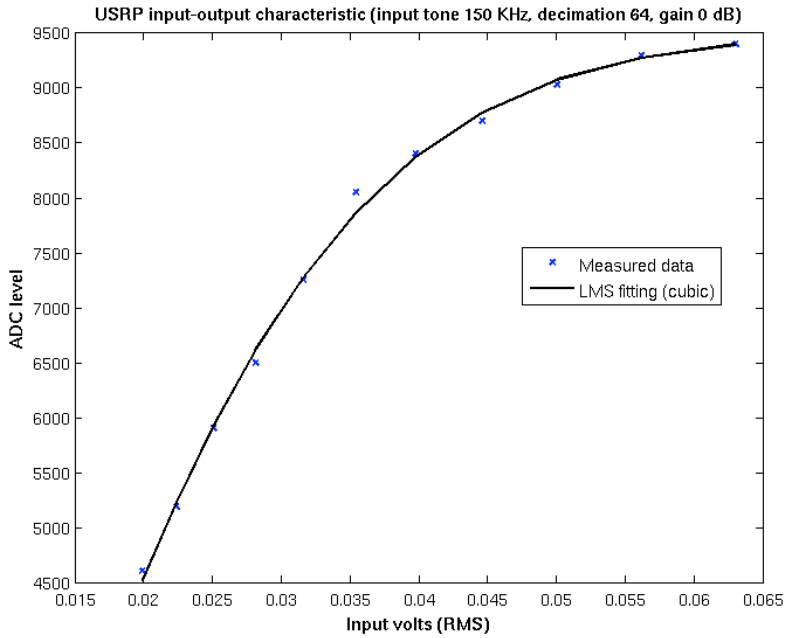


Fig. 3.3.7: Caratteristica ingresso-uscita dell'USRP (saturazione)

Il grafico di figura 3.3.7, mostra il comportamento dell'USRP (ADC + successivi filtri CIC, Hilbert ecc.) quando in input il segnale raggiunge il valore di 63 mV (corrispondente al livello 9400 per l'ADC con input gain a 0 dB). Chiaramente, in queste condizioni operative, non è possibile sfruttare quanto detto in condizioni di linearità e quindi diviene impossibile risalire al valore di tensione in input ed all'energia misurata.

3.4 Misure sperimentali sull'USRP2

In questa sezione si descrivono i risultati ottenuti con l'USRP2. Questa radio SDR, come già descritto, presenta diversi miglioramenti rispetto alla versione precedente. Nell'USRP2 infatti si hanno a disposizione ADC/DAC a 100 *MS/s* (14 bit) e 400 *MS/s* (2 ADC + 2 DAC) che permettono di raggiungere una banda in ricezione di 25 *MHz*.

I risultati che seguono sono stati ottenuti mediante misure eseguite con gli stessi strumenti già descritti nel par. 3.4. L'obiettivo è stato quello di ottenere una caratterizzazione dell'ADC lineare dell'USRP (LTC2284). In queste misure si è cercato di ottenere il valore del coefficiente angolare della retta che descrive il rapporto tra livelli in output e la tensione in input in condizioni di linearità.

Impiegando lo stesso setup sperimentale del par. 3.4 si è scelto di misurare i livelli in uscita per un range di potenze in input compreso tra [-50, -5.66] dBm con attenuazione crescente a passi di 5 dB. I valori di potenza in input sono stati misurati mediante l'analizzatore di spettro R&S a disposizione. Il livello in uscita, registrato nei campioni prodotti dall'USRP2, è stato misurato mediante *peak hold* del valore numerico istantaneo (andamento sinusoidale), così da ottenere un valore di livello relativo al corrispondente valore di picco della tensione in input.

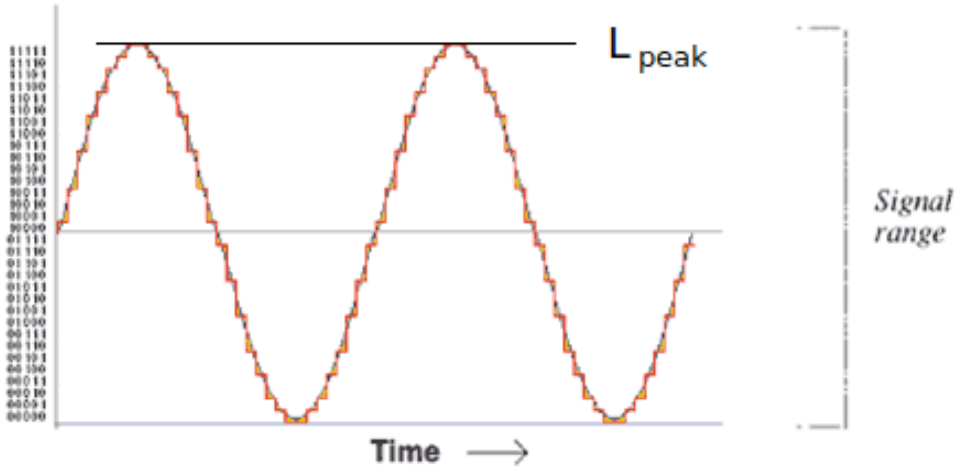


Fig. 3.4.1: Quantizzazione della sinusoide in ingresso all'USRP2

Il valore di picco L_{peak} in uscita dall'USRP2 risulta proporzionale al valore di picco della tensione in ingresso sull'antenna. La Fig. 3.4.2 mostra la relazione ingresso-uscita tra tensione in ingresso e livello registrato e visualizzato mediante GNUradio. L'andamento di questa caratteristica è chiaramente lineare e quindi in tali condizioni risulta agevole estrarre il coefficiente angolare della retta e l'eventuale intercetta. Nel grafico di Fig. 3.4.2, inoltre, si è scelto di indicare il valore RMS della tensione ovvero, per un input sinusoidale, pari al valore di picco diviso per √2.

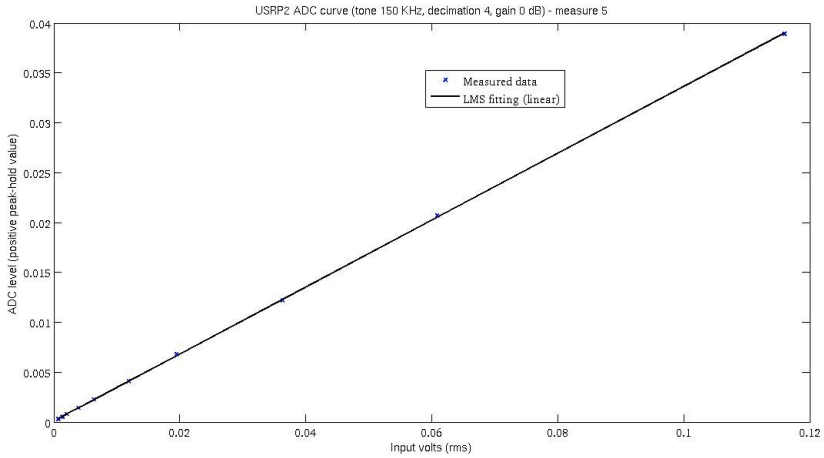


Fig. 3.4.2: Caratteristica ingresso-uscita nell'USRP2

Dalle misure ottenute e dal successivo fitting lineare (LMS) si è ottenuto un coefficiente angolare pari a 0.3351577, con una intercetta di valore pari a 0.0001.

Con questi dati si può facilmente ottenere il valore di tensione in input e da esso è facile poi calcolare l'energia a breve termine finestrando periodicamente il segnale ricevuto.

$$E_N(\boldsymbol{v}) = \sum_{i=1}^{N} |v_i|^2 \cdot T_S$$

Nella precedente N è la lunghezza della finestra temporale per il calcolo dell'energia, v_i rappresenta il campione i-esimo di tensione all'interno della finestra considerata e T_S il periodo di campionamento. Il calcolo dell'energia sarà ulteriormente approfondito nel Cap. 4.

Un'altra sessione di misure ha condotto a verificare l'effetto del guadagno introdotto per mezzo dei *Programmable Gain Amplifier* (PGA) sui rami I&Q in banda base e le relative rette in uscita (Fig. 3.4.3).

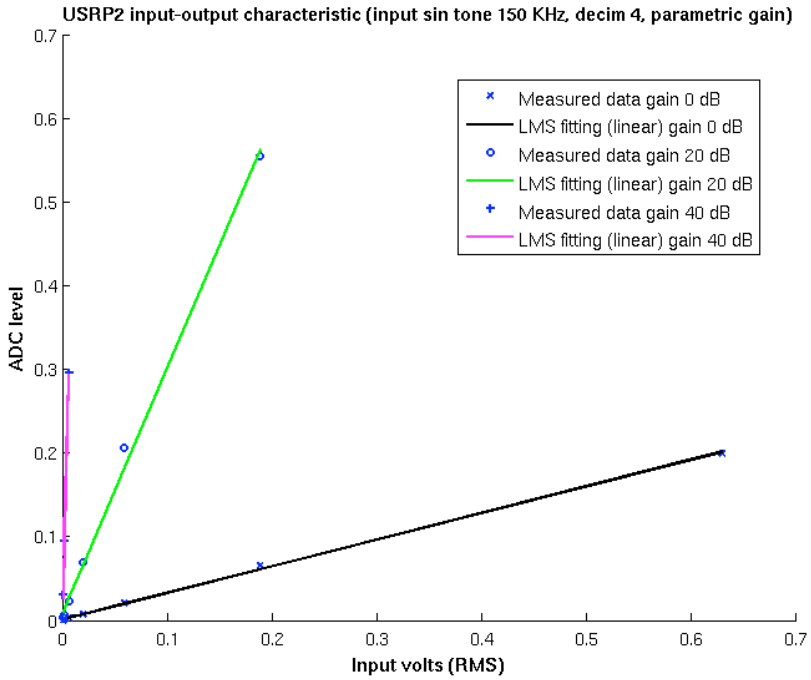


Fig. 3.4.3: Caratteristica ingresso-uscita nell'USRP2 (gain 0, 20, 40 dB)

I coefficienti angolari delle rette ottenute da queste misure sono risultati essere quelli della tabella seguente. Il valore di tali coefficienti permette di risalire al valore RMS della tensione in input sull'antenna.

PGA GAIN [dB]	USRP2 ADC GRADIENT
0	3.790980e-01
20	2.933752e+00
40	4.592423e+01

Attraverso questo lavoro di calibrazione è stato possibile ottenere un parametro di conversione per ottenere dei campioni di valore quanto più vicino alla tensione di input allo stadio a RF. Ovviamente si tratta di una misura che risulta valida per lo specifico hardware usato, e che quindi, necessita di essere condotta per ciascun USRP impiegato nello *spectrum sensing*.

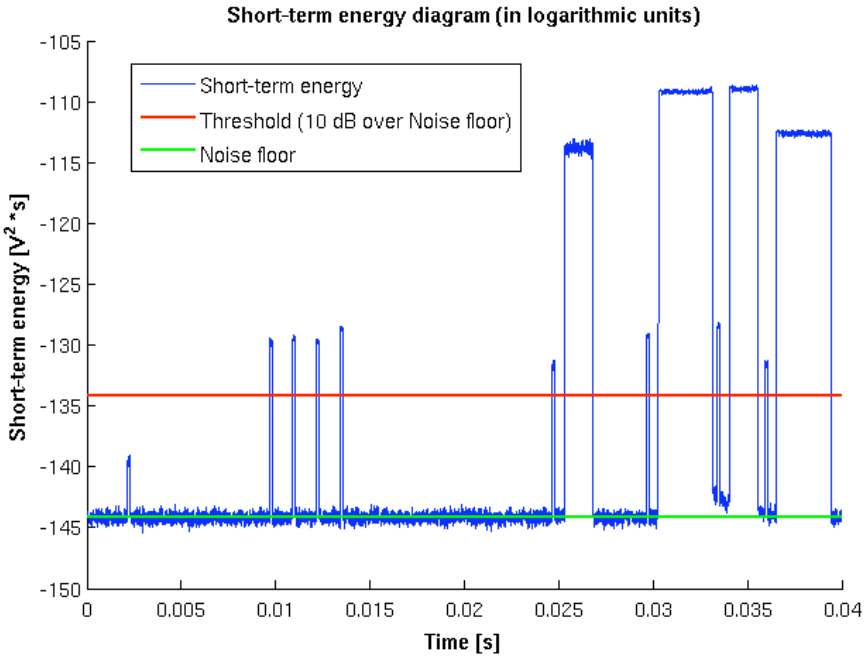


Fig. 3.4.4: Energy detection con l'USRP2 (25 MHz) e relativo Noise Floor

Con questi test, si è evidenziato il comportamento dell'USRP e dell'USRP2, al fine di ottenere dei valori di tensione in ingresso utili al calcolo dell'energia. In Fig. 3.4.4 vi è un esempio del diagramma prodotto dal calcolo dell'energia a breve termine (su finestre da 250 campioni), considerando la calibrazione appena descritta. Nel prossimo paragrafo, si proseguirà con la descrizione di un importante strumento, utile per sviluppare moduli software che realizzano funzioni di DSP per la radio USRP, chiamato GNUradio.

3.5 Il tool di sviluppo GNUradio

GNUradio è un ambiente di sviluppo software che offre un framework per la definizione di sistemi radio SDR attraverso due potenti e flessibili linguaggi object-oriented: il C++ ed il Python. La progettazione di radio SDR consiste nel realizzare moduli software in grado di svolgere le funzioni che nelle radio tradizionali sono svolte da elementi hardware come, ad esempio: la codifica, la modulazione, ecc. L'obbiettivo è quello di modificare le funzioni di un ricetrasmettitore attraverso dei moduli software [1].

Un sistema di questo tipo è in grado di offrire una elevata flessibilità di

configurazione, in quanto attraverso un aggiornamento software, è possibile cambiare radicalmente la struttura interna (modulazione, codifica, ecc.) della radio SDR. Attraverso i moduli software si possono definire le forme d'onda in trasmissione, e quindi anche demodularle, attraverso l'impiego di algoritmi di Digital Signal Processing (DSP). L'elaborazione richiesta da una radio SDR segue un approccio diverso da quello impiegato con hardware analogico perché si basa sull'immediato campionamento del segnale ricevuto (idealmente svolto sul segnale in antenna) e successiva elaborazione numerica. Globalmente il risultato finale non cambia, la SDR è una radio in grado di trasmettere e ricevere segnali modulati a radiofrequenza come le radio tradizionali. La versatilità di una radio SDR, tuttavia, permette di impiegare il medesimo hardware per molteplici applicazioni e studiare una grande varietà di soluzioni consentendo, ad esempio, un rapido debug di nuovi protocolli e nuovi sistemi di comunicazione.

Il software GNUradio si inserisce in tale contesto come un'ambiente di sviluppo per creare Software Defined Radio. Attraverso il linguaggio Python, GNUradio permette di connettere tra loro, più blocchi funzionali definiti in C++, simulando in tal modo l'intera catena trasmissiva dalla sorgente all'antenna e viceversa. L'ambiente di sviluppo GNUradio presenta una vasta libreria di funzioni (blocchi predefiniti) le quali possono essere utilizzate per definire rapidamente radio SDR (vedi APPENDICE B).

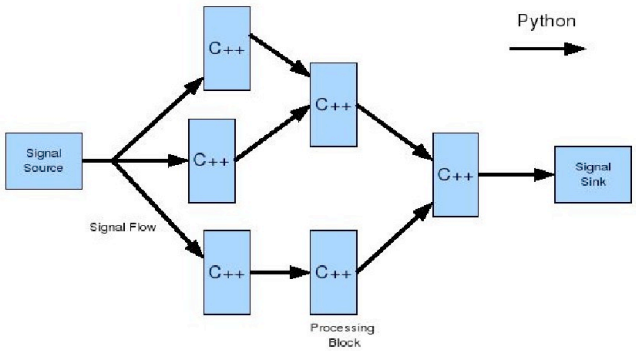


Fig. 3.5.1: Struttura di un flow graph di GNUradio

GNUradio è distribuito su licenza GPL (GNU General Public License) ovvero è possibile avere accesso al codice sorgente e partecipare allo sviluppo del progetto stesso. La caratteristica di essere *opensource*, ha con-

sentito la rapida diffusione di GNUradio nella ricerca sulle radio SDR e presso le comunità di radioamatori.

GNUradio può essere utilizzato in modo indipendente, oppure congiuntamente ad un hardware SDR (come ad es. l'USRP) o anche sfruttando la capacità di campionamento e ricostruzione di una semplice scheda audio.

L'utilizzo congiunto di GNUradio e USRP permette di realizzare ricetrasmettitori SDR e verificare con segnali reali l'effettivo funzionamento del codice sviluppato. Dal punto di vista architetturale, una radio SDR in ricezione è scomponibile nei seguenti sotto sistemi ovvero in ricezione:

Antenna –> Front-end RX –> ADC –> Software SDR

e in trasmissione:

Software SDR –> DAC –> Front-end TX –> Antenna

Come si vede la quasi totalità delle operazioni di trattamento dei segnali ricevuti e trasmessi è svolta dal software SDR. E' importante sottolineare, tuttavia, che gli elementi che compongono i *front-end* in trasmissione e in ricezione non sono ancora configurabili via software a causa di limitazioni hardware relative degli attuali filtri di aliasing/ricostruzione e degli ADC/DAC. Tuttavia non appena saranno disponibili tecnologie hardware digitali in grado di campionare a velocità ancora più elevate e buone risoluzioni sarà possibile campionare direttamente il segnale passabanda e successivamente impiegare un *Digital Down Converter* (DDC) con filtraggio opportuno in modo da realizzare radio *all-digital* dove la componentistica analogica sia quasi o del tutto scomparsa.

Un importante applicativo contenuto in GNUradio è dato dallo GNUradio Companion o GRC. Tale applicativo consente di definire rapidamente, mediante una interfaccia grafica intuitiva, complessi sistemi SDR, impiegando sia i blocchi predefiniti che quelli definiti dall'utente.

Fig. 3.5.2: Uno screenshot di GNUradio Companion (GRC)

Attraverso l'impiego di blocchi predefiniti come, ad esempio, l'USRP Source o l'FFT sink, in GRC è molto facile realizzare molteplici funzioni senza dover necessariamente metter mano al codice. Inoltre, definendo i blocchi per l'elaborazione dei segnali (DSP) ricevuti dall'USRP, si possono realizzare complesse funzioni che, una volta connesse tra loro, siano in grado di realizzare il processing necessario per la ricetrasmissione dei segnali.

Come già detto, GNUradio offre un framework per lo sviluppo di software defined radio attraverso un modello ibrido di programmazione che include il Python e il C++. Il linguaggio Python viene impiegato per definire, in modo semplice, il grafo che connette i vari blocchi che svolgono funzioni di signal processing. Il linguaggio C++ viene impiegato per la definizione dei blocchi offrendo così elevate prestazioni per le operazioni time critical. Questi due linguaggi si interfacciano per mezzo di SWIG (Simplified Wrapper and Interface Generator) che offre la possibilità di gestire il flusso dati che attraversa i blocchi (visti all'esterno come black box) mediante il Python. Grazie a SWIG i blocchi definiti in C++ possono essere istanziati direttamente all'interno dell'applicazione Python. Lo script Python che gestisce il flusso dati impiega uno scheduler per gestire l'attivazione dei vari blocchi e tanti buffer quanti sono i flussi che attraversano i blocchi. Il GRC offre una interfaccia grafica (GUI) al processo di creazione del grafo. In uscita esso produce il codice Python necessario a realizzare l'elaborazione prodotta dai blocchi posti in cascata. Tale script Python assume il nome dell'ID impostato nel blocco “Options” (nell'esempio è stato scelto “myblock”). Una volta definito il grafico e prodotto lo script Python è possibile lanciare lo script sia at-

traverso GRC che attraverso l'interprete Python da Shell.

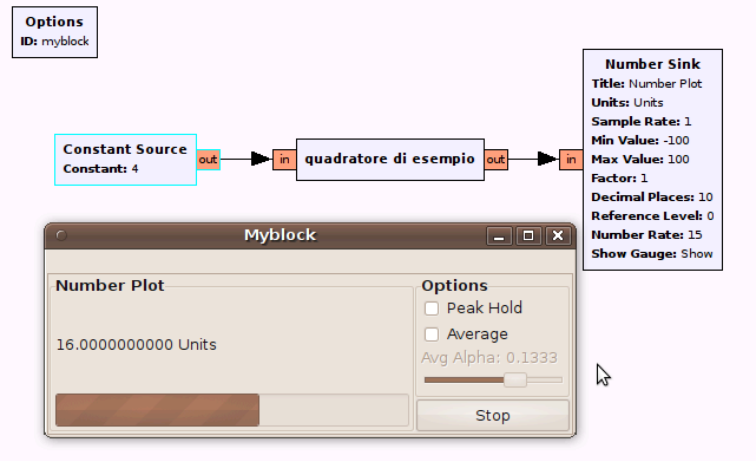


Fig. 3.5.3: Esempio di utilizzo di GNUradio companion

La figura mostra un esempio di blocco “quadratore di esempio” il quale, dopo essere stato definito in C++, è stato integrato in GRC. In questo modo connettendo al quadratore una sorgente costante di tipo opportuno (di tipo float in questo esempio) all'input del blocco quadratore, si ottiene in uscita (number sink) il quadrato del valore in input.

Analizzando più in dettaglio le varie fasi dello sviluppo di moduli SDR mediante GNUradio, si devono affrontare i seguenti passi: la definizione del diagramma di flusso in Python, la definizione di un blocco di elaborazione dei dati in C++ ed, eventualmente, l'importazione del blocco in GRC.

Il diagramma di flusso (flow chart) è definito in Python e contiene i blocchi e le relative connessioni tra di essi. Un valido esempio per analizzare la struttura di un codice Python in Gnuradio è fornito da Eric Blossom in “Exploring GNUradio” [1] ed è relativo alla definizione di un sistema per la rirpoduzione del dial tone statunitense (due toni a 400 Hz e 350 Hz).

```
#!/usr/bin/env python

from gnuradio import gr
from gnuradio import audio
```

```
def build_graph():
    sampling_freq = 48000
    ampl = 0.1

    fg=gr.flow_graph()
    src0=gr.sig_source_f(
            sampling_freq,
            gr.GR_SIN_WAVE, 350, ampl)

    src1=gr.sig_source_f(
            sampling_freq,
            gr.GR_SIN_WAVE, 440, ampl)

    dst=audio.sink(sampling_freq)
    fg.connect((src0, 0), (dst, 0))
    fg.connect((src1, 0), (dst, 1))

    return fg

if __name__ == '__main__':
    fg = build_graph ()
    fg.start ()
    raw_input ('Press Enter to quit: ')
    fg.stop ()
```

Lo script Python di un grafo GNUradio inizia sempre con l'importazione dei moduli necessari all'esecuzione del programma. In questo caso con

```
from gnuradio import gr
from gnuradio import audio
```

è possibile importare dal modulo gnuradio i metodi base di gnuradio (gnuradio.gr) e quelli per la gestione dell'interfaccia della scheda audio (gnuradio.audio). Successivamente si procede alla definizione del metodo build_graph() nel quale viene creato il flow graph. Dopo aver definito le variabili sampling_freq e ampl, il metodo standard gr.flow_graph()

consente di istanziare un oggetto flow graph nel quale verranno importati i vari blocchi e rispettivi collegamenti.

In ogni flow chart, in genere, sono definiti almeno una sorgente (signal source), un blocco di elaborazione (signal processing) ed una destinazione (signal sink) per il flusso di dati. Nell'esempio precedente, una delle 2 sorgenti è data dalla seguente riga di codice.

```
src0=gr.sig_source_f(freq_camp,tipo_segnale,freq_segnale,ampiez-
za)
```

Il suffisso "_f" di signal_source sto ad indicare che la sorgente di segnale produce in uscita dati di tipo float. Per gli altri tipi di dato si procede in modo analogo (_c per complex, ecc.). Gli argomenti sono dati dalla frequenza di campionamento, il tipo di segnale (ad es. gr.GR_SIN_WAVE per la sinusoide), la frequenza del segnale e la sua ampiezza.

Esistono diversi tipi di sorgenti di segnale (GR_CONST_WAVE per un segnale costante, GR_SQR_WAVE per l'onda quadra, ecc.). Più in generale, esistono diversi tipi di sorgenti. Ad esempio, la classe gr.noise_source_c definisce sorgenti di rumore attraverso 3 parametri: funzione di densità di probabilità, ampiezza, seme. Nell'esempio seguente la pdf (probability density function) è data da una gaussiana (GR_GAUSSIAN), di ampiezza 1 e seme 42. Il seme sta ad indicare il seed del generatore di numeri casuali impiegato per implementare la sorgente di rumore. Un diverso seme garantisce la generazione di una diversa sequenza pseudo-random ad ogni invocazione del metodo. Un seme costante garantisce che la sequenza generata sia pseudo-random ma consista sempre nella stessa sequenza di valori per ogni invocazione del metodo.

```
self.gr_noise_source_x_0 = gr.noise_source_c(gr.GR_GAUSSIAN, 1,
42)
```

Nell'esempio la destinazione del flusso di dati è rappresentata da un sink audio in grado di inviare i campioni dei toni sinusoidali (con rate pari a sampling_freq) alla scheda audio.

```
dst = audio.sink(sampling_freq)
```

In flow_graph è definito il metodo connect il quale permette di definire le connessioni tra i blocchi. L'uso del metodo connect avviene per mezzo di tuple (nome blocco, porta) per indicare sorgente e destinazione di ciascun flusso come nell'esempio seguente.

```
self.connect((blocco_sorg, porta),(blocco_dest, porta))
```

Nel caso in cui il blocco possieda una sola porta, è possibile indicare il solo nome del blocco.

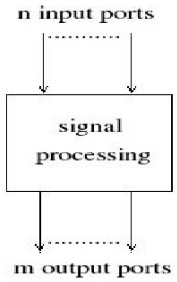


Fig. 3.5.4: Il blocco e le porte

Il blocco è l'elemento base per lo sviluppo in GNUradio. Esiste una intera libreria di blocchi predefiniti (sorgenti, sink, modulatori, ecc.) che permettono di velocizzare lo sviluppo. La creazione o la modifica di un blocco in GNUradio avviene per mezzo del linguaggio C++.

La classe gr_block definisce la base per la definizione di qualsiasi blocco e in essa sono specificati i metodi per la connessione di un blocco all'interno di un flow graph. In gr_block è possibile valorizzare il nome del blocco, il numero e il tipo di porte in ingresso e in uscita dal blocco, il rapporto tra campioni in ingresso e in uscita (eventuale decimazione o interpolazione).

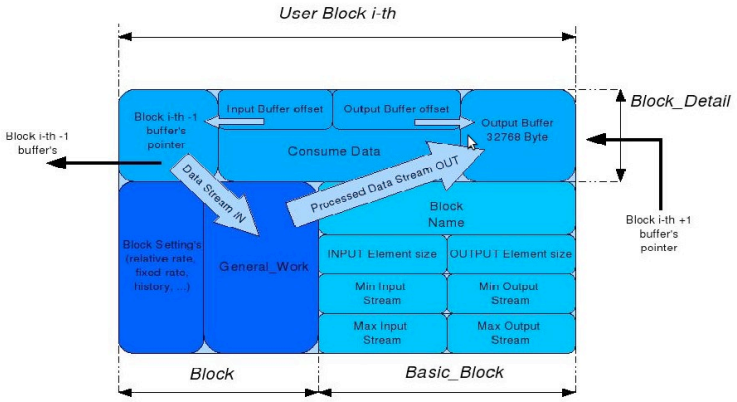


Fig. 3.5.5: Componenti di un blocco GNUradio

La classe gr_block prevede il metodo general work il quale ha lo scopo di realizzare la funzione di elaborazione del blocco (DSP). Il metodo general work viene lanciato dallo Scheduler che esegue in successione tutti i blocchi del flow graph.

```
virtual int general_work(
      int noutput_items,
      gr_vector_int &ninput_items,
      gr_vector_const_void_star &input_items,
      gr_vector_void_star &output_items)
```

Altri campi di un oggetto gr_block sono: il nome del blocco, dimensione dei valori in ingresso e in uscita (vettore o scalare), numero di stream in ingresso e in uscita (porte), l'history ovvero la lunghezza di memoria necessaria in input per produrre un dato output (ad. es. il numero di tap di un FIR).

Il metodo forecast si prefigge lo scopo di fornire una stima del numero di valori in input necessari per fornire un dato numero di valori in output.

```
virtual void gr_block::forecast(int noutput_items, gr_vector_int
& ninput_items_required)
```

Il metodo consume consente di specificare allo Scheduler quanti dati

transitati per una data porta di input sono stati elaborati.

```
void gr_block::consume(int which_input, int how_many_items)
```

Il metodo consume_each invece permette di specificare allo Scheduler quanti dati siano stati elaborati da ogni porta di input.

```
void gr_block::consume_each(int how_many_items)
```

La struttura della cartella di un progetto GNUradio è al seguente:

```
<topdir>/Makefile.am          file Makefile.am generale
.../Makefile.common           Makefile conf file
.../bootstrap                 autoconf, automake, libtool
.../config                    macro m4
.../configure.ac              file di input per autoconf
.../src                       cartella src
.../src/lib                   sorgenti C++
.../src/lib/Makefile.am       sub-Makefile
.../src/python                script Python
.../src/python/Makefile.am    sub-Makefile
.../src/python/run_test       script per test in build tree
```

In GNUradio la struttura contenuta in <topdir>/ è denominata build tree. In tale percorso è possibile sviluppare e testare il codice prima dell'installazione. L'install tree è invece dato dal path:

```
<prefix>/lib/python<version>/site-packages
```

dove <prefix> indica il path passato in fase di lancio del configure di GNUradio (vedi guida all'installazione di GNUradio in appendice B). L'install tree corrisponde all'ambiente di esecuzione dei progetti compilati mentre il build tree la directory provvisoria di sviluppo di un dato progetto nella quale si procede ad un primo debugging. Per realizzare ciò in GNUradio si prevede una prima fase di sviluppo nella quale lo

script che definisce il flow chart venga denominato con il prefisso “qa_” (quality assurance). In tal modo attraverso il comando “make check” lanciato dalla build tree è possibile invocare lo script run_test. Tale script python esegue gli script con prefisso qa_ presenti nella build tree. In tal modo esso permette di verificare la corretta configurazione del progetto al di fuori dell'install tree ovvero prima dell'installazione in GNUradio.

4 Identificazione del segnale Bluetooth

I metodi più comuni per il riconoscimento dei segnali, spesso fanno leva su caratteristiche di strato fisico delle tecnologie impiegate. Tali approcci consentono, ad esempio, di rilevare il tipo di modulazione impiegato, misurando alcuni parametri del segnale ricevuto. Simili tecniche, tuttavia, spesso richiedono la presenza di ricevitori ad-hoc per la specifica tecnologia da analizzare. Considerando che, nel paradigma CR, l'applicazione di interesse è il riconoscimento di una moltitudine di Standard, tale specificità della radio impiegata per lo spectrum sensing, rappresenta un limite.

In questa ricerca, si è scelto di concentrare l'attenzione su caratteristiche degli Standard per tecnologie wireless, che fossero di livello superiore a quello fisico. Appena al di sopra dello strato fisico infatti, a livello MAC, l'unità informativa elementare è data dal pacchetto. Ciascuna tecnologia wireless definisce, in genere nel corrispondente Standard, la struttura dei propri pacchetti. Studiando le specifiche relative ai pacchetti scambiati nei diversi Standard per tecnologie wireless, si è osservato che spesso esistono delle caratteristiche uniche in ogni Standard ovvero, come indicato nella letteratura sul riconoscimento, delle *features* di livello MAC. L'obbiettivo di quest'ultimo capitolo, è quello di illustrare il lavoro svolto per l'estrazione di *features* dalla tecnologia Bluetooth, in grado di garantirne l'identificazione e la classificazione.

Prima di procedere all'identificazione di tali *features* Bluetooth, si è provveduto a studiare il segnale ricevuto dall'USRP2 mediante tecniche di analisi in frequenza. In questo contesto, assume un ruolo fondamentale l'algoritmo *Fast Fourier Transform* (FFT), il quale permette di calcolare in modo efficiente lo spettro di un segnale.

Nelle implementazioni su calcolatore, la trasformazione di Fourier viene svolta su sequenze, in generale aperiodiche e di lunghezza finita, composte dai campioni del segnale. La Trasformata di Fourier Tempo Discreta (*Discrete Time Fourier Transform*, DTFT) è l'operazione che permette l'analisi di sequenze aperiodiche nel dominio della frequenza. Data una sequenza x[n] con $n \in \mathbb{Z}$ la formula di analisi (DTFT) diviene:

$$X(\varpi)=\sum_{n=-\infty}^{\infty} x[n]e^{j\varpi n}$$

$$\varpi=\omega T_s=2\pi f T_s=2\pi\left(\frac{f}{f_s}\right)$$

La pulsazione ω è una grandezza normalizzata con unità di misura data da *rad/sec / campioni/sec = rad/campione*. In tal modo lo spettro viene ad essere definito tra [-π, π] ovvero periodico di periodo 2π. La trasformazione inversa che ricostruisce ciascun elemento della sequenza (formula di sintesi) è data da:

$$x(n)=\frac{1}{2\pi}\int_{-\pi}^{\pi} X(\varpi)e^{j\varpi n}d\varpi$$

Come si vede dalla formula di analisi, la DTFT è un operatore che può essere applicato a serie aperiodiche infinite di campioni. Un calcolatore, tuttavia, può lavorare solo su grandezze discrete e di lunghezza finita. In tal caso, l'approccio comunemente impiegato è quello di rendere periodica la sequenza di lunghezza finita (considerando più repliche di essa in successione) così da poter definire la cosiddetta Trasformata Discreta di Fourier (DFT). Assumendo nota una sequenza finita x[n], tale trasformazione prevede la seguente formula di analisi.

$$X[\omega_k]=\sum_{n=0}^{N-1} x[t_n]e^{j\omega_k t_n} \qquad k=0,1,2,\ldots,N-1$$

La $X[\omega_k]$ questa volta è essa stessa una sequenza e rappresenta lo spettro di x[n] alle frequenze ω_k (dominio discreto), t_n rappresenta l'istante di campionamento pari a nTs. La variabile ω_k può essere scomposta in:

$$\omega_k=k\Omega=k\frac{2\pi}{NT_s}=k\frac{2\pi}{N}f_s$$

Attraverso l'impiego della DTFT si può pensare di calcolare la densità spettrale di potenza mediante il metodo del periodogramma.

Il periodogramma è una funzione in grado di offrire una stima della PSD data una sequenza aperiodica di lunghezza infinita x[n]. Considerando una finestratura (generalmente rettangolare) tale che $x_w[n] = w[n]x[n]$ con w[n] contente M campioni non nulli, il periodogramma risulta definito come segue.

$$P_{x,M}(\omega)=\frac{1}{M}\left|DTFT\,(x_w)\right|^2=\frac{1}{M}\left|\sum_{n=0}^{M-1} x_w(n)e^{-j\omega n}\right|^2$$

Al tendere di M ad infinito, ovvero all'aumentare della dimensione della finestratura, il Periodogramma offre una stima sempre migliore della PSD $S_x(\omega)$ del segnale x[n].

$$E[\lim_{M\to\infty} P_{x,M}(\omega)]=S_x(\omega)$$

con $S_x(\omega)$ densità spettrale di frequenza (PSD) della sequenza x[n].

Nel caso di sequenze di campioni di lunghezza finita non è possibile, come già detto, calcolare la DTFT, così la suddetta procedura si modifica nella seguente. Data la sequenza x[n] di N campioni, si definisce funzione di autocorrelazione $R_{xx}(l)$ la seguente espressione:

$$R_{xx}(l)=\frac{1}{N}(x*x)(l)=\sum_{n=0}^{N-1}\overline{x(n)}\,x(n+l)$$

Una stima della funzione di densità spettrale di potenza $\hat{S}_x(\omega_k)$ (*Power Spectral Density*, PSD) basata su N campioni complessi, è ottenuta attraverso la DFT della funzione di autocorrelazione R_{xx} appena definita.

$$\widehat{S_x}(\omega_k) = DFT_k(R_{xx}) = \frac{\left|X[\omega_k]\right|^2}{N} = \frac{1}{N}\left|\sum_{n=0}^{N-1} x[n]e^{-j\omega_k n}\right|^2$$

Al crescere del numero di campioni N, si ottiene una stima sempre migliore della PSD del segnale campionato.

La finestratura di una sequenza viene effettuata generalmente per smussare (*smoothing)* le discontinuità presenti a inizio e fine sequenza che altrimenti produrrebbero (ad es. dopo il calcolo dell'FFT) componenti spurie ad alta frequenza. Tali discontinuità vengono generate nel momento in cui, per rendere fattibile la DFT, si rende periodica la sequenza introducendo così dei salti ad inizio e fine del periodo considerato. Moltiplicando la sequenza x[n] per una finestra w[n] opportuna (vedi tabella seguente) si riesce ad attenuare l'effetto delle discontinuità ai bordi della sequenza. Esistono a tale scopo, diversi tipi di finestratura: rettangolare, triangolare (Bartlett), Hamming e altre.

Window	Definition
Rectangular	w[k] = 1 0<=k<=N-1
Triangular (Bartlett)	$w[k]=1-\left\|1-\frac{2k}{N-1}\right\|$ 0<=k<=N-1
Hamming	$w[k]=0.54-0.46\ \cos(\frac{2k\pi}{N-1})$ 0<=k<=N-1
Hann	$w[k]=\frac{1}{2}(1-\cos(\frac{2k\pi}{N-1}))$ 0<=k<=N-1

Un algoritmo molto efficiente che implementa il calcolo della DFT è, come già accennato, noto sotto il nome di Fast Fourier Transform (FFT). Mediante questo algoritmo, si è riusciti a sviluppare rapidamente tutta una serie di applicazioni di analisi spettrale in real time e che sono alla base dei moderni algoritmi di Digital Signal Processing (DSP).

Supponendo una velocità di campionamento f_s, il teorema di Nyquist, impone che la banda rappresentabile B sia minore o uguale a $f_s/2$ (caso complesso: $B = f_s$). Un parametro fondamentale della FFT è dato dal nu-

mero di punti (N) con i quali si intende rappresentare lo spettro del segnale. Dal valore N, si ricava il numero di bin che rappresentano lo spettro di un segnale reale, ovvero N/2 (per un segnale analitico ovvero non simmetrico il numero di bin coincide invece con N). A crescere del numero dei bin, si ottiene una risoluzione sempre migliore. Si ha quindi:

$$risoluzione = \frac{sampleRate}{N} = \frac{B}{bins}$$

La risoluzione massima è limitata dal numero di punti dello schermo che visualizza lo spettro stesso. Dal numero di bin deriva, inoltre, il tempo di riempimento (*fill time*) del buffer FFT durante il quale viene calcolato un plot completo dello spettro, che è dato da:

$$fillTime = \frac{1}{risoluzione}$$

Questo tempo, assieme a quello di elaborazione per il calcolo della FFT, determina il ritardo di visualizzazione dello spettro. Al crescere del numero dei punti aumenta il *fill time*.

Il suddetto approccio, permette quindi uno studio mirato ad evidenziare l'occupazione di banda di un certo segnale, con l'ipotesi che questo sia sufficientemente stazionario durante il calcolo dello spettro. Per segnali come il Bluetooth, infatti, ci si trova a dover analizzare una occupazione spettrale caratterizzata da rapide variazioni (FHSS a 1600 hop/s). Inoltre, attraverso l'analisi dello spettro basata sulla FFT, ci si trova nell'impossibilità di identificare segnali diversi che occupano la stessa banda in un intervallo di tempo minore del *fill time*. Inoltre nel dominio della frequenza può accadere che più segnali da analizzare risultino parzialmente o completamente sovrapposti (ad es. segnali UWB e WiFi). In questo scenario l'analisi appena citata non permette di separare i diversi segnali presenti.

Attraverso un modulo nativo di GRC (FFT sink) in grado di rappresentare in real time lo spettro del segnale ricevuto dall'USRP2, si è risuciti a verificare in tempo reale l'effettiva posizione nello spettro dei canali Bluetooth (Fig. 4.1). In questo modo è stato possibile prevedere quanti di questi sarebbero stati intercettati dall'*energy detector* sviluppato in MATLAB.

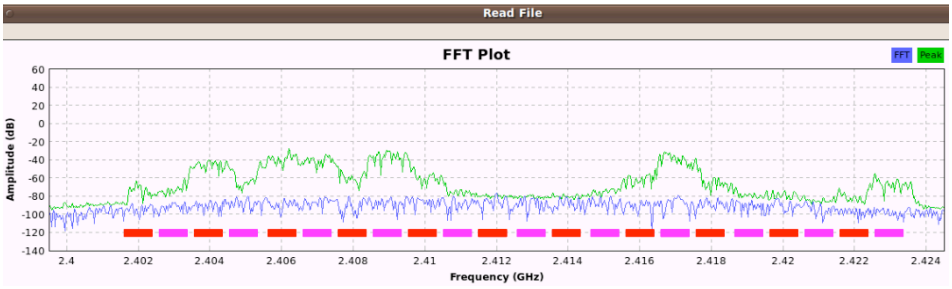


Fig. 4.1: Posizione di 22 canali Bluetooth catturati in 25 MHz)

Nel paragrafo successivo si vedrà in che modo l'*energy detector* possa essere impiegato con successo per l'identificazione delle comunicazioni Bluetooth garantendo semplicità realizzativa ed efficienza computazionale.

4.1 Energy detector

A differenza di altri sistemi di spectrum sensing, l'Energy Detector (ED) è un metodo di analisi nel dominio del tempo che permette di ottenere stime sull'utilizzazione dello spettro in tempi molto rapidi, essendo basato su semplici operazioni svolte nel dominio del tempo.

L'ED è un ben noto risultato della Teoria della decisione al problema della detezione di segnali aleatori immersi in rumore AWGN. Questo approccio, basato sulla verifica di ipotesi (*hypothesis testing*), consente di risolvere problemi di decisione binaria (presenza o assenza di segnale utile) a partire dall'osservazione di una sequenza di campioni ricevuti (come è nel caso di forme d'onda campionate dall'USRP).

Si assume, quindi, che sia disponibile un insieme di campioni di segnale statisticamente indipendenti (x[0], x[1], ..., x[N-1]). Su questi dati si applica una opportuna funzione T(**x**), detta statistica sufficiente, in grado di evidenziare un parametro discriminante per la decisione stessa. La determinazione della funzione T(**x**) e del criterio di decisione, ovvero di una soglia di decisione, sono i nodi centrali di questo approccio. Per la rivelazione di segnale utile immerso in rumore bianco gaussiano (*Additive White Gaussian Noise*, AWGN), si defiscono le seguenti ipotesi:

$H_1 : y[n] = x[n] + w[n]$ *Segnale presente*
$H_0 : y[n] = w[n]$ *Segnale assente*
$n = 1, \ldots, N$ *con N finestra temporale di misurazione*

I campioni di rumore w[n] sono considerabili realizzazioni di un processo aleatorio gaussiano statisticamente indipendenti ed identicamente distribuite (i.i.d.) di media nulla e varianza σ_w^2. La rivelazione del segnale utile è di tipo non coerente (il segnale non viene demodulato) per cui, anche i campioni di segnale x[n], si possono assumere come realizzazioni di un processo aleatorio gaussiano i.i.d. a media nulla e varianza σ_x^2. In questo modo le suddette ipotesi possono venir riscritte come segue.

$H_1 : \quad \mathcal{N}(0, \sigma_x^2 + \sigma_n^2)$
$H_0 : \quad \mathcal{N}(0, \sigma_n^2)$

La rilevazione di segnale mediante l'approccio di Neyman-Pearson (NP) consiste nel confrontare il rapporto di verosimiglianza L(x) con la soglia γ:

$$L(\boldsymbol{x}) = \frac{p(\boldsymbol{x}; H_1)}{p(\boldsymbol{x}; H_0)} > \gamma$$

dove le verosimiglianze $p(\boldsymbol{x} ; H_1)$ e $p(\boldsymbol{x} ; H_0)$ sono rispettivamente le funzioni di densità di probabilità (d.d.p.) associate al vettore di campioni osservato $\boldsymbol{x}$ quando è vera l'ipotesi H_1 oppure H_0. Il calcolo della soglia γ mediante NP passa attraverso la valutazione del seguente integrale, avendo deciso a-priori una data probabilità di falso allarme (P_{FA}):

$$P_{FA} = \int_{\boldsymbol{x}: L(\boldsymbol{x}) > \gamma} p(\boldsymbol{x}; H_0) d\,\boldsymbol{x} = \alpha$$

Date le ipotesi appena citate, $L(\boldsymbol{x})$ diviene:

$$L(\boldsymbol{x})=\frac{\frac{1}{\left[2\pi(\sigma_x^2+\sigma_n^2)\right]^{\frac{N}{2}}}e^{-\frac{1}{2(\sigma_x^2+\sigma_n^2)}\sum_{n=0}^{N-1}x^2[n]}}{\frac{1}{(2\pi\sigma_n^2)^{\frac{N}{2}}}e^{-\frac{1}{2\sigma_n^2}\sum_{n=0}^{N-1}x^2[n]}}$$

Prendendo poi i logaritmi di numeratore e denominatore e raccogliendo i termini indipendenti da n nella soglia γ si ottiene la seguente semplice espressione per la funzione di test T(**x**):

$$T(\boldsymbol{x})=\sum_{n=0}^{N-1}x^2[n]>\gamma'$$

dove in γ' si sono inclusi i termini non dipendenti da n. L'ED appena ricavato calcola l'energia dei campioni di segnale che cade in una finestra temporale di durata N. Se il segnale è presente (H_1), ci si aspetta un valore per T(**x**) maggiore della soglia. Quando, invece, non vi è segnale (H_0) allora T(**x**) resta al di sotto della soglia. Se sulla funzione di test T(**x**) viene operata la seguente normalizzazione:

$$\frac{1}{N}T(\boldsymbol{x})$$

si ottiene una stima della varianza del segnale. Quest'ultima cresce nel passare dall'ipotesi H_0 all'ipotesi H_1. Attraverso l'applicazione della soglia quindi diviene possibile rilevare questi due eventi.

Le prestazioni dell'approccio NP per l'*energy detector* sono ricavabili a partire dalle probabilità di corretta decisione P_D e di falso allarme P_{FA}. Innanzitutto si fa notare che:

$$\frac{T(\boldsymbol{x})}{\sigma_x^2+\sigma_n^2}\sim\chi_N^2 \qquad \text{se è vera } H_1$$

$$\frac{T(\boldsymbol{x})}{\sigma_n^2}\sim\chi_N^2 \qquad \text{se è vera } H_0$$

In entrambe le ipotesi si ha la somma dei quadrati di N variabili aleatorie gaussiane statisticamente indipendenti e identicamente distribuite (i.i.d.) che assume distribuzione di tipo χ^2 (chi-quadro):

$$X_\nu(x)=\frac{1}{2^{\frac{\nu}{2}}\Gamma(\frac{\nu}{2})}x^{\frac{\nu}{2}-1}e^{-\frac{x}{2}} \qquad per \quad x>0$$

con k ϵ N\{0} gradi di libertà e supporto x ϵ [0,∞). Al fine di ricavare la P_{FA} e la P_D si ricorda che l'integrale sulle code della funzione Q per una distribuzione di tipo χ^2 risulta:

$$Q_{X_N^2}(x)= \begin{cases} 2Q(\sqrt{x}) & N=1 \\ 2Q(\sqrt{x})+\frac{e^{-\frac{1}{2}x}}{\sqrt{\pi}}\sum_{k=1}^{\frac{N-1}{2}}\frac{(k-1)!(2x)^{k-\frac{1}{2}}}{(2k-1)!} & N>1 \; e \; dispari \\ e^{-\frac{1}{2}x}\sum_{k=0}^{\frac{N}{2}-1}\frac{(\frac{x}{2})^k}{k!} & N \; pari \end{cases}$$

La probabilità di falso allarme P_{FA} e la probabilità di corretta decisione P_D divengono quindi:

$$P_{FA} = P_r\{T(x)>\gamma';H_0\} = Q_{X_N^2}\left(\frac{\gamma'}{\sigma^2}\right)$$

$$P_D = P_r\{T(x)>\gamma';H_1\} = Q_{X_N^2}\left(\frac{\gamma'}{\sigma_s^2+\sigma^2}\right)$$

Dividendo l'argomento della funzione Q nell'espressione della PD per la varianza del rumore σ^2 e definendo $\gamma'' = \gamma'/\sigma^2$ si ottiene:

$$P_D = P_r\{T(x) > \gamma'; H_1\} = Q_{\chi^2_N}\left(\frac{\gamma''}{\frac{\sigma_s^2}{\sigma^2}+1}\right)$$

In quest'ultima espressione si vede che al crescere del rapporto segnale a rumore (SNR = σ_s^2 / σ_2) l'argomento della funzione Q decresce ovvero la P_D aumenta essendo la funzione Q monotona decrescente. Per N elevato, la somma dei quadrati di N variabili aleatorie gaussiane i.i.d., per il teorema del limite centrale viene ad assumere una distribuzione gaussiana.

$$T(\boldsymbol{x}) \sim N(N\sigma_w^2, 2N\sigma_w^4) \qquad \text{se è vera } H_1$$

$$T(\boldsymbol{x}) \sim N(N(\sigma_w^2+\sigma_x^2), 2N(\sigma_w^2+\sigma_x^2)^2) \qquad \text{se è vera } H_0$$

In questo caso la P_{FA} e la P_D assumono la forma seguente:

$$P_{FA} = Q\left(\frac{\gamma - N\sigma_w^2}{\sqrt{2N\sigma^4}}\right)$$

$$P_D = Q\left(\frac{\gamma - N(\sigma_w^2+\sigma_x^2)}{\sqrt{2N(\sigma_w^2+\sigma_x^2)^2}}\right)$$

Per un dato valore della soglia, fissati N e le varianze di rumore e di segnale, le precedenti forniscono la cosiddetta ROC (Receiver Operating Curve) che caratterizza le prestazioni di un dato ricevitore.

Nella precedente trattazione, come già detto, la funzione di test T(**x**) viene confrontata con una soglia in grado di offrire, fissato un certo SNR, determinate prestazioni (P_{FA}, P_D). In questo lavoro si è scelto di fissare la soglia γ ad un valore pari a 10 dB sopra il livello del rumore osservato. Tale semplificazione è giustificata dal fatto che non si è interessati al calcolo della P_{FA} e della P_D ma, piuttosto, a caratterizzare la distribuzione statistica delle durate e dei tempi di interarrivo dei pacchetti (*packet exchange patterns*).

Il valore di 10 dB rappresenta, nel caso in esame, un buon compromesso

per poter catturare praticamente tutti i pacchetti scambiati tra i dispositivi Bluetooth posti in prossimità dell'SDR. Nello stesso tempo si è cercato di introdurre un margine (10 dB appunto) in grado di garantire una certa robustezza nei confronti del rumore e di possibili interferenti nel raggio di copertura dell'SDR.

La struttura di un ED è molto semplice e nella figura 4.1.1 è illustrato lo schema a blocchi dell'ED, che opera sui campioni ottenuti mediante l'USRP2.

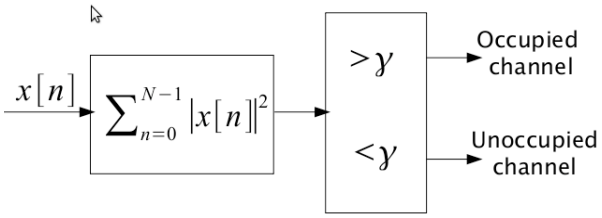


Fig. 4.1.1: Schema di un energy detector

La sequenza x[n], posta in input nello schema di Fig. 4.1.1, è da ritenersi filtrata attraverso l'USRP2 mediante decimazione. In questo modo è molto semplice campionare il canale Bluetooth n-esimo mediante sintonizzazione dell'USRP2 al centro del canale scelto e impostazione del giusto valore di decimazione (ad es. un valore pari a 4 per ottenere 25 *MS/s* complessi ovvero 25 *MHz* di banda).

Utilizzando la formula dell'*energy detector* appena vista, si è provveduto a definire la funzione *short-term energy* (E_N).

$$E_N(\boldsymbol{r}) = \sum_{i=1}^{N} |r_i|^2 \cdot T_S$$

Nella precedente N è la lunghezza della finestra temporale per il calcolo dell'energia, r_i rappresenta il campione i-esimo di tensione all'interno della finestra considerata e T_S il periodo di campionamento.

La funzione *short-term energy* è stata quindi implementata in MATLAB consentendo di ricavare l'evoluzione nel tempo dell'energia (a breve termine). Tale diagramma è utile alla rilevazione dei pacchetti intercettati dall'USRP2. Applicando infatti la soglia appena menzionata, si è potuto

graficare un diagramma dei pacchetti catturati (*packet diagram*) dal quale è stato possibile estrarre le informazioni sui pacchetti relative all'istante di inzio (*timestamp*) e alla durata del pacchetto (*duration*). Queste tuple (timestamp, duration) sono state raccolte nel vettore MATLAB "*packets*" ed hanno rappresentato il punto di partenza per la verifica della validità delle *features* proposte per le trasmissioni basate su link ACL (dati) e SCO (voce).

Nel par. 4.2 verranno quindi descritte le features proposte per il caso di trasmissioni Bluetooth basate su link di tipo ACL (dati) e SCO (voce). Si evidenzierà poi, la loro validità illustrando i risultati ottenuti dall'osservazione di traffico reale dati e voce. Nel par. 4.3 si evidenzieranno i risutati ottenuti per le features proposte nel caso di trasmissione basata su link ACL (dati). Allo stesso modo nel par. 4.4 si commenteranno i risutati relativi al caso di trasmissioni basate su link SCO (voce). Nel par. 4.5 si descriverà un parametro basato sul tempo di riconoscimento del Bluetooth ricavabile dall'impiego delle *features* proposte e si trarranno alcune conclusioni.

4.2 Estrazione delle features

L'obbiettivo di questo lavoro è stato quello di ottenere alcune features che fossero caratteristiche della tecnologia Bluetooth. Si è cercato di identificare delle peculiarità dei protocolli di livello MAC, che fossero estraibili attraverso l'impiego di un semplice *energy detector*.

Come descritto, ciò ha richiesto uno studio attento sia della tecnologia Bluetooth (Cap. 2), che dell'hardware impiegato, la radio SDR USRP2 (Cap. 3). I precedenti studi condotti su questo approccio [1] e le conoscenze acquisite in questo percorso, hanno permesso di identificare e proporre alcune *features* di livello MAC per il Bluetooth. Come anticipato in [1], queste caratteristiche relative ai pacchetti Bluetooth, dovranno poi essere integrate nel modulo AIR-AWARE di spectrum sensing *multi-standard* basato sull'*energy detector*.

Dalle specifiche presentate nel Cap. 2 e descritte dettagliatamente in [1, 2, Cap. 2], si evince che una peculiarità della tecnologia Bluetooth è ricavabile dalla durata di alcuni tipi di pacchetti di controllo, i quali assumono una lunghezza costante (ID, FHS, NULL, POLL). L'idea che ne scaturisce è che, estraendo attraverso un *energy detector* la durata dei pacchetti

catturati dall'USRP2, si possa identificare l'attività di una o più radio Bluetooth.

Sia nel caso di pacchetti per il trasporto di dati che per il flusso voce, non è generalmente possibile considerare una lunghezza fissa ovvero, questo metodo dovrebbe risultare inapplicabile. Tuttavia, è ragionevole immaginare che durante una trasmissione dati o voce, una volta scelto il tipo di pacchetti (1, 3 o 5 slot) da impiegare, il protocollo utilizzato incapsuli in modo efficiente il flusso in uscita utilizzando, ove possibile, l'intero payload disponibile. Ciò significa che nel mezzo di una trasmissione Bluetooth, i pacchetti da 1 o più slot avranno dimensione vicina a quella massima ammessa dallo Standard per quel dato formato.

In questo lavoro, si è scelto di proporre la durata dei pacchetti come *feature* caratterizzante delle comunicazioni Bluetooth. L'analisi della distribuzione delle durate dei pacchetti (1, 3 o 5 slot) ha permesso di studiare, in condizioni di traffico reale, quale sia la percentuale di pacchetti scambiati in base agli slot occupati.

Un'altra caratteristica rilevante della tecnologia Bluetooth è quella di organizzare le trasmissioni mediante uno schema TDD/TDMA ovvero attraverso l'impiego di *timeslot* di durata 625 μs. Ciò significa che qualunque evento di ricetrasmissione in Bluetooth avviene con cadenza minima pari alla durata di 1 slot.

Per ricavare tale valore è sufficiente calcolare il tempo di interarrivo dei pacchetti presenti nel *packet diagram*. Sebbene, in generale, una comunicazione Bluetooth presenti pacchetti di dimensione da 1, 3 o 5 slot, nel lungo termine, la più frequente dovrebbe risultare quella minima di uno slot. Inoltre, nel caso di trasmissioni voce, la particolare allocazione periodica dei pacchetti H1, HV2 e HV3 dovrebbe introdurre ulteriori valori del tempo di interarrivo molto frequenti. Ad esempio nel caso di trasmissione voce che impieghi pacchetti HV3, dallo Standard si ha una cadenza di invio T_{SCO} pari a 3750 μs, ovvero una situazione simile a quella riportata in Fig. 4.2.1.

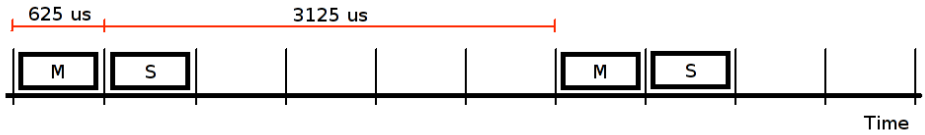


Fig. 4.2.1: Trasmissione voce con pacchetti HV3 (T_{SCO}=)

Come si può notare, nel caso di una sola trasmissione voce basata su pacchetti HV3, i tempi di interarrivo più frequenti dovrebbero essere pari al T_{SLOT} e a T_{SCO} - T_{SLOT} ovvero a 3125 μs (vedi Fig. 4.2.1). Anche per il caso di pacchetti HV2 e HV1 possono essere fatte le medesime considerazioni a partire dai valori dello Standard del parametro T_{SCO} (2500 μs per l'HV2 e 1250 μs per l'HV1). Da queste osservazioni si è scelto di proporre una seconda *feature* relativa altempo di interarrivo dei pacchetti.

Al fine di verificare la validità delle features proposte si è provveduto a definire un digramma dei pacchetti scambiati (*packet diagram*), utile per ricavare *timestamp* e durata dei pacchetti Bluetooth intercettati con l'USRP2. Dal vettore contenente le tuple (*timestamp*, durata) di ogni pacchetto ricevuto è stato possibile calcolare: la distribuzione delle durate, il tempo di interarrivo e il tempo minimo di riconoscimento del Bluetooth in condizioni di traffico reale. Attraverso queste analisi è stato possibile verificare la validità delle *features* proposte.

Verrà ora illustrata la distribuzione statistica delle due *features* proposte in condizioni di traffico reale sia per il caso di trasferimento di un file (link ACL) che di trasmissione di flusso voce (link SCO).

4.3 Durata e tempo di interarrivo dei pacchetti (link ACL)

Lo scenario studiato per la trasmissione dati (ACL) è stato quello del trasferimento di un file da un host con adattatore Bluetooth Belkin F8T012 (Class 1, 20dBm, EDR 2.0, MAC: 00:0A:3A:6D:EA:18) verso un altro host con adattatore Trust Ultra Small Bluetooth USB (Class 2, 4dBm, EDR 2.0, MAC: 00:02:72:19:8D:C0). La distanza tra i dispositivi Bluetooth è stata scelta attorno ai 50 cm (circa 4 lunghezze d'onda a 2.4 *GHz*). L'USRP2 è stato posto a circa 1 m di distanza da entrambi gli adattatori Bluetooth. In queste condizioni si è potuto ottenere un SNR elevato per il rilevamento dei pacchetti scambiati dai 2 dispositivi.

Sull'USRP2 è stata montata una *daughterboard* XCVR2450 operante in banda ISM 2.4 e 5.8 *GHz*. L'antenna montata sull'USRP2 è stata una lineare (2.4 *GHz*) con guadagno pari a 3dBi. Le impostazioni di GNUradio sono state fissate come illustrato in Fig. 4.3.1.

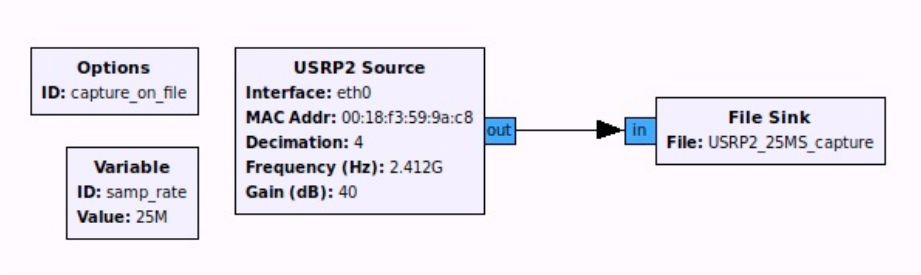


Fig. 4.3.1: Flow graph in GRC del setup per la cattura su file

La scelta di impostare la frequenza centrale a 2.412 *GHz* è dovuta al fatto di ottenere, nella banda [-12.5, 12.5] *MHz*, il maggior numero di canali Bluetooth. Partendo dal canale 1 posto a 2.402 *GHz*, con l'USRP2 si sono potuti intercettare un totale di 22 canali. Si è infatti considerata una banda di guardia, tra la banda corrispondente al primo canale e il limite inferiore della banda di ricezione, di circa 2.5 *MHz*. In questo modo si è evitato di subire l'attenuazione agli estremi della banda dell'USRP2 introdotta dalle non idealità dei suoi filtri. In 25 *MHz* è stato quindi possibile monitorare 22 canali Bluetooth (Fig. 4.3.2).

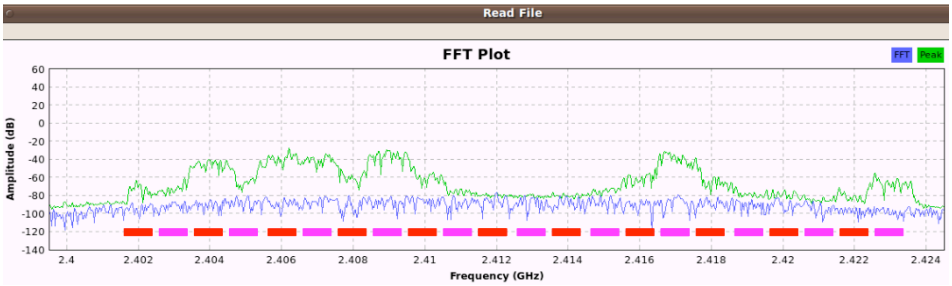


Fig. 4.3.2: Spettro del segnale catturato dall'USRP2 (22 canali in 25 MHz)

Una volta definito lo scenario e la banda di ricezione si è provveduto ad ottenere i campioni dall'USRP2 impiegando il modulo "*fileSink*" di GRC. Si è quindi provveduto a inviare un file (~6 *MB*) tra i due *host* connessi attraverso adattatori USB Bluetooth. Il salvataggio su file dei campioni ricevuti è stato avviato a trasmissione iniziata ed è durato alcuni secondi.

L'analisi dei campioni è stata condotta attraverso una serie di script MATLAB in grado di implementare un semplice *energy detector*. Attraverso questo è stato quindi possibile graficare l'andamento dell'energia a breve termine ricevuta dall'USRP2. Dall'osservazione dello *short-term energy diagram* (Fig. 4.3.3) ottenuto dalle catture relative al setup sperimentale si

è scelto di fissare la soglia di decisione dell'ED a +10 dB dal *Noise Floor*.

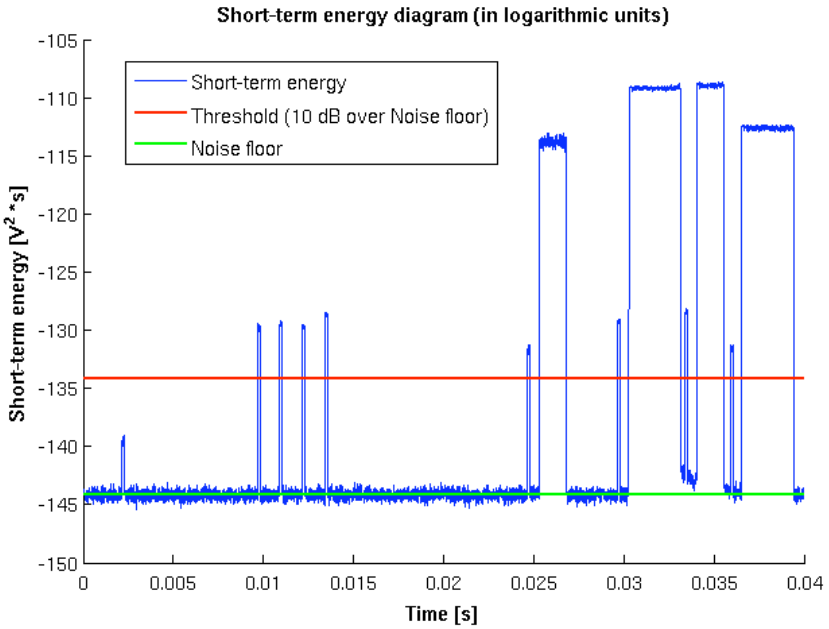


Fig. 4.3.3: Short-term energy (N=250, overlap 50%, BW=25 MHz)

In questo modo è stato possibile catturare praticamente tutti i pacchetti scambiati tra i due dispositivi escludendo, per quanto possibile, altre comunicazioni tra dispositivi Bluetooth o Wifi. La stima del livello di rumore (Noise Floor) nel caso di energy detection è un problema molto complesso, tutt'ora oggetto di ricerca [10]. La scelta di fissare la soglia di decisione dell'ED a +10 dB dal Noise Floor osservato, è scaturita dalla necessità di semplificare la trattazione e di ottenere un packet diagram ugualmente valido in grado di evidenziare la presenza delle *features* proposte. Il risultato dell'applicazione della soglia allo *short-term energy* diagram è l'andamento nel tempo della presenza o assenza di pacchetto (il *packet diagram*). Attraverso questo grafico (Fig. 4.3.4) si sono potuti estrarre *timestamp* e durata dei pacchetti intercettati.

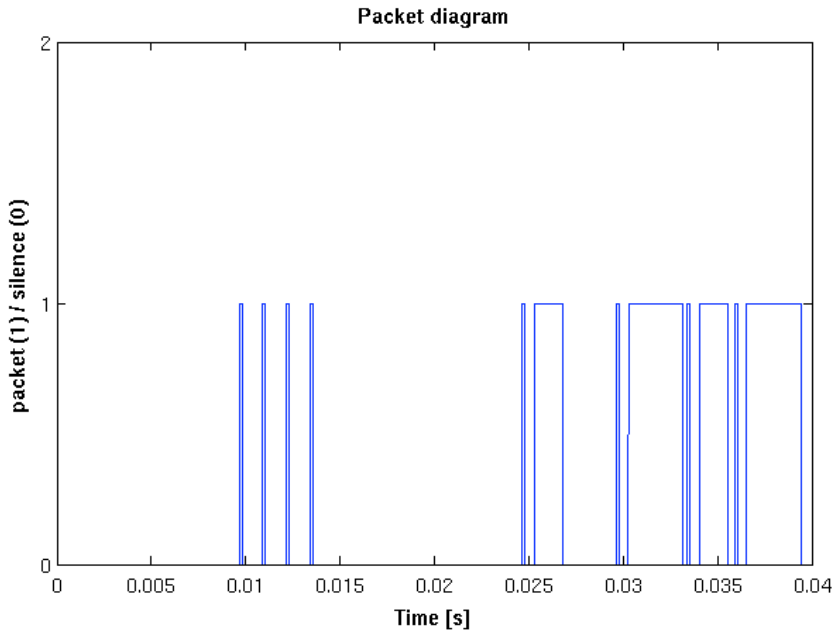


Fig. 4.3.4: Packet diagram relativo allo short-term energy diagram precedente

Come anticipato, si è scelto di focalizzare l'attenzione sulla lunghezza dei pacchetti e sul tempo di interarrivo. Ciò ha permesso di evidenziare che la lunghezza dei pacchetti dati scambiati durante, ad esempio, il trasferimento di un file, appare sufficientemente concentrata attorno a 3 valori che corrispondono alle dimensioni massime dei pacchetti da 1, 3 e 5 slot previste dallo Standard. Nel caso di traffico dati Bluetooth i pacchetti scambiati, come descritto nel Cap. 2, possono occupare 1, 3 oppure 5 slot. In particolare le figure seguenti riassumono tipo e durata di tutti i possibili pacchetti dati.

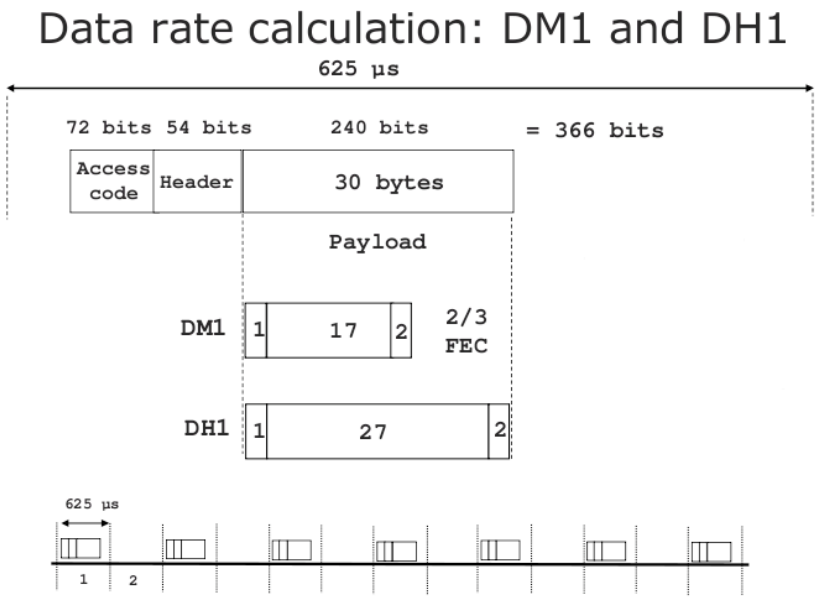


Fig. 4.3.5: Pacchetti dati da 1 slot

Nel caso di pacchetti DM1 (Fig. 4.3.5) è presente una codifica FEC 2/3 realizzata mediante un codice di Hamming (15,10) in grado di correggere un errore per ogni codeword. Il pacchetto DM1, oltre all'Access Code e l'Header, prevede 17 bytes di dati, 1 byte di *data header*, 2 byte di CRC. Nel caso di pacchetti DH1 si prevede l'assenza di FEC e quindi il payload raggiunge i 27 byte. Per entrambe queste tipologie (DM1 e DH1) la durata massima è fissata a 366 μs. Nel caso di pacchetti da 1 slot ricadono anche i pacchetti: ID (68 μs), NULL/POLL (126 μs) e l'FHS (366 μs). L'unica periodicità in grado di caratterizzare il tempo di interarrivo per i pacchetti da 1 slot su link ACL è il tempo T_{SLOT}.

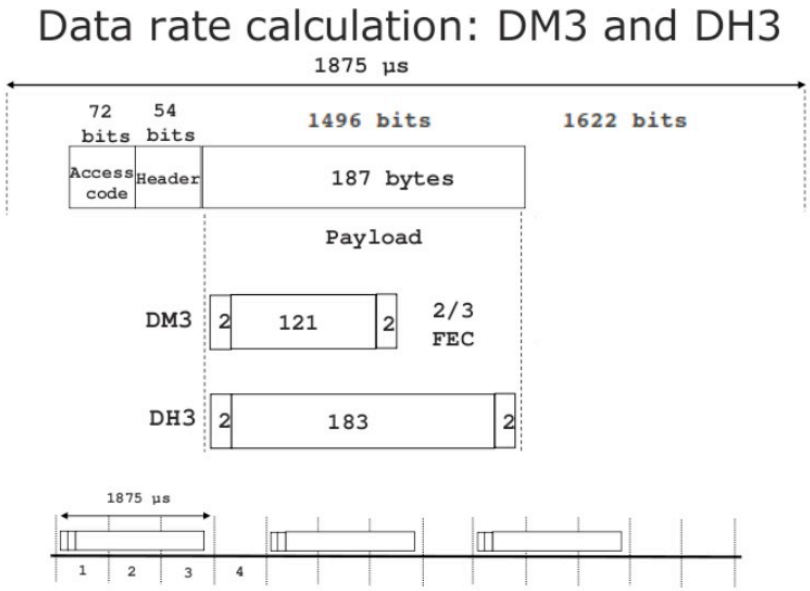


Fig. 4.3.6: Pacchetti dati da 3 slot

Analogamente per il caso di pacchetti DM3, DH3, DM5 e DH5 si adottano schemi di Forward Error Correction solo per i pacchetti DM. Il caso di invio di una serie di pacchetti multislot (da 3 e 5 slot) si prevede una spaziatura minima tra questi pari a 1 slot.

Per i pacchetti DM3 e DH3 le durate massime risultano fissate a 1622 μs. Il tempo di interarrivo tipico di queste trasmissioni (vedi Fig. 4.3.6) dovrebbe essere, quello dato da 4 T_{SLOT} ovvero 1875 μs + 625 μs = 2500 μs. Tuttavia, anche in presenza di pacchetti da 3 slot, il tempo di interarrivo pari a T_{SLOT} è presente e risulta dalla presenza di pacchetti di ACK (NULL). Dal punto di vista dell'ED, nel caso di pacchetti da 3 slot seguiti sempre da un ACK, ciò comporta un tempo di interarrivo tipico di 1875 μs. L'obbiettivo di questi studi è stato proprio quello di evidenziare lo scenario più comune.

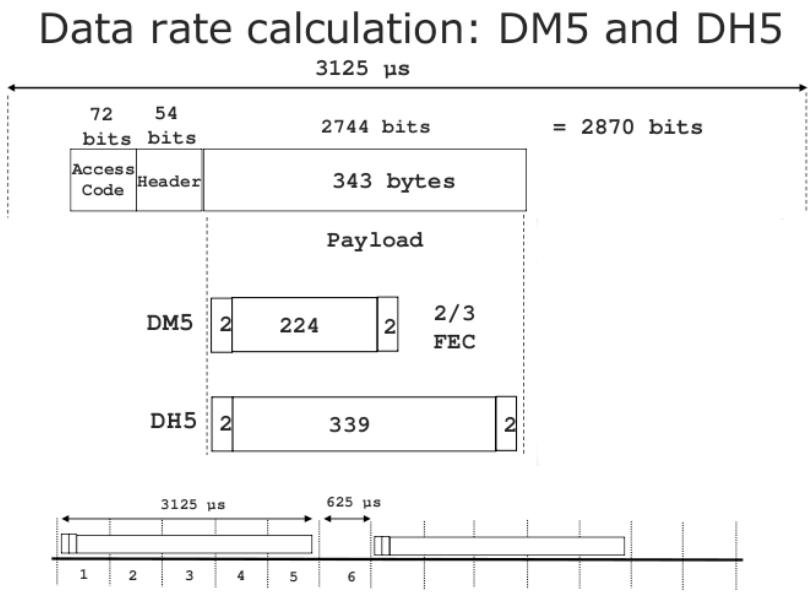


Fig. 4.3.7: Pacchetti dati da 5 slot

Per i pacchetti da 5 slot (Fig. 4.3.7) la situazione è del tutto simile a quella già vista per i pacchetti da 3 slot. In questo caso la durata massima del pacchetto "in aria" risulta di 2870 *µs*. Anche qui, in caso di trasmissione di soli pacchetti DM5 e DH5, il tempo di interarrivo tipico dovrebbe essere pari a 3125 *µs* + 625 *µs* = 3750 *µs*. In presenza di ACK esso si ridurrebbe a 3125 *µs*.

Secondo quanto descritto, considerando un impiego efficiente dei vari formati di pacchetto da 1, 3 e 5 slot, dagli strati deputati alla segmentazione e riassemblaggio del flusso dati in pacchetti, è ragionevole assumere la presenza di alcune regolarità nella distribuzione delle durate dei pacchetti. Misurando, quindi, la durata di tutti i pacchetti presenti nel *packet diagram* su un tempo di sensing di 3 secondi ed una banda di 25 *MHz*, si è evidenziato che nel caso di trasmissioni ACL (invio di un file tra dispositivi) risultano ben evidenti i 3 tipi di pacchetto: pacchetti da 1 slot (~144 *µs*), pacchetti da 3 slot (~1540 *µs*) e pacchetti da 5 slot (~2890 *µs*). L'istogramma che illustra tale situazione è riportato in Fig. 4.3.8.

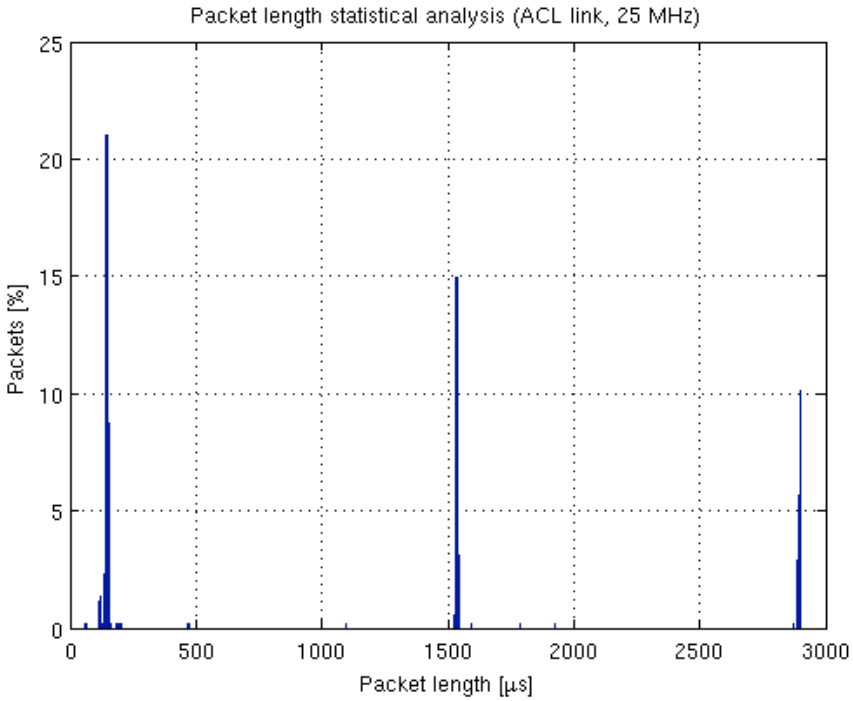


Fig. 4.3.8: Distribuzione statistica delle durate dei pacchetti dati (ACL)

Attraverso queste misure è scaturito che per i pacchetti da 1 slot più frequenti, l'USRP2 mostra una durata attorno ai 144 *µs* (21% del totale dei pacchetti rilevati). Dalla posizione occupata nel pattern di invio/ricezione tra Master e Slave (osservando lo *short-term energy diagram*) da questi pacchetti, si può inferire che si tratti di pacchetti NULL con funzione di *acknoledgment* inviati dallo Slave.

I pacchetti da 3 slot invece appaiono concentrati attorno al valore 1540 *µs* (15% del totale dei pacchetti rilevati). In questo caso lo Standard prevede una durata massima di 1622 *µs*. Infine, per i pacchetti da 5 slot si registra un valore medio pari a 2890 *µs* (10% del totale dei pacchetti rilevati) dove nello Standard risulta 2870 *µs*. Osservando la Fig. 4.3.8, risulta chiaro che la segmetazione del flusso dati in pacchetti, nel caso di trasmissioni dati (ACL) Bluetooth, si traduce nella presenza di un gran numero di pacchetti a payload pieno (il 46% dei pacchetti rilevati). Il restante 54% dei pacchetti rilevati assume una durata aleatoria tra i valori ammessi dallo Standard. Sfruttando questa informazione si può pensare di classificare il traffico Bluetooth attraverso l'osservazione della durata dei pacchetti rilevati. Ciò risulta facilitato dalla presenza di quasi la metà dei pacchetti dati rilevati di lunghezza praticamente costante (picchi in Fig. 4.3.8). Questa caratteristica è scaturita dall'osservazione del link ACL in condizioni di traffico reale.

Al fine di verificare la validità della seconda *feature* proposta si è proceduto nel seguente modo. Una volta ottenuto il *packet diagram* è stato anche possibile ricavare, per ciascun pacchetto, la tupla (*timestamp*, durata). In tal modo si è potuta analizzare la distribuzione dei tempi di interarrivo dei pacchetti intercettati. Nel caso di trasmissioni dati (ACL) questa analisi ha portato al risultato di Fig. 4.3.9.

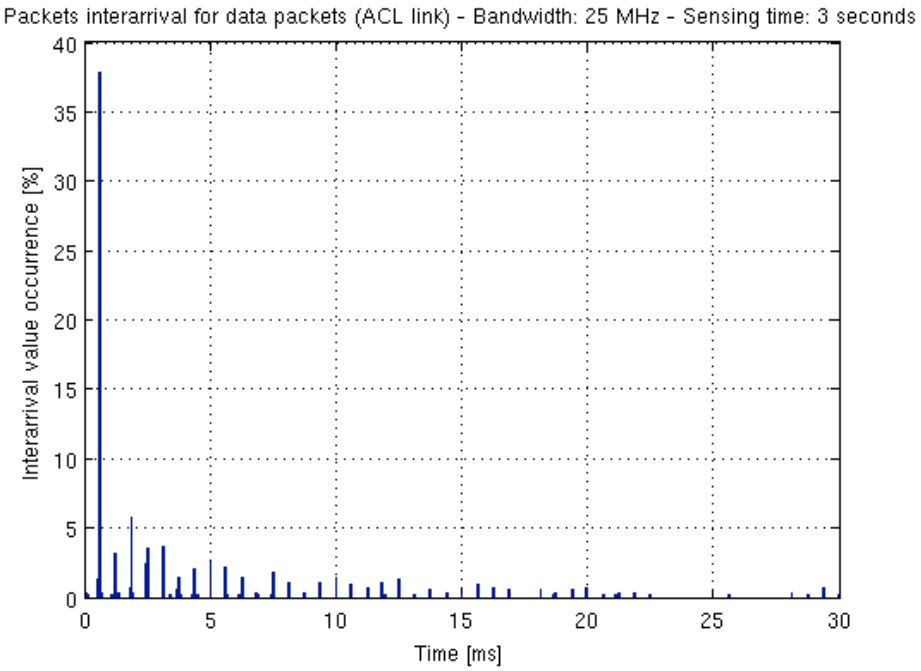


Fig. 4.3.9: Distribuzione del tempo di interarrivo dei pacchetti (ACL link)

In Fig. 4.3.9 è illustrata la distribuzione dei tempi di interarrivo per i pacchetti dati rilevati in 3 s su una banda di 25 MHz mediante l'ED. Anche in questo caso, per il calcolo della *short-term energy*, si è adottata una finestratura rettangolare da 250 campioni con overlapping del 50%. Il picco in Fig. 4.3.9 pari a 628 *μs* (37.8% dei valori del tempo di interarrivo) corrisponde, a meno dello 0.48%, alla durata di un *time slot* (T_{SLOT} = 625 *μs*). I picchi successivi corrispondono a incrementi di un T_{SLOT}. Ciò appare ragionevole pensando a un link in cui sono presenti ritrasmissioni, ACK e pacchetti di 3 diverse durate. Nel caso, quindi, di trasmissioni dati il tempo di interarrivo pari a 1 T_{SLOT} è l'unico che sembra offrire una sufficiente robustezza.

4.4 Durata e tempo di interarrivo dei pacchetti (link SCO)

Un altro interessante scenario oltre quello del trasferimento dati è quello di comunicazioni di tipo voce basate su link SCO (pacchetti HV1, HV2, HV3), molto comuni nel caso di impiego di auricolari senza fili. In questo caso la trasmissione avviene tra un cellulare e l'auricolare a corto raggio (~1 m). L'istogramma calcolato a partire dall'osservazione del *packet diagram*, in questo caso, restituisce un pattern diverso, caratterizzato da pacchetti di lunghezza ~420 μs come si può vedere dalla figura (Fig. 4.4.1).

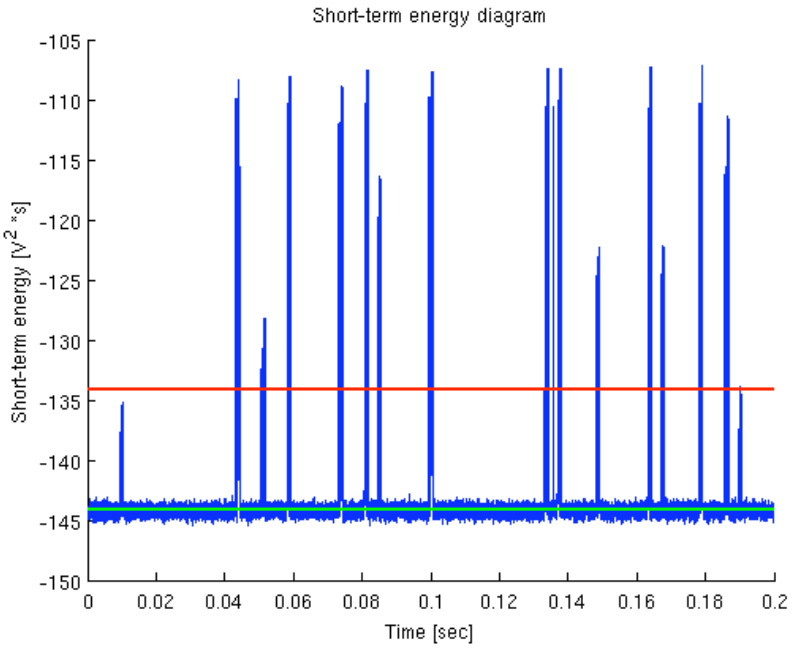


Fig. 4.4.1: Short-term energy in uno scenario di trasmissione voce (SCO)

Lo schema seguente (Fig. 4.4.2) riassume dimensione e struttura dei pacchetti nel caso di una trasmissione di tipo voce.

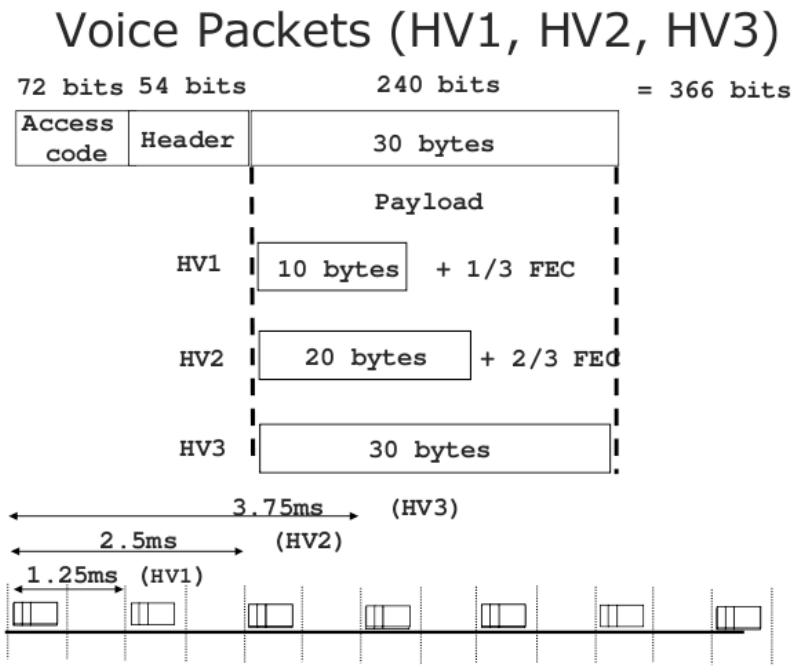


Fig. 4.4.2: Pacchetti HV per link di tipo voce (SCO)

I pacchetti HV1, HV2 e HV3 si ha una dimensione massima di 366 bit (366 μs) e quindi di un solo slot (625 μs). Fra le 3 tipologie di pacchetto, la differenza sostanziale è il tempo che intercorre tra 2 pacchetti consecutivi detto anche tempo di interarrivo. Per i pacchetti HV1, HV2 e HV3 questo risulta pari ripettivamente a 1250 μs, 2500 μs e 3750 μs.

Le misure effettuate, su una banda di 25 *MHz* e per un sensing timedi 3 *s*, per verificare la distribuzione statistica delle durate dei pacchetti in trasmissioni di tipo voce (SCO) hanno prodotto il risultato di Fig. 4.4.3.

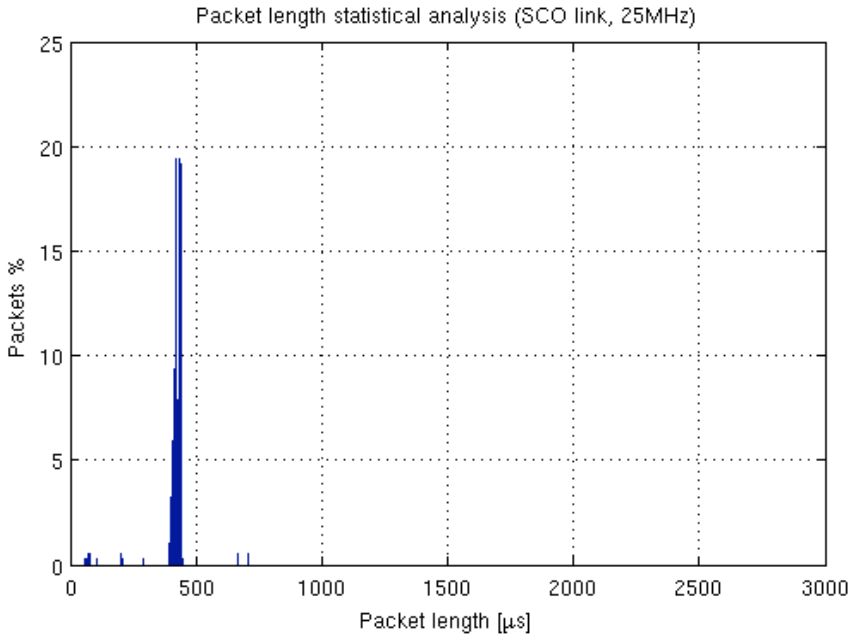


Fig. 4.4.3: Distribuzione della durata dei pacchetti per trasmissioni voce

Come si può notare la trasmissione voce è caratterizzata esclusivamente da pacchetti da un solo slot (durata minore di 625 μs). Nell'intervallo 420 – 440 μs, cade più della metà (58%) dei pacchetti rilevati. Nel caso di trasmissione voce, era interessante verificare se le periodicità introdotte dalla presenza del parametro T_{SCO}, fossero facilmente estraibili osservando l'istogramma dei tempi di interarrivo. Allo stesso modo di quanto fatto per il caso di link ACL, si è provveduto a graficare la distribuzione dei tempi di interarrivo nel caso di link SCO (Fig. 4.4.4).

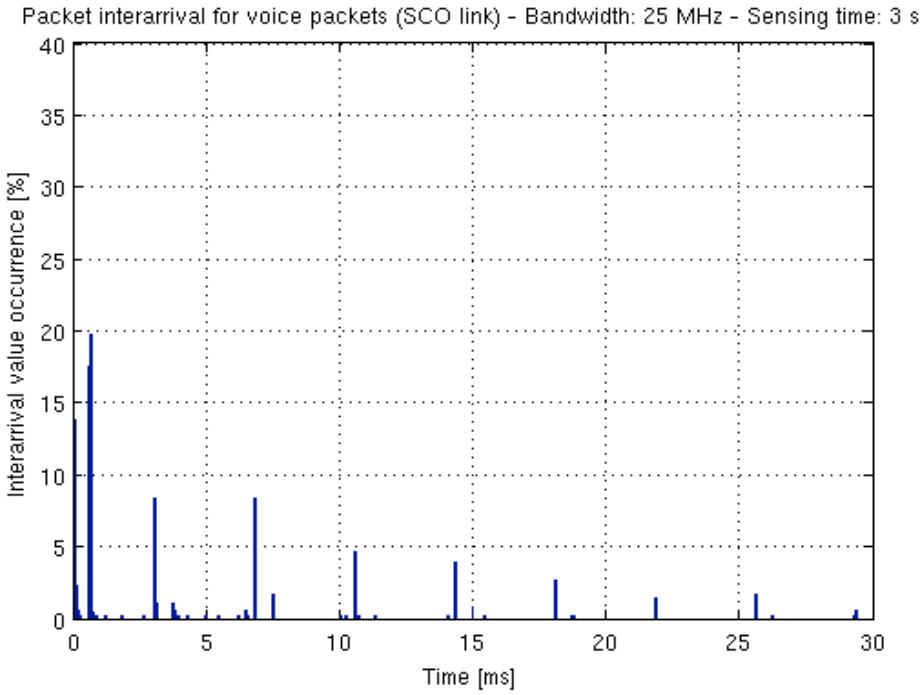


Fig. 4.4.4: Distribuzione del tempo di interarrivo dei pacchetti (SCO link)

In Fig. 4.4.4 si vede che, a differenza del caso di trasmissione dati, nell'istogramma relativo ai tempi di interarrivo dei pacchetti HV, siano presenti dei picchi corrispondenti ai valori: 625 *μs*, 3070 *μs*, 3770 *μs*, 6860 *μs*, 7490 *μs*, ecc.

Il primo valore è pari a T_{SLOT}. I tempi di interarrivo seguenti, come già anticipato nel par. 4.2 ed illustrato in Fig. 4.2.1, sono caratteristici di una trasmissione voce che impieghi pacchetti HV3 con un T_{SCO} pari a 3750 *μs*. I valori previsti dallo Standard sono: 625 *μs*, 3125 *μs*, 3750 *μs*, ecc. I valori ricavati su un tempo di sensing di 3 secondi, per una trasmissione voce con pacchetti HV3 (Fig. 4.4.4), presentano uno scarto relativamente basso rispetto ai valori dello Standard: dello 1.76% rispetto a 3125 *μs* (T_{SCO} - T_{SLOT}), dello 0.53% rispetto a T_{SCO}. Dal confronto dei valori ricavati dallo Standard con quelli ricavati dall'istogramma di Fig. 4.4.4, risulta chiaro che tale *feature* sia facilmente estraibile.

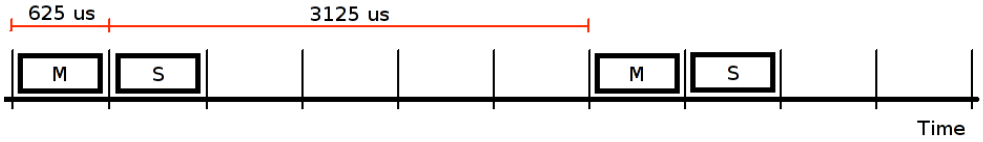


Fig. 4.4.5: Tempi di interarrivocaratteristici per una trasmissione voce (HV3)

Dai risultati ottenuti è, inoltre, possibile immaginare una identificazione del tipo di traffico (dati o voce) intercettato sulla base delle diverse distribuzioni dei tempi di interarrivo dei pacchetti. Nel caso particolare di link SCO, si potrebbe pensare di sfruttare, mediante CR, i periodi di silenzio periodici presenti in caso trasmissioni voce. Quando, ad esempio, si è in presenza di una sola comunicazione voce, tali *silence gaps*, possono arrivare a valori pari a 4 T_{SLOT} ovvero 2500 μs (vedi Fig. 4.4.5). Ulteriori studi potranno fornire una risposta in tal merito verificando alcuni possibili scenari.

4.5 Recognition time di una trasmissione Bluetooth

Una volta caratterizzato il traffico Bluetooth reale, sia per ciò che riguarda le durate dei pacchetti che il loro tempo di interarrivo, si è provveduto a studiare un possibile semplice algoritmo di riconoscimento. Quello proposto si basa sull'osservazione del *packet diagram*, considerando la durata tipica osservata nel caso di pacchetti da 1, 3 e 5 slot.

Nell'algoritmo sviluppato, il *sensing time* in presenza di segnale Bluetooth, corrisponde al tempo necessario ad osservare N pacchetti Bluetooth (di durata 1-slot OR 3-slot OR 5-slot), nel corso di comunicazioni di tipo dati (ACL link). L'analisi statistica dell'occupazione spettrale (Spectral Occupancy Statistics) e il calcolo del tempo minimo di riconoscimento attraverso misure basate su *energy detection*, è stato affrontato anche in [9]. In questo caso, si è voluto descrivere la distribuzione dei pacchetti scambiati nel tempo (*packets exchange pattern*), al variare della banda analizzata e del numero dei pacchetti da intercettare.

Utilizzando i dati raccolti per trasmissioni dati, relative alla frequenza dei 3 tipi di pacchetto (1, 3 e 5 slot), si è potuto simulare il riconoscimento dei pacchetti scambiati da 2 *device* Bluetooth, intercettando quelli inviati in porzioni dello spettro pari a: 1, 5, 10 e 25 *MHz*. Per simulare il codice di FH, si è scelto di impiegare la funzione "rand" di MATLAB sui 79

possibili canali. Il risultato di tale simulazione è stato quello graficato in Fig. 4.5.1.

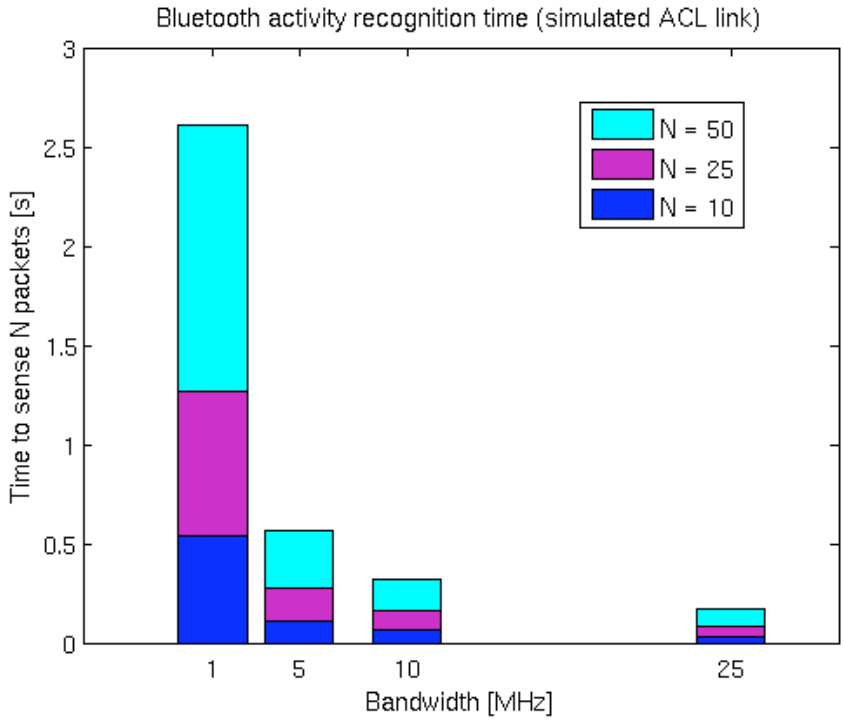


Fig. 4.5.1: Recognition time nel caso di trasmissione dati simulata (ACL link)

Ovviamente, come si vede dalla Fig. 4.5.1, il *recognition time* diminuisce al crescere della banda considerata. Tuttavia, si nota una brusca variazione del tempo di riconoscimento nel passaggio da 1 a 5 *MHz*. Considerando le bande 1, 5, 25 *MHz* ovvero 5 volte il numero di canali della banda precedente, si ha che il tempo di riconoscimento si divide per lo stesso fattore. Nel caso del Bluetooth (pacchetti da 366 μs, 1622 μs e 2870 μs, spaziati da almeno uno slot di reply di 625 μs) si è ottenuto un tempo di *sensing* per 50 pacchetti in 1 *MHz* pari a circa 2.610 *s* (mediando su 100 misure). Lo stesso numero di pacchetti su 25 *MHz* di banda, è stato osservato in appena 0.172 *s*.

Al fine di analizzare il traffico dati reale, il *setup* sperimentale prevedeva la presenza di 2 adattatori Bluetooth posti a distanza di circa 1 *m* tra loro. Tra questi dispositivi è stato posto l'USRP2 (distanza da entrambi pari a circa 0.5 *m*) in condizioni tali da ottenere un ottimo SNR, rispetto a segnali provenienti da altri dispositivi nelle vicinanze (AP WiFi, ecc.). Tra i due device (EDR 2.0) è stato quindi inviato un file (~6 *MBytes*) in un tem-

po di poco più di 20 secondi. La cattura dei campioni di segnale ricevuto è avvenuta a trasmissione avviata e si è conclusa prima del completamento dell'invio (~5 *s* di cattura).

Dalla sequenza di 25 *MHz* ottenuta dall'USRP2, sono state ricavate (mediante filtraggio) 4 sequenze: 1, 5, 10, 25 *MHz*. Utilizzando la medesima sequenza di partenza, è stata garantita la perfetta sincronizzazione sull'istante di inizio delle 4 tracce di diversa banda, in modo da permettere un confronto tra i risultati ottenuti. Le 4 tracce sono state ottenute tutte a partire dal primo canale intercettato nella sequenza di 25 *MHz* (canale 1, 2.402 *GHz*). In questo modo la sequenza rappresentante 5 *MHz* include i pacchetti intercettati nella sequenza da 1 *MHz*, e così via.

Per ciascuna traccia è stato inoltre valutato il corrispondente valore di *Noise Floor*, ed è stata applicata la regola già vista per la definizione della soglia (+10 *dB* dal Noise Floor). Lo *short-term energy diagram* in questo caso è risultato essere quello di Fig. 4.5.2.

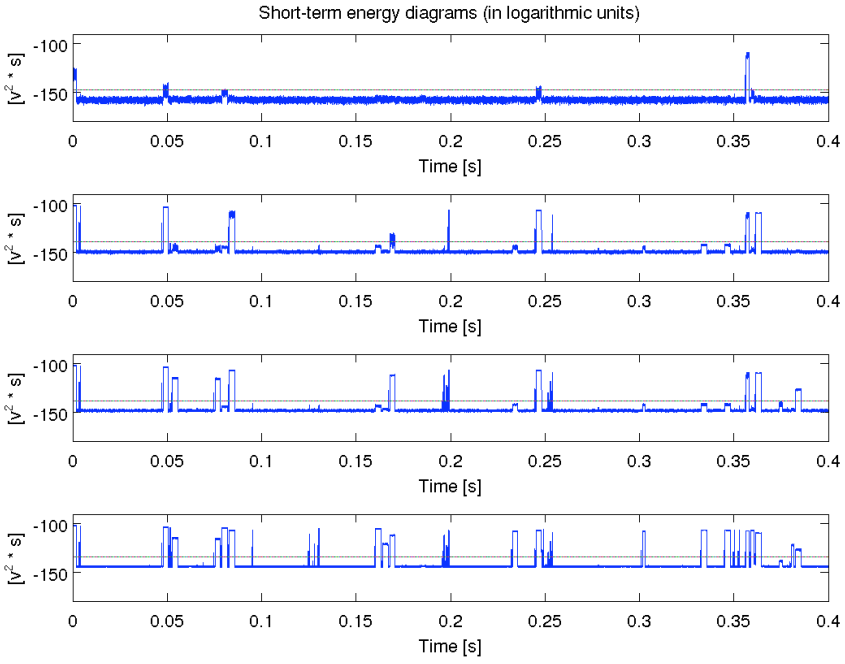


Fig. 4.5.2: Short-term energy diagram (sensing time di 400 ms) su diverse bande, dall'alto: 1, 5, 10, 25 MHz

Nella traccia da 1 *MHz* (in alto in Fig. 4.5.2), si vede chiaramente che risultano catturati solo pochi pacchetti, ovvero quelli inviati sul canale 1 (2.402 *GHz*). Aumentando il numero di canali considerati (5, 10, 25 *MHz*)

si vede come aumentino anche i pacchetti catturati. Applicando la soglia di decisione ai diagrammi di Fig. 4.5.2, si è ottenuto il *packet diagram* delle 4 tracce analizzate. Estraendo da questo *timestamp* e durata dei pacchetti, è stato possibile applicare una regola di riconoscimento basata sul numero di pacchetti Bluetooth intercettati. Tale regola è consistita nel confrontare la durata dei pacchetti rilevati con gli intervalli scelti per i pacchetti da 1, 3 o 5 slot (estratti dalle misure viste nei paragrafi precedenti).

```
[68 μs, 400 μs]        per i pacchetti da 1 slot
[1500 μs, 1650 μs]     per i pacchetti da 3 slot
[2800 μs, 2900 μs]     per i pacchetti da 5 slot
```

Successivamente, si sono utilizzate le tuple (*timestamp*, durata) dei pacchetti appena filtrati, per calcolare il tempo necessario a rilevare N pacchetti consecutivi per ciascuna delle bande di ricezione considerate. Il periodo così misurato è stato calcolato dall'istante di inizio dell'osservazione (*sensing start*) al termine dell'ultimo degli N pacchetti catturati. In tal modo, si è ottenuto l'andamento del tempo di decisione in un intervallo di osservazione di 3 *s* al variare della banda considerata e del numero di pacchetti classificati (vedi Fig. 4.5.3).

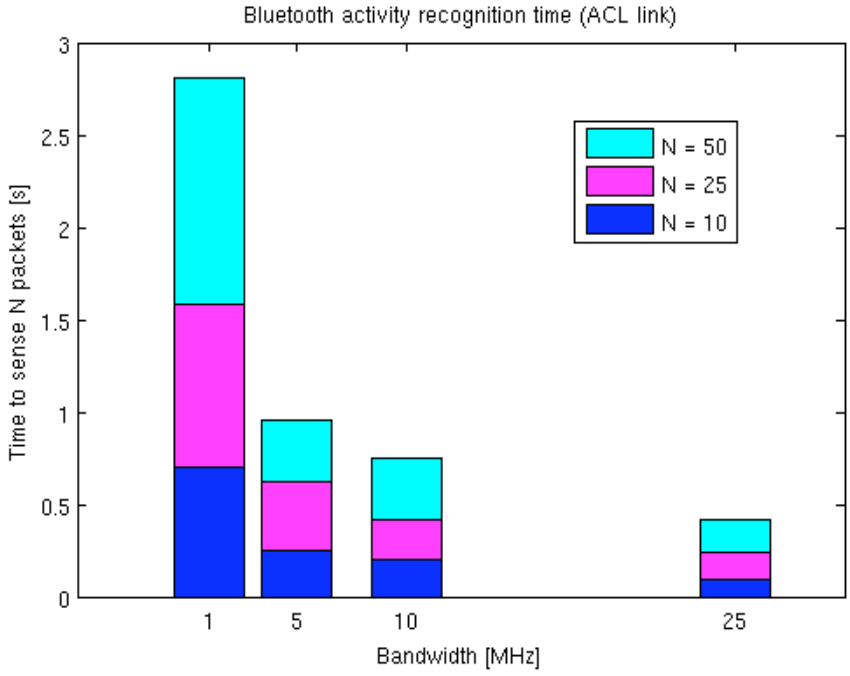


Fig. 4.5.3: Recognition time nel caso di trasmissione dati reale (ACL link)

La particolarità in Fig. 4.5.3, già vista nella precedente simulazione (Fig. 4.5.1), è data dalla presenza di una variazione brusca del tempo di riconoscimento di N pacchetti, nel passaggio da una banda di 1 *MHz* a 5 *MHz*. Tale variazione, resta poi abbastanza ridotta passando a bande di larghezza via via superiore (10 *MHz*, 25 *MHz*). Valutando il rapporto tra i tempi di riconoscimento nelle bande da 1, 5 e 10 *MHz*, della Fig. 4.5.1 e poi della Fig. 4.5.3, si conferma il fatto che essendo presente un pattern di salto (*frequency hopping pattern*) basato su codice PN, risulta che i pacchetti intercettati si distribuiscono uniformemente su tutti i canali (80 *MHz*). Ciò resta ovviamente valido considerando porzioni ridotte della banda (25 *MHz*). Nel caso di traffico dati reale il tempo di riconoscimento di 50 pacchetti in 1 *MHz*, è risultato pari a circa 2.7 *s*, mentre con una banda di 25 *MHz* è sceso al valore di circa 0.4 *s*.

Dall'analisi svolta sulla tecnologia Bluetooth, per l'estrazione di alcune feature relative ai pacchetti scambiati, è emerso che almeno due caratteristiche (durata e tempo di interarrivo dei pacchetti) possono essere efficientemente estratte e impiegate per classificare tale tecnologia. Tale processo di estrazione, similmente a quanto visto in [1], richiede l'impiego di un semplice *energy detector*. L'impiego di *features* di livello MAC, relative quindi ai pacchetti, consente di avere un sistema di *spectrum sensing* notevolmente flessibile.

Lo studio congiunto delle specifiche pubblicate nei diversi Standard, permette di considerare features uniche per ciascuna tecnologia, così da consentire una agevole classificazione. Il modulo che racchiude *energy detector*, estrattore di features e classificatore, prende il nome di AIR-AWARE. Gli studi che seguiranno questo lavoro, avranno come obbiettivo quello di integrare, nel classificatore del modulo AIR-AWARE, il maggior numero possibile di tecnologie wireless operanti in banda ISM.

Attualmente il progetto prevede lo studio congiunto di tre tecnologie molto comuni: WiFi, Bluetooth e ZigBee. Il prossimo passo consisterà nello studio approfondito dello Standard ZigBee, al fine di evidenziare altre *features*, in grado di facilitare la classificazione in presenza di trasmissioni WiFi e Bluetooth.

4.6 APPENDICE A

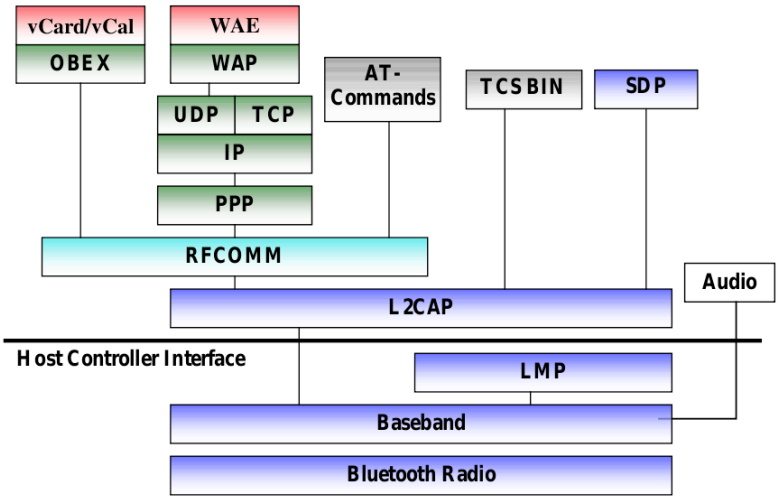
Lo stack protocollare Bluetooth

Il sistema Bluetooth presenta una complessa struttura protocollare a stack specificatamente designata per garantire efficienza e mantenere una alta interoperabilità, mediante l'adozione di protocolli di rete come il PPP (Point To Point Protocol, impiegati per trasportare traffico IP.

L'insieme di moduli che concorrono a creare la pila protocollare nel Bluetooth può essere riassunto con un primo strato (*physical layer*) detto Baseband. In esso si effettuano tutte le operazioni di mo-demodulazione e decodifica del segnale Bluetooth. Un secondo strato è rappresentato dal Link Manager, il quale permette di gestire le connessioni attraverso un opportuno traffico di segnalazione (Link Manager Protocol, LMP).

Successivamente il traffico attraversa una importante interfaccia detta Host Controller Interface (HCI) la cui implementazione, nei dispositivi connessi tramite USB, permette lo scambio di dati e comandi tra Host e Controller Bluetooth. Nel caso di host rappresentato da un calcolatore con sistema operativo GNU/Linux, è possibile inviare comandi al Controller Bluetooth in modo da attivare specifiche funzionalità (ad es. *inquiry mode*, *master/slave role switch*, ecc.). Altra possibilità data dall'interfaccia HCI è quella di catturare i pacchetti in transito da e per gli strati superiori escludendo tuttavia tutto il traffico di controllo gestito da LMP.

In GNU/Linux, oltre l'HCI lo stack Bluetooth è gestito, in genere, dai moduli installati sul sistema operativo. Per il Bluetooth esistono diverse implementazioni dello stack protocollare, tuttavia la più utilizzata è senz'altro quella che va sotto il nome di BlueZ.

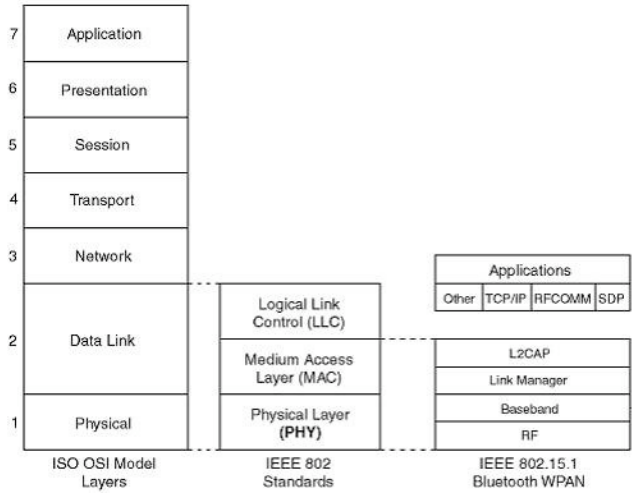


Bluetooth Protocol Stack

Si hanno due aree ben distinte separate da un'interfaccia denominata HCI (Host Controlled Interface). Questa consente di inviare comandi al controller del modulo Baseband, al Link Manager, di accedere ai registri di controllo e a quelli indicanti lo stato del device Bluetooth.

La garanzia di una elevata interoperabilità del sistema BT è data dall'impiego di protocolli già utilizzati per altre applicazioni e quindi ben testati, come: l'OBEX (Object Exchange Protocol), il PPP (Point to Point Protocol), l'Internet Protocol, l'UDP e TCP.

Il Bluetooth, inoltre, definisce alcuni nuovi protocolli denominati LMP (Link Manager Protocol), L2CAP (Logical Link Control and Adaptation layer Protocol), RFCOMM (Radio Frequency COMMunication), SDP (Service Discovery Protocol e TCS (Telephony, Control Protocol). Lo strato più basso, denominato Bluetooth Radio assieme a quest'ultimi appena citati, sono stati progettati ad hoc per rispondere alle esigenze del sistema BT.



Studiando a fondo le peculiarità delle più comuni implementazioni commerciali del sistema Bluetooth si possono scoprire alcune problematiche relative alla possibilità di catturare pacchetti dagli strati MAC/PHY. Lo stack Bluetooth è come già detto articolato in diversi layers, due dei quali al di sotto dell'Host Controller Interface (HCI): il Baseband e il Link Manager Protocol (LMP).

L'HCI rappresenta l'interfaccia tra ciò che viene detto il Controller Bluetooth (il dispositivo vero e proprio) e gli strati superiori. Tale controller è in genere progettato come dispositivo esterno (dongle USB) o in forma di scheda integrata. Al di sopra del controller Bluetooth, quindi a partire dall'HCI in poi, l'implementazione avviene attraverso moduli software. In sistemi operativi GNU/Linux, tali moduli di interfaccia sono gestiti dal kernel del sistema operativo. In particolare, nel caso di dongle USB Bluetooth, si hanno moduli del kernel in grado di gestire la porta USB alla quale l'adattatore è connesso. Una caratteristica di questa architettura è data dal fatto che al di sotto dell'interfaccia HCI, i diversi strati sono implementati in hardware o software embedded proprietario (firmware) e questi non prevedono in genere alcun metodo per impostare la modalità "monitor" (ovvero di ascolto delle comunicazioni).

Si sono sviluppate recentemente alcune soluzioni commerciali basate su appositi Bluetooth USB dongles aventi speciali *firmware* in grado di catturare il traffico di una piconet intercettando il FH code impiegato. La presenza di più *piconet*, tuttavia, anche in questo caso rende impossibile la cattura di tutti i pacchetti scambiati a causa dell'impiego di diversi codici di salto per ogni *piconet*. Si capisce quindi che l'estrazione di *features* di livello MAC di questa tecnologia non è realizzabile attraverso l'uso di dispositivi Bluetooth.

4.7 APPENDICE B

Installazione e configurazione di GNUradio

Il software GNUradio è un ambiente di sviluppo per radio SDR open-source liberamente scaricabile e modificabile. I sorgenti sono disponibili dal sito del progetto (http://gnuradio.org) attraverso i repository GIT. Dai sorgenti è possibile installare GNUradio sia su GNU/Linux che su Windows o MAC/OS. Questa breve guida all'installazione, d'ora in avanti si riferirà al caso di installazione di GNUradio su GNU/Linux Ubuntu (con particolare riferimento alla release 9.10 "Karmic Koala"). Per altre informazioni di installazione su un differente sistema operativo si può fare riferimento al wiki di gnuradio.org:

http://gnuradio.org/redmine/wiki/gnuradio/BuildGuide

Prima di installare GNUradio, come avviene in genere per tutte le installazioni su Linux, è necessario disporre di tutti i pacchetti da cui esso dipende. Nel caso di GNUradio questa procedura è particolarmente laboriosa a causa delle numerose dipendenze che è necessario soddisfare. In particolare sono necessari i seguenti componenti.

La lista completa delle dipendenze è anche su:

http://gnuradio.org/redmine/repositories/changes/gnuradio/README

- Tools di sviluppo
  - g++
  - subversion
  - make
  - autoconf, automake, libtool
  - sdcc
  - guile
  - ccache

- Librerie
  - Python-dev
  - FFTW 3.x (fftw3, fftw3-dev) per il calcolo della FFT
  - cppunit (libcppunit, libcppunit-dev
  - Boost (1.35 o succesiva)
  - libusb, libusb-dev
  - wxWidgets (wx-common) e wxPython (python-wxgtk2.8)
  - python-numpy (python-numpy-ext)
  - ALSA (alsa-base,libasound2, libasound2-dev)
  - Qt (libqt3-mt-dev)
  - SDL (libsdl-dev)
  - GSL GNU scientific library

- SWIG (1.3.31 o successivo)1
- QWT (opzionale, 5.0.0 o successivo)
- QWT Plot3d Lib
- Doxygen (opzionale)
- Octave (opzionale)

In Ubuntu GNU/Linux 9.10 è possibile soddisfare tutte le dipendenze fondamentali attraverso il seguente comando di apt-get:

$ sudo apt-get -y install swig g++ automake libtool

python-dev libfftw3-dev libcppunit-dev libboost1.38-dev libusb-dev for-t77 sdcc sdcc-libraries libsdl1.2-dev python-wxgtk2.8 subversion git-core guile-1.8-dev libqt4-dev python-numpy ccache python-opengl libgsl0-dev python-cheetah python-lxml doxygen qt4-dev-tools libqwt5-qt4-dev libqwtplot3d-qt4-dev pyqt4-dev-tools

Una volta preparato il sistema operativo è necessario ottenere una copia dei sorgenti di GNUradio. Si può scegliere di installare l'ultima versione

“stable” rilasciata oppure optare per la corrente versione di sviluppo (development trunk). Questa seconda scelta in genere è preferibile se si vogliono provare le nuove funzionalità così come avviene nel caso di questo lavoro.

Digitando la seguente riga da shell:

```
$ git clone http://gnuradio.org/git/gnuradio.git
```

Si può ottenere l'ultima versione in fase di sviluppo. Il comando git crea automaticamente una cartella “gnuradio” contenente tutti i file necessari per la compilazione di GNUradio. Una volta ottenuta la cartella .../gnuradio si può procedere con la compilazione, digitando:

```
<yourPath>/gnuradio/$ ./bootstrap
$ ./configure –prefix=/<yourPath>/gnuradio
$ make
$ make install
```

A questo punto si può procedere alla creazione del gruppo di utenti “USRP” e si può assegnare ad esso il nostro profilo utente.

```
$ sudo addgroup usrp
$ sudo usermod -G usrp -a <YOUR_USERNAME>
$ echo 'ACTION=="add", BUS=="usb", SYSFS{idVendor}=="fffe", SYSFS{idProduct}=="0002", GROUP:="usrp", MODE:="0660"' > tmpfile
$ sudo chown root.root tmpfile
$ sudo mv tmpfile /etc/udev/rules.d/10-usrp.rules
```

In questo modo si garantiscono i necessari permessi all'utente corrente nei confronti di GNUradio e dell'USRP.

N.B.: controllare che dopo tale operazione sia presente un solo file 10-

usrp.rules in /etc/udev/rusles.d/. In caso contrario eliminare i duplicati per evitare problemi ad UDEV con conseguente blocco del gdm (Gnome Desktop Manager) in avvio.

Successivamente si può modificare il file /home/<nomeUtente>/.bashrc per inserire le opportune variabili d'ambiente. Una possibile configurazione è la seguente:

Nel file /home/sergio/.bashrc:

```
#-----------------ADDED BY USER SERGIO-----
export PATH=
$PATH:/home/sergio/gnuradio/bin

export LD_LIBRARY_PATH=
$LD_LIBRARY_PATH:/home/sergio/gnuradio/lib

export PKG_CONFIG_PATH=
$PKG_CONFIG_PATH:/home/sergio/gnuradio/lib/pkgconfig

export PYTHONPATH=
$PYTHONPATH:
/home/sergio/gnuradio/lib/python2.6/site-packages

source /home/sergio/.gnuradiorc
#------------------------ ----------------
```

L'installazione è completa. Per verificare il corretto funzionamento di GNUradio è possibile connettere un USRP alla porta USB e verificare il funzionamento di uno degli script di esempio in .../gnuradio/gnuradio-examples/python/usrp/ ovvero ad esempio usrp_benchmark_usb.py. Tale script Python verifica il throughput massimo ottenibile da e verso l'USRP (32 MB/s) mediante una serie di test a velocità via via crescenti.

Ecco un possibile risultato corretto.

```
Testing 2MB/sec... usb_throughput = 2M
ntotal    = 1000000
nright    = 997526
runlength = 997526
delta     = 2474
OK
Testing 4MB/sec... usb_throughput = 4M
ntotal    = 2000000
nright    = 1995797
runlength = 1995797
delta     = 4203
OK
Testing 8MB/sec... usb_throughput = 8M
ntotal    = 4000000
nright    = 3995449
runlength = 3995449
delta     = 4551
OK
Testing 16MB/sec... usb_throughput = 16M
ntotal    = 8000000
nright    = 7972791
runlength = 7972791
delta     = 27209
OK
Testing 32MB/sec... usb_throughput = 32M
ntotal    = 16000000
nright    = 15985347
runlength = 15985347
```

delta = 14653

OK

Max USB/USRP throughput = 32MB/sec

A questo punto è possibile cominciare lo sviluppo mediante GNUradio. Il tool GRC (GNUradio Companion) è un ottimo primo strumento per familiarizzare con i moduli predefiniti.

4.8 Bibliografia

Capitolo 1 – Introduzione

[1] J. Mitola III et al., "Cognitive radio: making software radios more personal," IEEE Personal Commun., vol. 6, pp. 13-18, 1999

[2] J. Mitola III, "Cognitive radio for flexible mobile multimedia communications," in Proceedings of the IEEE International Workshop on Mobile and Multimedia Communications, San Diego, Calif, USA, pp. 3-10, 1999

[3] S. Haykin, "Cognitive radio: brain-empowered wireless communications," IEEE Journal on Selected Areas in Communications, vol. 23, pp. 201-220, 2005

[4] Bruce Fette, "Cognitive Radio Shows Great Promise", COTS Journal, October 2004

[5] N. Golmie, R. E. Van Dyck, A. Soltanian, "Interference of Bluetooth and IEEE 802.11: simulation modeling and performance evaluation", in Proceedings of the 4th ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, Rome, Italy, pagg. 11-18, 2001

[6] FCC, Cognitive Radio NPRM, ET Docket no.03-108, Notice of Proposed. Rule Making and Order, 18 FCC Rcd 26859, 2003

[7] IEEE Computer Society, IEEE Std. 802.15.2, "Coexistence of Wireless Personal Area Networks with Other Wireless Devices Operating in Unlicensed Frequency Bands", 2003

[8] FCC, "Unlicensed operation in the TV broadcast bands", NPRM, RCC Docket No. 04-113, Maggio 2004

Capitolo 2 – La tecnologia Bluetooth

[1] IEEE Computer Society, IEEE Std. 802.15.1 "Part 15.1: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs)", 14 Giu 2005

[2] IEEE Computer Society, IEEE Std. 802.11 “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications ”, 12 Giu 2007

[3] Bluetooth Special Interest Group (SIG), "Bluetooth Core Specification v4.0", 17 Dic 2009

Capitolo 3 – GNUradio e l'USRP

[1] Eric Blossom, "Exploring GNU Radio", http://www.gnu.org/software/gnuradio/doc/exploring-gnuradio.-html

[2] Eric Blossom, “How to write a signal processing block”,

http://www.gnu.org/software/gnuradio/doc/howto-write-a-block.-html

[3] gnuradio.org wiki, suggested readings, http://gnuradio.org/redmine/wiki/gnuradio/SuggestedReading

[4] Hogenauer, Eugene B., "An economical class of digital filters for decimation and interpolation", Aprile 1981, IEEE Transactions on Acoustics, Speech and Signal Processing 29 (2): 155–162

[5] Matthew P. Donadio, CIC Filter Introduction, 18 July 2000,

http://users.snip.net/~donadio/cic.pdf

Capitolo 4 – Identificazione del segnale Bluetooth

[1] M.G. Di Benedetto, Stefano Boldrini, Carmen Juana Martin Martin, and Jesus Roldand Diaz, “Automatic network recognition by feature extraction: a case study in the ISM band”,

[2] S. M. Kay, “Fundamentals of Statistical Signal Processing, Volume 2: Detection Theory“, Prentice Hall, 1998

[3] Harry Urkowitz, "Energy detection of unknown deterministic signals," Proceedings of IEEE, vol. 55, no. 4, pp. 523-531, April 1967

[4] D. Cabric, A. Tkachenko, and R. W. Brodersen, “Experimental Study of Spectrum Sensing based on Energy Detection and Network Cooperation,” in Proceedings of Workshop on Technology and Policy for Accessing Spectrum (TAPAS), Boston, MA, USA, August 2006

[5] Hsue and Samir S. Soliman, “Automatic modulation classification using zero-crossing“ IEEE Proceedings, vol.137, Pt. F, No.6, pp.459-464,. December 1990

[6] Z. Yu, Y. Q. Shi, and W. Su, "M-ary frequency shift keying signal classification based-on discrete Fourier transform," in Proceedings of the IEEE Military Communications Conference (MILCOM '03), vol. 2, pp. 1167-1172, 2003

[7] Jo Lynn Tan, Ahmad Zuri bin Sha'ameri, “Signal analysis and classification of digital communication signals using Adaptive Smooth-Windowed Wigner-Ville Distribution”, in Proceedings of IEEE 2008 6th National Conference on Telecommunication Technologies, Putrajaya, Malaysia, 26-27 August 2008

[8] C. Regazzoni et al., "Use of Time Frequency Analysis and Neural Networks for Mode Identification in a Software Radio Based Wireless measurement equipment", EURASIP, 2004

[9] Sithamparanathan Kandeepan, Radoslaw Piesiewicz, Tuncer C. Aysal, Abdur Rahim Biswas, Imrich Chlamtac , ”Spectrum Sensing for Cognitive Radios with Transmission Statistics: Considering Linear Frequency Sweeping ”, EURASIP Journal on Wireless Communications and Networking , Volume 2010, 13 pagine

[10] Zhuan Ye, Gokhan Memik, John Grosspietsch, "Energy detection using estimated noise variance for spectrum sensing in cognitive radio networks", IEEE Wireless Communication and Networking Conference (WCNC), pp. 711-716, Aprile 2008

4.9 Allegato A (paper CogART 2010)

Identification of packet exchange patterns based on energy detection: the Bluetooth case

Sergio Benco (*,**), Stefano Boldrini (*), Andrea Ghittino (**), Stefano Annese (**), and Maria-Gabriella Di Benedetto, *Senior Member, IEEE* (*)

(*) "Sapienza" University of Rome, School of Engineering, INFOCOM Dpt., ACTS lab.
(**) CSP "ICT Innovation", Turin, Italy

Abstract* — *A time-domain recognition of different wireless technologies may be obtained using energy detection. In this work, an energy detector was implemented using the Universal Software Radio Peripheral SDR platform. The energy detector output allows the formation of a packet presence/absence diagram. Experimental results indicate that the observation of Bluetooth packet exchange patterns reveals technology-specific MAC layer procedures, leading to the conclusion that technology recognition can be obtained on the basis of time-domain technology-specific features.

Keywords* — *cognitive networking; network discovery; automatic network classification; energy detection; universal software radio peripheral

I. Introduction

Automatic classification amongst different technologies in the ISM band based on MAC features was first analyzed in [1], in the framework of the AIR-AWARE Project. This project aims at creating a black box – the AIR-AWARE module – capable of classifying technologies, as well as different types of interference in play.

Many wireless technologies, such as Wi-Fi (IEEE Std 802.11), Bluetooth (IEEE Std 802.15.1) and ZigBee (IEEE Std 802.15.4), operate in the ISM 2.4 *GHz* band. Every technology has its own particular MAC sublayer behaviour, as defined by the Standard specifications. Recognition of this behaviour can reveal that the corresponding technology is currently present in the air. Thanks to this approach, the recognition can be done using a generic device, such as an energy detector. This device does not have to demodulate the received signal, it only detects how much energy is present in the air in time (i.e. with a reasonable sampling frequency), to determine whether a packet is currently being sent or not. By analyzing then the time-domain diagram of presence vs. absence of packets, it can recognize features, that are specific of different technologies. In this way, it can reveal those technologies that may be currently active in the area.

Manuscript submitted July 15, 2010. This work was developed under the trilateral agreement of "Sapienza" University of Rome, "Politecnico di Torino", and the CSP "ICT Innovation" (Turin, Italy). This work was supported in part by the European Commission in the framework of COST Action IC0902: Cognitive Radio and Networking for Cooperative Coexistence of Heterogeneous Wireless Networks.
For references please contact the INFOCOM Department, School of Engineering, "Sapienza" University of Rome, Via Eudossiana 18, 00184, Rome, Italy. E-mail address: dibenedetto@newyork.ing.uniroma1.it.

In this work, the ISM 2.4 *GHz* band is taken into account, and the Bluetooth technology [2, 3] is analyzed. A "Universal Software Radio Peripheral" (USRP) Software Defined Radio (SDR) platform is used for energy detection. The USRP output consists in received signal samples used by the energy detector to obtain the temporal pattern of the short-term energy. This trend leads to a diagram that indicates the presence vs. absence of packets, the packet durations, the "silence" gaps, and the instants at which the packets start (packet timestamps). Based on [2] and [3], some Bluetooth features are proposed and recognized in the diagram. These features are technology-specific, i.e. they are peculiar to Bluetooth and allow to distinguish it from the other technologies operating in the same ISM band. This fact is important for the automatic recognition and technology classification, that is, the final goal of the AIR-AWARE Project.

The paper is organized as follows. Section II contains a detailed description of the USRP mentioned before, while in Section III it is described how it was used for energy detection. Section IV presents how the construction of the packet diagram was obtained from the energy detection, the analysis of this diagram, and the proposed features for the Bluetooth technology. In Section V the experimental results are reported, and these results are then discussed in Section VI, which also contains a guideline for proposed future directions.

II. The Universal Software Radio Peripheral SDR platform

The input data used by the energy detector were obtained through an SDR called USRP. This hardware has recently gained growing attention by the research community given its low cost and open source vision. This kind of SDR comes in two versions: the USRP, that is able to work with a bandwidth of 8 *MHz*, and the USRP2, that has an improved receiver bandwidth (up to 25 *MHz*). The USRP2, adopted in this work, consists in: a) a motherboard that hosts two 100 *MSamples/s* ADCs (14 bits) and two 400 *MSamples/s* DACs (16 bits); b) an FPGA (Xilinx Spartan 3); c) a Gigabit Ethernet controller; d) two slots: one for the receiver (RX) channel and one for the transmitter (TX) channel. These slots enable great flexibility in the USRP2 RF stage, given the simplicity in changing daughterboard (that can implement a RX, or a TX, or a transceiver) to adapt the USRP2 to a broad variety of applications (DVB, GSM/UMTS, Wi-Fi, etc.).

In the presented experimental set-up, a dual channel (TX/RX) board was used, the XCVR2450 that is able to demodulate signals in the ISM bands (2.400-2.483 *GHz* and 4.9-5.8 *GHz*). The adopted antenna was a dual band 2.400-2.483 and 4.9-5.8 *GHz* vertical antenna, with a gain of 3 *dBi* in the lower band. The experimental set-up is completed by: two Bluetooth USB adapters (20 *dBm* Class 1 devices) for the ACL-based file transfer sensing; a headset and a cellular phone for the SCO-based voice transmission sensing. The transmitting Bluetooth devices were placed at a distance of about 1 *m* from the sensing device to get a strong-enough signal in either data and voice transmission test cases.

The receiver bandwidth of about 25 *MHz* is obtained using a quadrature sampling process. This configuration provides a couple of ADCs that can work with a phase difference of exactly $\pi/2$, to produce an In-phase (I) and a Quadrature (Q) sampled signal. Each USRP2 ADC can offer a Nyquist frequency of 50 *MHz*. Due to the adopted quadrature sampling scheme, the complex samples (I+jQ) can, however, perfectly reconstruct a signal whose bandwidth is 100 *MHz* (100 *MSamples/s* I & Q). The received signal, sampled with a bit depth of 14 bits, is then stored in a 32 bits floating point variable (16 bits for I and 16 bits for Q) that can be further analyzed by software. At this point each received complex sample is represented by 4 Bytes. This value multiplied by the sampling rate gives a throughput of 3200 *Mb/s*. To make use of a Gigabit Ethernet interface, the received sequence at a rate of 100 *MSamples/s* has to be decimated by a minimum factor of 4 by the FPGA; this results in a 800 *Mb/s* data flow. In this way using quadrature sampling the USRP receiver bandwidth becomes 25 *MHz*.

Given that the Bluetooth technology provides FHSS (1600 *hops/s*) spread with a channel hopping code over 79 channels of 1 *MHz* each, the ideal way for detecting Bluetooth is to sense the entire 80 *MHz* bandwidth. In our set-up, the USRP2 bandwidth was set to its maximum value of 25 *MHz*. In this band a set of 22 Bluetooth channels was sensed, excluding some heavy attenuated channels on the edges of the selected band due to some RF impairments. The samples captured by the USRP2 were then processed by the energy detector, consisting of MATLAB scripts described in Section III.

III. ENERGY DETECTION

The detection of a random signal immersed in noise is a well-known problem of detection theory [4]. When the received signal is unknown, a standard assumption consists in modelling it as a zero-mean WSS random Gaussian process with variance σ_s^2. Noise can be modelled as Additive White Gaussian Noise (AWGN) with variance σ_n^2. The sufficient statistic $T(\mathbf{r})$ i.e. the expression of energy detector, is then:

$$T(\mathbf{r}) = \sum_{i=1}^{N} |r_i|^2 \qquad (1)$$

where $\mathbf{r}$ is the received sequence, r_i is the i^{th} sample of the sequence, and N is the time window length. Energy detection can also be viewed as an estimator of the variance from a set of N consecutive samples, that results in $T'(\mathbf{r}) = T(\mathbf{r})/N$. $T(\mathbf{r})$ represents short-time energy.

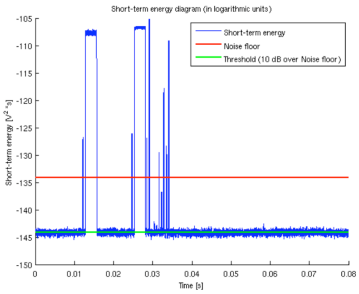
In order to translate sample USRP2 output values onto a Volt scale, the USRP2 was calibrated. The USRP2 output corresponds in fact to linear ADC quantization levels, that can be reduced to voltage based on the input-output USRP2 characteristic. Finally, for a short-term energy value to be obtained, $T(\mathbf{r})$ was multiplied by the sampling period T_S, that is:

$$E_N(\mathbf{r}) = \sum_{i=1}^{N} |r_i|^2 \cdot T_S \qquad (2)$$

Short-term energy was computed with overlapping of 50 % of the window length, while the window length was chosen equal to N=250 samples. This gives rise to sufficient resolution in both short-time energy space and time.

Figure 1 shows an example of short-term energy in time. The green line on figure indicates the average noise value (noise floor). As described in Section IV, this leads to the choice of the adopted threshold (red line in Fig.1).

Figure 1 - Short-term energy diagram vs. time



IV. PACKET PATTERN ANALYSIS AND FEATURE EXTRACTION

A sequence of "high" values of short-term energy indicates that for period of time (packet duration) a useful signal was present over the air interface, i.e. a packet was sent. Conversely, a sequence of "low" values indicates silence, i.e. an inter-packet interval. "High" vs. "low" values are determined against a threshold that must be fixed, where values above vs. below threshold are high vs. low values, respectively. The threshold was fixed by adding 10 *dB* to the noise floor, i.e. the average detected energy in the absence of any received signal in the ISM 2.4 *GHz* band. Computed noise floor was -144.2 *dBJ,* and threshold was set at -134.2 *dBJ*.

Noise peaks may also generate high values. Since, however, the shortest packet defined in the Bluetooth Standard is 68 μs (ID packet), packet filter was implemented that discarded all positive packets lasting less than 50 μs.

An example of the obtained packet diagram is illustrated in Fig. 2.

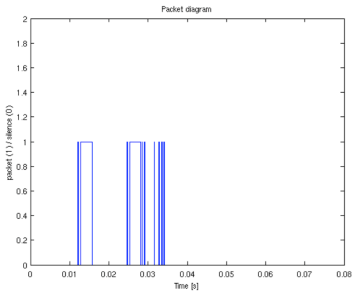


Figure 2 – Packet diagram (with reference to Figure 1)

Based on the packet diagrams, an analysis on possibleMAC sub-layer Bluetooth features was carried out.

Bluetooth technology uses a TDMA/TDD scheme to assure access to users. The slot duration is fixed at $T_{SLOT} = 625\,\mu s$. Another important characteristic of Bluetooth is that packets may occupy 1, 3 or 5 time slots. These three packet types have minimum and maximum allowed lengths. The NULL packet and the POLL packet, that are control packets used for the Acknowledgment (ACK) and for the Polling of the intended recipients, respectively, have a fixed length of 126 bits. At a bitrate of 1 *Mb/s* as specified by [2], durations corresponding to the above packet lengths, are as reported in Table I.

TABLE I. BLUETOOTH PACKET DURATIONS

	Fixed duration	Min duration	Max duration
Time slot	625 μs		
ID packet	68 μs		
NULL / POLL packet	126 μs		
1-time slot packet		126 μs	366 μs
3-time slot packet		1250 μs	1622 μs
5-time slot packet		2500 μs	2870 μs

In the voice transmission case (SCO link), packets are 1-time slot only. The piconet Master provides the Slave with periodic reserved time slots occurring every 2, 4 or 6 time slots, for so-called HV1, HV2 or HV3 packets, respectively; this corresponds to T_{SCO} values of 1.25 *ms*, 2.50 *ms* or 3.75 *ms* respectively. This configuration enables a two-way 64 *kb/s* PCM encoded symmetric voice transmission.

How to take advantage of these characteristics in order to characterize Bluetooth and permit its recognition? Based on the previous analysis of the Bluetooth protocol, two features are proposed: a) packet duration; b) packet inter-arrival interval.

The first proposed feature arises from the following consideration. If sensing is performed within the central portion of a data transmission or a voice call, one can expect that the link manager is segmenting the data by filling efficiently one of the possible SCO packet formats it can send. If this is true, the predominant packet length values fall around their maximum allowed value, and the detected packet durations should be concentrated around the values reported in the first and last columns of Table I.

Regarding the second proposed feature, since the system is TDMA/TDD based, we expect that the packet inter-arrival time will be concentrated around T_{SLOT} (625 μs), given that the energy detector shows all the packets exchanges between the two communicating devices. In the long run, this reveals a frequent inter-arrival period corresponding to one slot duration. Multiple of 625 μs may also be present, corresponding to multi-slot packets.

V. EXPERIMENTATION

The Bluetooth technology provides two different typologies of communication based on specialized transport layer protocols. The first is the ACL (Asynchronous Connection-Less link), able to transport data in a reliable way thanks to acknowledgment packets (NULL packets) and retransmission schemes (ARQ). The second is called SCO (Synchronous Connection-Oriented link), able to convey voice streams (64 *kb/s* PCM encoded) by keeping a constant delay (no retransmissions). These scenarios were reproduced in the following way. To obtain an ACL data link we chose to connect two hosts using two Bluetooth adapters (Class 1 devices) and to transmit one large file (about 6 *MBytes*) between them. Using version 2.0 EDR capable devices, the complete transmission of this file lasted about 24 seconds, that is long enough to sense several packets with the USRP2 placed at about 1 *m* from each device. For the other scenario, the employed devices were: a cellular phone (Nokia N73, 2.0 EDR) and a headset (Nokia BH-100, 2.0 EDR) placed similarly to the previous set-up. In the SCO voice link case, the transmission was established by setting up a voice call.

The first analysis performed on ACL-based data transmission provided the distribution of packet duration values over a certain period of time. We chose to sense 3 *s* of file transfer, corresponding to over 500 ACL data packets in a 25 *MHz* bandwidth. Using packet diagrams as in Figure 2, the histogram of Figure 3 was obtained.

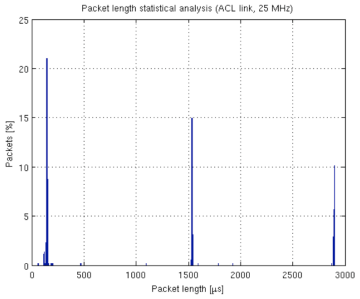


Figure 3 - Packet length statistics (ACL link, 25 *MHz*)

From the histogram one observes that most values are well concentrated around three values centred at: 144 μs, 1540 μs and 2890 μs. These values are related to the duration of the NULL/POLL packets and to the maximum durations of the 3 and 5-time slot packets (see Table I). This is reasonable since the Bluetooth transport layer does its best to encapsulate data into packets as efficiently as possible, and ACKs are needed. The presence of these three peaks indicate that the proposed feature is Bluetooth-specific.

As expected, packet length distribution is clearly different in the voice case, since the only packet type in use is the 1-time slot. Results of measurements are shown in Figure 4.

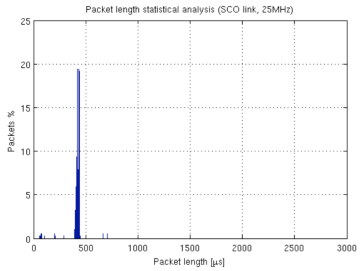


Figure 4 - Packet length statistics (SCO link, 25 *MHz*)

As expected, the packet length distribution in a SCO link is concentrated around one value (430 μs) with 58% of packets in the range 420–440 μs. The 430 μs value can be reconducted to the 1-time slot packet maximum duration of 366 μs. Also in this case, the good concentration of measurements confirm the validity of the packet duration proposed feature.

Based on the packet length distribution, it is possible to define a recognition time as the period from sensing start to detection of the n^{th} Bluetooth packet. When N consecutive packets belonging to 1-slot OR 3-slot OR 5-slot classes are detected, the classifier raises a flag. The following cases were analyzed: N=10, N=25, N=50. Bandwidth was set to one the following values: 1, 5, 10, 25 *MHz* (first Bluetooth channel i.e. 1 *MHz* bandwidth, first 5 channels i.e. 5 *MHz*, and first 10 channels, i.e. 10 *MHz*) and was obtained by filtering the original sequence (corresponding to the whole bandwidth of 25 *MHz*). Figure 5 shows the resulting recognition time graph.

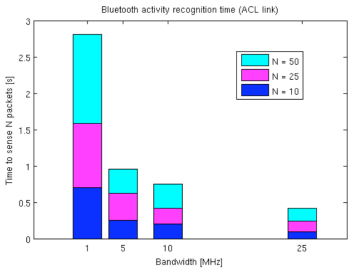


Figure 5 - Recognition time by varying N and bandwidth

As expected, the time to sense N packets grows as the considered bandwidth gets smaller. However from 25 *MHz* to 5 *MHz* this time remains small, compared to the one corresponding to 1 *MHz* bandwidth. Hence, a bandwidth of 5 *MHz* may be a good compromise between bandwidth and time to sense, when considering a FHSS technology such as Bluetooth. With only 5 *MHz* of bandwidth the packet length distribution becomes as shown in Figure 6. Note that the three peaks are preserved, even if slightly attenuated.

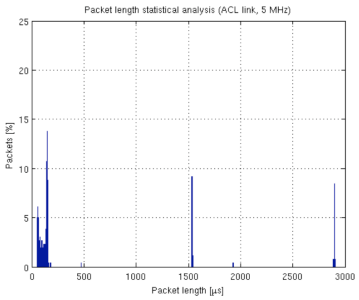


Figure 6 - Packet length statistics (ACL link, 5 *MHz*)

The increased number of occurrences of short length packet between 50 μs and the first peak can be interpreted as the effect of channels at bandwidth edge. The captured energy that falls in this portion of bandwidth determines low SNR packets close to detection threshold. In that case the fast fading can produce a multitude of threshold crossings, resulting in a huge amount of false positive short length packets. Since this happens however only around zero values, it does not affect conclusion.

Packet inter-arrival time period was defined as the difference between timestamps of two consecutive detected packets. Results for an ACL link are reported in Figure 7.

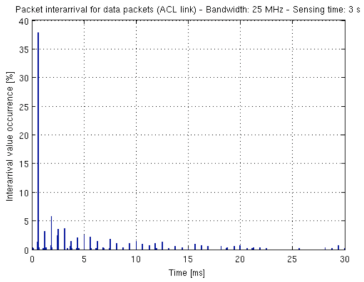


Figure 7 - Packet inter-arrival time (ACL link)

Even in this case a peak value stands out. This inter-arrival peak value is at 628 μs, closely resembling slot duration of 625 μs, [2] (there is only a difference of about 0.48% between these two values). The other peaks are extremely lower than the one at 628 μs. It is important to note that they are spaced of about one slot duration.

During voice transmission, the observed packet inter-arrival histogram is as in Figure 8. Note the regularities of the synchronous voice link that clearly arise.

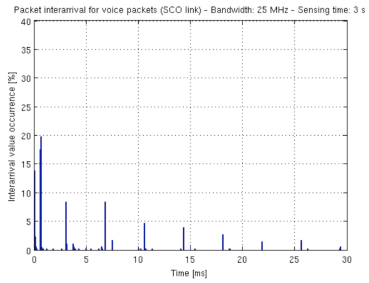


Figure 8 - Packet inter-arrival time (SCO link)

More precisely, the fundamental value of 625 μs can be recognized in the main peak at 643 μs. These two values differ of about 2.8%. Other peaks are present at 3070, 3770, 6860, 7490, 10600 μs, etc. This inter-arrival sequence reveals that the sensed communication was an HV3-based SCO link [2] with a T_{SCO} of 3750 μs (625+3125 μs). This behaviour is made clearer in Figure 9.

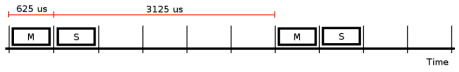


Figure 9 - Packet inter-arrivals in an HV3 packet exchange

Figure 9 shows slot occupation by Master and Slave HV3 packets. Naturally the energy detector cannot differentiate between Master and Slave packets. In this case the interesting value that can be extracted by an inter-arrival calculation is the time period between the last Slave voice packet and the following Master voice packet, that results in 3125 μs. This value differs from the observed second peak in Figure 8 (3070 μs) by only 1.76%.

Both Figures 7 and 8 reveal that the other proposed feature, i.e. packet inter-arrival period, is also valid, because of the presence of a well recognizable inter-arrival behaviour (i.e. the presence of peaks at specific time values) that is Bluetooth-specific.

VI. Discussion of Results and Future Directions

The AIR-AWARE Project aims at designing a black box capable to automatically detect and classify different radio technologies in the ISM band, using the output of an energy detector. This paper addresses the Bluetooth technology case.

Using the USRP, we computed the short-time energy diagram and the corresponding packet diagram; in this way we extracted Bluetooth packet timestamps and packet durations. Considering the Bluetooth MAC sub-layer Standard specifications, we proposed two technology-specific features: packet duration and packet inter-arrival period.

Reference Bluetooth communication scenarios were analyzed: data and voice links. Using this real traffic data we first calculated the distribution of packet lengths in both the data and voice link cases. In the data transfer case (ACL link) we found that, packet duration values were well concentrated at three values, that corresponded to the three main types of packets (1, 3 or 5 time slots). This behaviour is recognizable even with smaller bandwidths. Similarly, in the voice transmission case (SCO link) the histogram shows a single peak around which 58% of packets concentrate; this peak can be related to the duration of 1- slot packets.

As for the second proposed feature, i.e. packet inter-arrival period, we found in the histogram a prevalent peak, centred at the time slot duration value. There are also secondary peaks at values multiple of time slot duration. These secondary peaks are evident especially in the voice transmission case.

The proposed features seem to be valid for the purpose of this work, since they show a MAC sub-layer behaviour that is Bluetooth-specific, and that may permit its recognition in an heterogeneous networks scenario.

Further investigation should consider a testing case in the presence of other wireless technologies such as Wi-Fi and ZigBee, also against features proposed in [1] for the Wi-Fi case.

Acknowledgment

This work was supported in part by COST Action IC0902 "Cognitive Radio and Networking for Cooperative Coexistence of Heterogeneous Wireless Networks", funded by the European Science Foundation.

References

[1] M.-G. Di Benedetto, S. Boldrini, C.J. Martin Martin, and J. Roldan Diaz, *Automatic network recognition by feature extraction: a case study in the ISM band*, March 21, 2010 [*Proceedings of the 5th International Conference on Cognitive Radio Oriented Wireless Networks and Communications*, Special Session on Cognitive Radio and Networking for Cooperative Coexistence of Heterogeneous Wireless Networks, June 9-11, 2010, Cannes, France]

[2] IEEE Std 802.15.1 – 2005, IEEE Standard for Information technology – Telecommunications and information exchange between systems – Local and metropolitan area networks – Specific requirements – Part 15.1: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPANs), June 14, 2005

[3] Bluetooth SIG, *Specification of the Bluetooth system*, December 17, 2009

[4] S. M. Kay, *Fundamentals of Statistical Signal Processing. Vol. II: Detection Theory*, Prentice-Hall, Upper Saddle River, NJ; 1998

[5] Z. Ye, G. Memik, and J. Grosspietsch, "Energy Detection using Estimated Noise Variance for Spectrum Sensing in Cognitive Radio Networks", *WCNC 2008 proceedings*