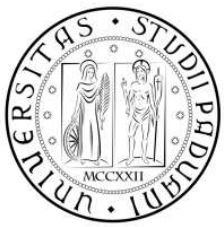


Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in Ingeneria delle Telecomunicazioni

tesi di laurea

Sistema embedded per la prototipazione di reti acustiche sottomarine

Relatore: Lorenzo Vangelista
Correlatore: Riccardo Masiero

Laureando: Ivano Calabrese

Padova, 23 Aprile 2012

Alla mia famiglia e
a Donatella.

Indice

Prefazione

Intorno agli anni 50 nacquero i primi calcolatori, i quali, oltre alle limitate prestazioni computazionali, erano fondamentalmente dispositivi stand-alone; cioè delle unità di calcolo a sè stanti, dove gli unici veicoli per il trasposto dell'informazione sono stati in un primo momento delle schede forate e, successivamente, una lunga evoluzione dei supporti magnetici. Nel 1961 L. Kleinrock[1] ha dimostrato che la commutazione di pacchetto era conveniente per le telecomunicazioni, tanto che da questa data in poi, oltre alla continua evoluzione tecnologica dei calcolatori (diventati poi nel corso degli anni Personal Computer), anche lo scenario che girava attorno ai dispositivi è cambiato; di fatto, partendo dai primi anni '70 con la dimostrazione pubblica di ARPAnet[2] e ALOHAnet[3] si è arrivati fino agli anni '90 in cui nasce il web identificato come la vera rivoluzione del network tra i Personal Computer; L'evoluzione tecnologica, poi, ha fatto in modo che buona parte dei dispositivi legati al'Information Tecnology venissero inseriti in un contesto wireless.

Si è scelto di fare questo piccolo excursus storico per capire, al meglio, quali sono le idee che stanno alla base di questa tesi e quali potrebbero essere gli sviluppi futuri, avendo una visione di insieme dello scenario delle reti sottomarine.

[1] Leonard Kleinrock (New York, 13 giugno 1934) è un informatico statunitense. I suoi più conosciuti e significativi lavori furono le prime ricerche sulla teoria delle code tra cui la commutazione a pacchetto; tecnologia che sta alla base dello standard ISO/OSI

[2] Advanced Research Projects Agency

[3] rete satellitare alle Hawaii.

Introduzione

Come già anticipato nella prefazione, l'accesso alle reti attuali utilizza la tecnologia wireless. Questo ha spinto diversi gruppi di ricerca a studiare e progettare sempre nuove applicazioni basandosi su questo tipo di connessione. In questo modo si è passati da reti costituite da dispositivi (come ad esempio: personal-computer, laptop e smart-phone), a reti di sensori capaci di comunicare tra loro in modo indipendente e prendere delle decisioni in funzioni di input ambientali.

L'evoluzione delle reti di sensori, hanno spinto alcune comunità di ricerca, come in questo caso, il laboratorio SIGNET[4] del Dipartimento di Ingegneria dell'Informazione dell'università di Padova, ad interessarsi alle così dette Reti Sottomarine. Di fatto al giorno d'oggi, c'è un grande interesse nel monitorare l'ambiente sottomarino dal momento che i due terzi della superficie terrestre è immersa in acqua.

Il progetto NAUTILUS (Network Architecture and protocols for Underwater Telerobotics via acustIc Links in Ubiquitous Sensing, monitoring and explorations) nato all'interno del laboratorio SIGNET e sovvenzionato dalla Italian Istitute of Technology (IIT) mira a fornire uno studio approfondito delle questioni tecniche, e di proporre una soluzione completa per l'architettura di rete e dei protocolli di comunicazione necessari per le tele-operazione di robot sottomarini (AUV/ROV)[5]. Nello svolgimento di questi obiettivi, dal momento che la necessità di implementare scenari realistici per la simulazione subacquea è risultata essere molto evidente, il contributo di questa tesi assume un aspetto importante nel momento in cui è necessario effettuare i test direttamente sul campo di utilizzo.

Di fatto, in questo momento, la prototipazione di nuovi protocolli per le comunicazioni sottomarine soffre molto della mancanza di un modello del canale subacqueo affidabile. A tal proposito, non potendo affidarsi a delle simulazioni di questo tipo, si è pensato di lavorare, per buona parte dello stack protocollare, in ambiente simulato (NS2) e, per la parte che riguarda il Livello Fisico[6] fare in

[4]Special Interest Group on NETworking

[5]Dall'inglese Autonomous Underwater Vehicle (AUV), è un robot che opera in acqua e che è in grado di portare a termine delle missioni in maniera autonoma. Si distinguono dai ROV, veicoli operati da remoto, dall'inglese Remotely Operated Vehicles, per il fatto che non hanno bisogno di essere collegati via cavo ad un pilota umano.

[6]il livello fisico corrisponde al primo livello dello stack ISO/OSI, detto anche Phisical Layer

modo che il simulatore si interfacci con i modem acustici e quindi che il canale corrisponda a un link reale.

Questo approccio, risulta essere vincente in quanto offre sicuramente un doppio vantaggio: la prototipazione dei protocolli citati sopra e, al tempo stesso, contribuisce alla stima del canale sottomarino.

Per l'appunto, l'obiettivo principale di questo elaborato è stato quello di fornire un dispositivo embedded capace di ospitare un Network Simulator, e allo stesso tempo interfacciare questo tool di sviluppo con i modem acustici a nostra disposizione. Le scelte di progetto si sono orientate, in prima istanza, verso l'utilizzo di una PandaBoard[7], come dispositivo embedded, e di NS2 come network simulator.

I passi che hanno portato alla stesura di questo elaborato sono:

- scelta del sistema operativo linux da installare sulla PandaBoard cercando un buon compromesso tra configurabilità e livelli prestazionali del software.
- customizzazione del sistema operativo secondo le necessità di progetto.
- installazione del Network Simulator (NS2) con i moduli NS-Miracle e DESERT
- prove di tipo evaluation-test in modo da fare un primo confronto tra Personal-computer e PandaBoard in termini di simulazione.
- analisi dello spazio occupato dal Network Simulator e tracking dell'utilizzo della RAM in fase di simulazione.
- customizzazione di NS2 in modo da avere un network simulator ottimizzato per la prototipazione di protocolli UnderWater.
- prove di simulazione effettuare direttamente in acque marine.

[7] questo dispositivo è un micro computer basato su una piccola scheda su cui è montato un processore Texas Instruments OMAP4430, classe ARM Cortex-A9

Capitolo 1

Propagazioni acustiche sottomarine

In questo capitolo verranno fornite delle nozioni inerenti le propagazioni acustiche sottomarine; partendo da una visione puramente storica si prosegue, nei successi paragrafi, con aspetti totalmente tecnici in cui verranno illustrati i dettagli sulla propagazione acustica sottomarina. Pertanto si parlerà del modello matematico del canale di comunicazione e infine si darà una visione panoramica dello scenario delle reti sottomarine.

1.1 Note storiche

In questa sezione si darà una visione del modo in cui il suono si propaga in acqua, un argomento che attualmente sta acquistando importanza nell'ambito delle telecomunicazioni. La conoscenza del fatto che il suono si propagasse sotto la superficie dell'acqua è dovuta a Leonardo Da Vinci; in effetti già nel 1490 scriveva che:portando una barca al largo e fermandola in mezzo al mare, inserendo in tubo in acqua in modo che sia immerso quasi del tutto, si nota che, appoggiando l'orecchio al capo non immerso, si può avvertire la presenza delle altre barche in navigazione. Questo primo esperimento di *sonar passivo* non dava però nessuna indicazione sulla direzione del target e soffriva di un scarso accoppiamento acustico tra aria e acqua; in ogni modo un sistema molto simile, a quello ideato da Leonardo Da Vinci, fu usato durante la prima guerra mondiale; di fatto con l'aggiunta di un secondo tubo si rese possibile all'ascoltatore la capacità di calcolare la posizione del Target. Dall'idea di Leonardo, si sono susseguite un grande numero di scoperte che hanno contribuito, più o meno direttamente, allo sviluppo dei sistemi acustici sottomarini. La prima accurata misurazione della velocità del suono in acqua è dovuta a *Daniel Collandon*, un fisico svizzero, e *Charles Sturm* un matematico francese. Nel 1827 analizzarono, infatti, la velocità del suono misurando la differenza tra il tempo di arrivo di un flash di luce e il colpo dato ad una campana sott'acqua. Altre scoperte come la *magnetostrizione* ad opera

di *James Joule* nel 1840, e della piezoelettricità da parte di *Pierre e Jacques Curie* nel 1880 hanno gettato le basi per la costruzione dei moderni trasduttori elettro-acustici e idrofoni. La prima applicazione pratica del suono sottomarino risale, tuttavia, alle fine del XVIII secolo quando fu inventato un sistema sottomarino per la misurazione delle distanze basato su campane: dalla valutazione del tempo, che intercorre tra il suono di una campana e il suono di una corno suonato simultaneamente, una nave potrebbe determinare la sua distanza da un'altra nave o da una piattaforma dove entrambi le sorgenti dei suoni sono installate. In 1912, Richardson ha archiviato un brevetto con la *British Patent Office* nel quale veniva usato il suono aereo per la misurazione delle distanze mediante effetto eco; qualche mese più tardi, lo stesso Richardson, deposita un altro brevetto per la controparte sottomarina del suo sistema, risultato essere poi il predecessore dei moderni sonar. Sfortunatamente la nave passeggero "Titanic" non ha potuto beneficiare di questa tecnologia in quanto quest'ultima collise con un iceberg cinque giorni antecedenti la prima deposizione dell'applicazione del brevetto. Il primo ad effettuare una implementazione dell'idea di Richardson fu Fessenden negli Stati Uniti, il quale inventò un simile sistema ma che attualmente lavora a frequenze acustiche più basse di quelle previste da Richardson. Prima del 1914, Fessenden dimostrò che il suo sonar poteva identificare un iceberg ad una distanza di due miglia e questo aspetto favorì l'installazione di dei suoi sonar sui sottomarini americani durante la prima guerra mondiale. Lo scoppio della guerra diede un grande impulso alla ricerca sull'acustica sottomarina tanto che, usando sempre l'effetto eco dei suoni, si riuscì ad identificare una lastra di acciaio a più di un chilometro di distanza. Detto ciò si tiene a precisare che negli anni antecedenti alla seconda guerra mondiale era già iniziata la produzione, in larga scala, di equipaggiamenti sonar relativamente economici e di sottomarini dotati di questa tecnologia. Tuttavia, una delle più rilevanti scoperte nell'ambito dell'acustica sottomarina negli anni tra le due guerre mondiali; infatti alcuni esperimenti marini sulla misurazione di distanze mostrarono che il canale trasmissivo era affetto da variazione inaspettate delle sue caratteristiche; di fatto si analizzò spesso che ottimi eco si ottenevano durante la mattina piuttosto che nel pomeriggio. Questo afternoon effect (effetto pomeriggio) fu inteso come derivante dallo slittamento del gradiente di temperature che era in grado rifrangere le onde sonore curvando la loro traiettoria e creando zone d'ombra in cui i segnali acustici non venivano propagati. La rivelazione di un aspetto così peculiare della propagazione subacquea fu un punto di svolta per altre scoperte in questo campo. Dalla fine delle guerre, lo sfruttamento del suono sottomarino nelle applicazioni civili diventò diffuso e molti strumenti per la navigazione, telemetria e comunicazioni sono tutt'ora oggi commercialmente disponibili.

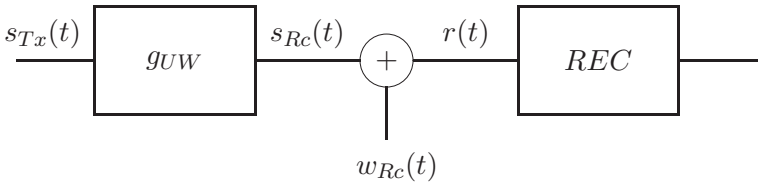


Figura 1.1: Moto uniformemente accelerato

1.2 Modello del canale

L'uso delle onde a pressione acustica per le comunicazioni digitali sottomarine pone un numero elevato di sfide. La prima difficoltà deriva della velocità di propagazione delle onde sonore, approssimativamente pari a 1500m/s, risultante essere cinque ordini di grandezza più piccolo della velocità delle onde radio in aria. Attualmente si considera che la velocità dipenda dalla profondità (e quindi dalla pressione), della temperatura e dal livello locale di salinità. Inoltre i fenomeni di attenuazione sonora sono differenti da quelli comunemente considerati per le onde radio; infatti le onde acustiche dipendono fortemente dalla frequenza del segnale trasmesso, tanto che segnali a frequenza più bassa sono meno attenuati e per questo motivo possono ricoprire distanze più ampie. Inoltre, fondali bassi contribuisco a costituire un guida d'onda tanto che il suono si propaga in una geometria cilindrica piuttosto che in una sferica in scenari del genere. Una terza differenza macroscopica, tra trasmissioni subacquee e aeree, sta nelle fonti di rumore che disturbano le comunicazioni, la cui *PSD*(Power Spectral Density) è dipendente dalla frequenza: in altre parole il rumore del canale è colorato. Infine, i complicati pattern di rifrazione del suono in acqua causano dei fenomeni multipath molto forti e il rilevante effetto doppler originato dal costante movimento dei sistemi di trasmissione e ricezione, dovuto ad esempio a correnti marine, cambia in funzione dalla frequenza portante usata per trasmissione. Queste sfide hanno incoraggiato a fornire una moltitudine di contributi al campo delle elaborazioni del segnale, tanto che hanno permesso di avere, tra due nodi, dei collegamenti di comunicazioni con un grado di affidabilità accettabile; in questo modo è stato dato un primo impulso alla ricerca nel campo delle reti acustiche sottomarine. In questa sezione verranno illustrate le equazioni fondamentali che permettono di approssimare, in forma matematica, un canale trasmissivo sottomarino.

Il segnale trasmesso è una forma d'onda acustica $s_{Tx}(t)$, caratterizzata da una potenza statistica $M_{s_{Tx}} = E[s^2_{Tx}(t)]$ rappresentata come una pressione, misurata in Pascal(Pa). In molti casi pratici il segnale trasmesso è assunto essere un tono

acustico sinusoidale di frequenza f_0 e il collegamento sottomarino avere una lunghezza d. Per convertire la potenza acustica in potenza elettrica si usa la seguente relazione:

$$(P_{Tx})_{dBm} = (M_{s_{Tx}})_{dB\mu Pa} - 17.2 - 10\log_{10}(\eta) \tag{1.1}$$

dove -17.2 è il fattore di conversione dalla potenza acustica $dB\mu Pa$ alla potenza elettrica, e η è l'efficienza generica dei circuiti elettronici(come amplificatori di potenza e trasduttori). Il valore tipico per η è 0.25. L'attenuazione $A_{ch}(d, f)$ in cui incorre un tono di frequenza f_0 può essere espresso come il prodotto di due componenti

$$a_{Ch}(d, f_0) = d^k \cdot [\alpha(f_0)]^d \tag{1.2}$$

in cui il termine d^k è chiamato *perdita di dispersione* e dipende solo dalla geometria della propagazione, mentre il termine $[\alpha(f_0)]^d$ è chiamato *perdita di assorbimento*. Il fattore k chiamato *coefficiente di dispersione* è utilizzato per definire la geometria della propagazione (k= 1 cilindrica, K=2 sferica e k= 1.5 è un valore pratico per rappresentare una condizione di propagazione intermedia)[1]. Il coefficiente di assorbimento $\alpha(f)$ è stato estratto empiricamente da William H. Thorp e può essere espresso dalla seguente relazione:

$$10\log_{10}\alpha(f) = 0.11\frac{f^2}{1+f^2} + 44\frac{f^2}{4100+f^2} + 2.75\cdot 10^{-4}f^2 + 0.003 \tag{1.3}$$

che restituisce un valore di $\alpha(f)$ in dB/Km per f in KHz. In Figura 1.2 viene riportato il grafico di $\alpha(f)$ in funzione di un ampio range di frequenze. Il coefficiente di assorbimento è il fattore più importante che limita la potenza ricevuta ad un a distanza prefissata dal momento che cresce rapidamente all'aumentare della frequenza. Si noti che la 1.2 descrive l'attenuazione di un percorso di propagazione libero da ostacoli; inoltre, se un tono a frequenza f_0 e potenza $M_{s_{Tx}}$ è trasmesso su una distanza d, la potenza del segnale ricevuto sarà $M_{s_{Tx}}/A_{Ch}(d, f)$.

Il rumore effettivo $w_{Ch}(t)$ nell'oceano può essere modellato come la sovrapposizione di 4 componenti; *turbolenza* (t), *navigazione* (s), *onde* (w), e *rumore termico* (th) all'ingresso del ricevitore. Si assume che ogni componente abbia una distribuzione gaussiana e che sia descritta da un PSD costante. Per ogni componente di rumore la seguente relazione riporta i corrispondente PSD in $dB\mu Pa/Hz$,

[1] In realtà questa ultima conclusione, anche se riportata in alcuni paper non è del tutto veritiera

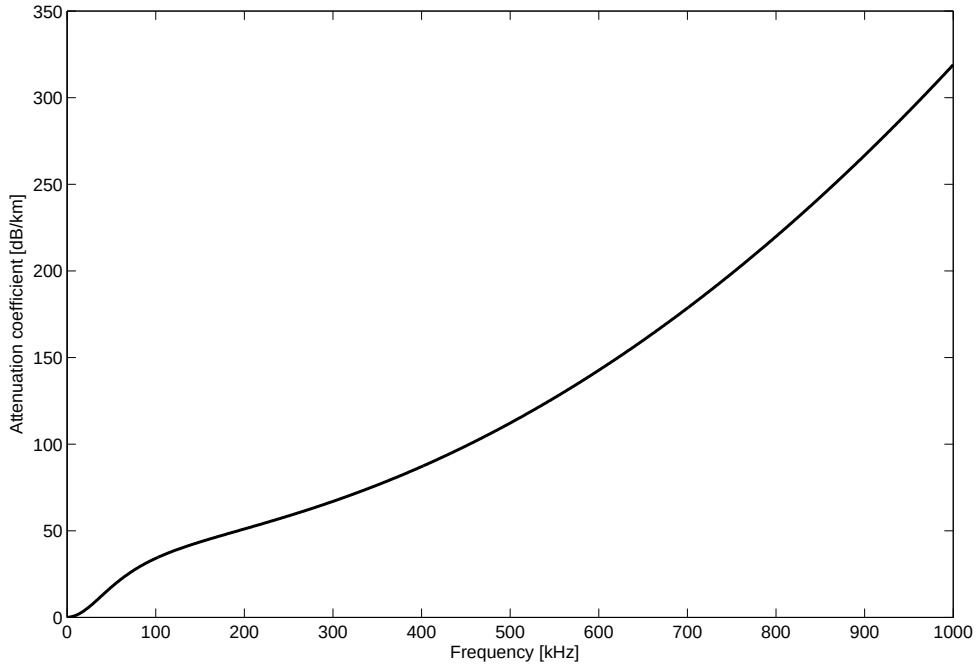


Figura 1.2: Andamento del coefficiente di assorbimento $\alpha(f)$

$$10\log_{10}\mathcal{P}_{w_t} = 17 - 30\log_{10}(f) \quad (1.4)$$

$$10\log_{10}\mathcal{P}_{w_s} = 40 - 20(S-0.5) + 26\log_f - 60\log_{10}(f+0.3) \quad (1.5)$$

$$10\log_{10}\mathcal{P}_{w_w} = 50 + 7.5\sqrt{W} + 20\log_{10}(f) - 40\log_{10}(f+0.4) \quad (1.6)$$

$$10\log_{10}\mathcal{P}_{w_{th}} = -15 + 20\log_{10}(f) \quad (1.7)$$

In questa espressione S è il cosiddetto *fattore di navigazione* che rappresenta l'intensità delle attività di navigazione sulla superficie dell'acqua, e assume valori tra 0 e 1, dove W è la velocità del vento in m/s. Il rumore totale in PSD infine è espresso come

$$\mathcal{P}_{w_{Rc}}(f) = \mathcal{P}_{w_t}(f) + \mathcal{P}_{w_s}(f) + \mathcal{P}_{w_w}(f) + \mathcal{P}_{w_{th}}(f)$$

In Figura 1.3 viene riportato il grafico $\mathcal{P}_{w_{Rc}}(f)$ per differenti valori del fattore S (S=0, 0.5 , 1) e della velocità del vento (W= 0, 10m/s). Da questa figura si osserva che ogni componente influenza il PSD a frequenze differenti.

In particolare la turbolenza influenza la regione di frequenze molto basse $f < 10Hz$. Il rumore di navigazione ha un'importanza maggiore nella regione di frequenze tra 10 e 100Hz, mentre il vento influenza la porzione di frequenze

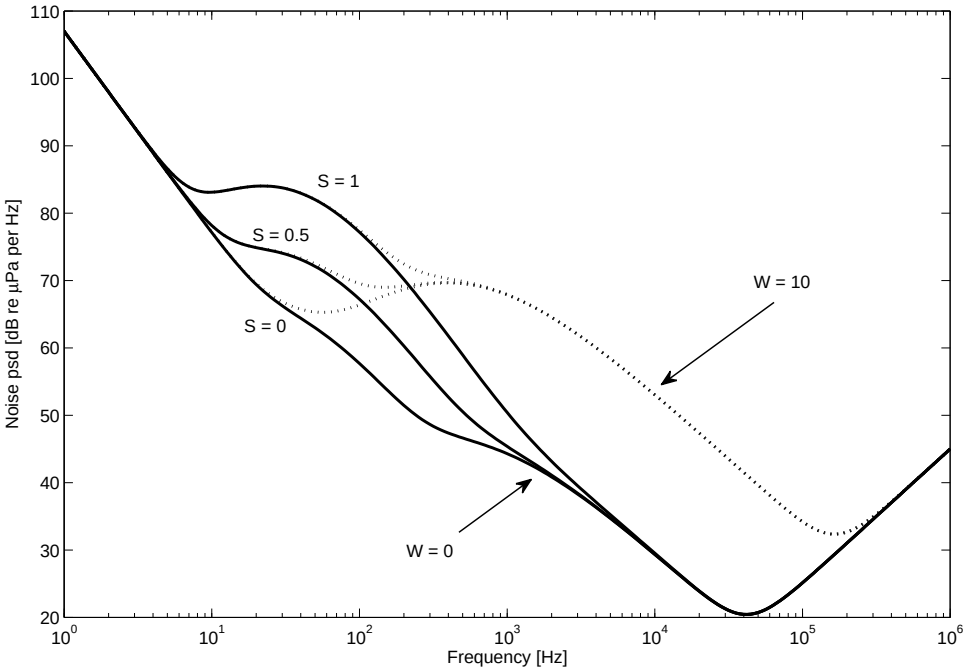


Figura 1.3: Power Spectral Density in funzione della frenquenza della portante per differenti valori del fattore di navigazione S e differenti velocità del vento W

da $100Hz$ a $100KHz$. In fine il rumore termico è la componente dominante per $f > 100KHz$. Dall'uso delle equazioni inerenti l'attenuazione e il rumore, si può estrapolare il rapporto segnale-rumore (SNR) Γ associato ad un tono di frequenza f_0 e potenza $M_{s_{TX}}$ ricevuto ad una distanza d dalla sorgente (si assume una larghezza di banda B):

$$\Gamma(d, f_0) = \frac{M_{s_{TX}}}{\mathcal{P}_{w_{Rc}}(f_0) 2B a_{Ch}(d, f_0)} \tag{1.8}$$

in cui il rumore PSD $\mathcal{P}_{w_{Rc}}(f)$ è assunto costante all'interno della larghezza di banda B attorno ad f_0. Nell'equazione appena citata il fattore $1/[a_{Ch}(d, f_0)]$ è il termine dipendente dalla frequenza. Dal momento che $a_{Ch}(f)$ aumenta con il crescere della frequenza mentre, $\mathcal{P}_{w_{Rc}}(f)$ diminuisce (fino ad un certo punto), il prodotto tra i due ha un massimo per una qualche frequenza ottima. Questo massimo garantisce un effetto di minima combinazione tra attenuazione e rumore, per cui è la migliore frequenza da usare per le trasmissioni. Per raffigurare meglio questo fenomeno è possibile analizzare la Figura 1.4 in cui vengono mostrate delle linee concave grige che rappresentano il fattore a_{Ch} come una funzione della frequenza con d che varia tra 10m a 100Km.

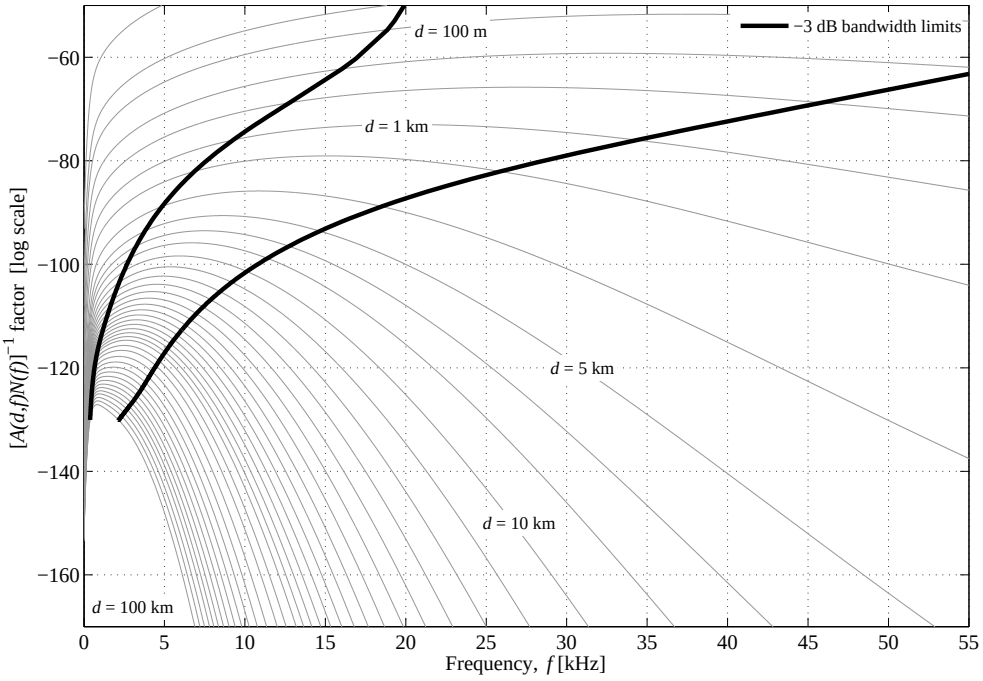


Figura 1.4: Questa figura si riferisce alla parte di $\Gamma(d, f_0)$ dipendente dalla frequenza per la trasmissione sottomarina di un tono acustico a frequenza f_0. Le curve di colore grigio rappresentano l'andamento del fattore $[\mathcal{P}_{w_{Rc}}(f)a_{Ch}(d, f)]^{-1}$ per differenti distanze; le curve di colore nero, invece, rappresentano il limite superiore e inferiore della banda di frequenze in cui è possibile trasmettere

La prima osservazione è che ogni curva esibisce un massimo, mostrando l'esistenza di una frequenza ottimale $f_{opt}(d)$ da usare per la trasmissione per ogni distanza di collegamento. Si nota che la $f_{opt}(d)$ diminuisce all'aumentare delle distanze spiegando il motivo per cui il sistema di eco ranging, basato su basse frequenze di Fessenden, funzionava meglio di quello di Richardson. Un'altra importante osservazione può essere fatta definendo la larghezza di banda acustica $B_{Ch}(d)$ ai meno 3dB attorno ad $f_{opt}(d)$ in accordo con il valore massimo di Γ. In Figura 1.4 vengono graficate due linee nere che delimitano la banda ottima del canale. Si nota, inoltre, che la banda disponibile diminuisce all'aumentare della distanza.

1.3 Dettagli sulla propagazione sottomarina

Mentre le precedenti equazioni permettono di definire un canale in maniera grossolana, la propagazione sottomarina è più complessa di quanto espresso dal modello finora considerato. Non è l'obiettivo di questa sezione entrare nei dettagli di questo argomento, ma vale la pena menzionare come le principali caratteristiche fisiche dell'acqua influenzino la propagazione di onde a pressione acustica. Si ricordi che la velocità del suono è pari, indicativamente, a 1500 m/s ma in effetti dipende dalla temperatura, profondità e dal grado di salinità dell'acqua.

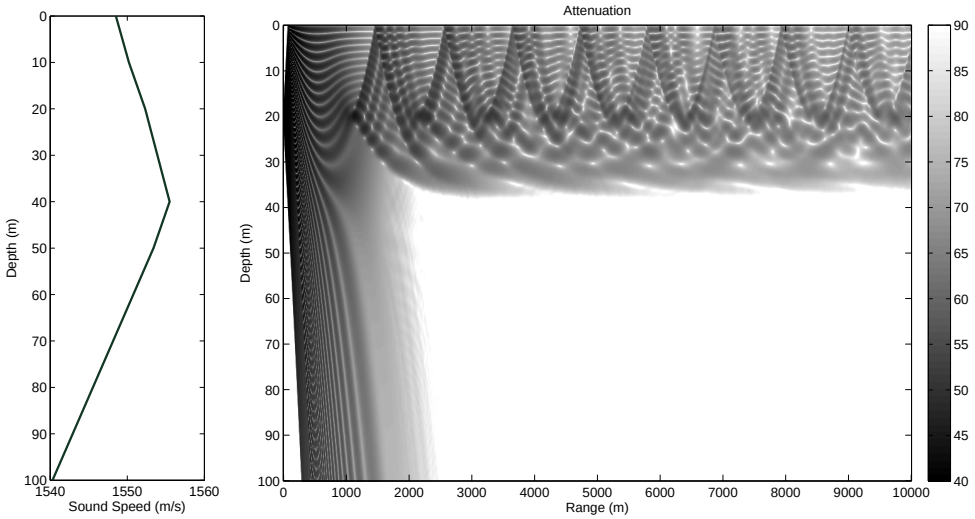


Figura 1.5: Canale trasmissivo in fondali bassi: a sinistra viene riportata il profilo della velocità del suono in funzione della profondità; a destra invece è illustrato il profilo di attenuazione del suono, si può notare la presenza di una zona d'ombra dai 2500m in poi dalla sorgente e una profondità di maggiore o uguale a 38m

Un tipico approccio consiste nel valutare queste tre quantità al variare della profondità, e poi valutare la velocità del suono in funzione della stessa. Questa funzione è chiamata *profilo della velocità del suono* (dall'inglese: Sound Speed Profile (SSP)). L'SSP influenza il modo in cui le onde sono rifratte mentre si propagano ed è un fattore di primaria importanza. Differenti SSP implicano diversi profili di propagazione, che possono in caso, dare origine a strani fenomeni come l'effetto pomeriggio descritto dagli esperimenti di Stephenson, in base al quale l'intensità del segnale acustico ricevuto varia in maniera significativa durante le ore del giorno.

Le Figure 1.5 e 1.6 fanno riferimento rispettivamente, ad uno scenario di bassa e alta profondità. Ciascuna immagine contiene sulla sinistra un grafico

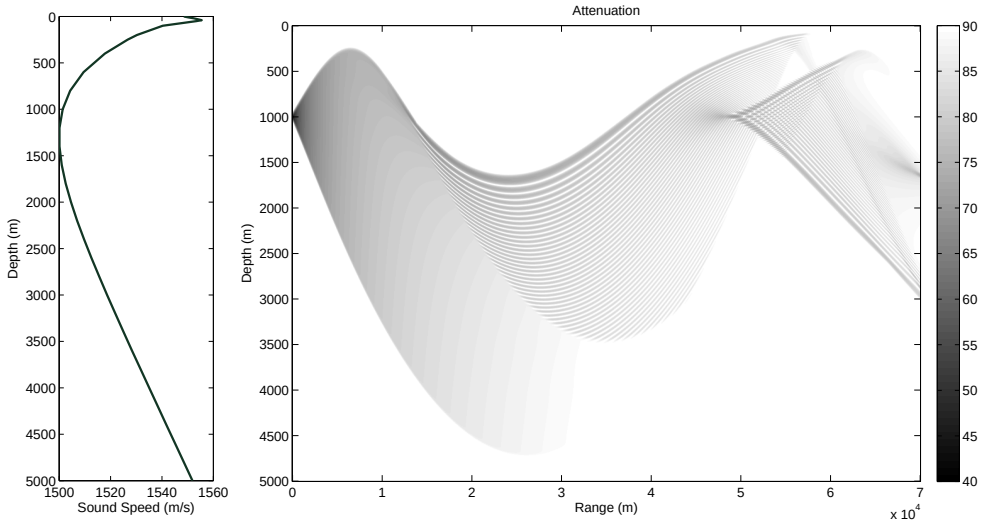


Figura 1.6: Canale trasmissivo in fondali profondi: a sinistra viene riportata il profilo della velocità del suono in funzione della profondità; a destra invece è illustrato il profilo di attenuazione del suono, si può notare la presenza di una zona di convergenza a 50000m e una profondità di 1000m

del profilo della velocità del suono in cui le ordinate rappresentano la profondità e le ascisse la corrispondente velocità del suono. Invece sul lato destro si rappresenta, con una scala di grigi, l'attenuazione in decibel subita dal segnale trasmesso come una funzione della distanza della trasmittente (sulle ascisse) e della profondità (sulle ordinate). Quindi un grigio più scuro rappresenta un'attenuazione più bassa: le porzioni basse sono perciò regioni in cui nessun segnale è effettivamente ricevuto. Si noti che la trasmittente è posta ad una profondità di 20m in Figura 1.5 e di 1000m in Figura 1.6, ed emette suoni verso destra. La Figura 1.5 evidenzia un fenomeno noto come zona d'ombra , e a causa del modo in cui il suono è rifratto nessun segnale raggiunge l'area sotto i 34m di profondità, dopo una distanza di 2500m. Il pattern di propagazione è inoltre complicato da riflessioni di superficie, che contribuiscono a generare copie multiple del segnale trasmesso, che si sovrappongono in maniera diversa in differenti punti dello spazio causando uno scarto dell'attenuazione rispetto al valore medio fornito dall'equazione (1.2). La Figura 1.5 riporta un esempio di propagazione del suono in acque profonde e mostra un fenomeno noto come *Deep Sound Axis*. In questa figura, SSP rappresenta la condizione dell'acqua in primavera a circa 43° nord, in un punto a metà strada tra Newfoundland e la Gran Bretagna. In questo caso si osserva, che dopo essere state trasmesse, lo onde sonore vengono intrappolate nell'acqua e non raggiungono più la superficie. Inoltre si osserva la presenza di una zona di convergenza ad una distanza di 50 mila metri e ad una

profondità di 1000, una zona dove differenti percorsi di propagazione convergono e provocano localmente una potenza del segnale maggiore. Le figure 1.5 e 1.6 danno un'idea di quanto sia complessa la propagazione sottomarina: infatti la più accurata riproduzione della propagazione del suono può essere ottenuta solo risolvendo l'equazione d'onda attraverso il tracciamento di curve come fatto in figura. Oltretutto il problema di sommare tutti gli effetti di propagazione in fenomeni statistici, come il *Fading*, è ancora un problema aperto. Infatti un gran numero di fattori influenza il modo in cui il suono si propaga per il quale non esiste attualmente un modello statistico esaustivo. In ogni caso le equazioni di un link generico riportate precedentemente possono aiutare a capire almeno il comportamento generico delle comunicazioni sottomarine sono state usate in molti paper riguardanti le reti sottomarine.

1.4 Panoramica sulle reti sottomarine

In questo primo capitolo si è partiti con un paragrafo sulla storia delle trasmissioni subacquee. Col passare degli anni gli scenari sono cambiati ma l'idea di fondo, ciò quella dell'interazione tra diversi entità non è mai svanita; di fatto si è passati dalla d'interazione diretta in piccola scala a interazioni a distanza in vastissima scala (basti pensare quanta informazione ogni secondo si muove all'interno di una rete internet), in parallelo a questo si è aggiunta la necessità di conoscere il mondo che ci circonda e quindi da singole analisi territoriali si è passati ad una rete di sensori distribuiti in modo da avere una visione più dettagliata dei target prefissati e con risposte pressocchè immediate. Tutto questo evolversi della tecnologia, ha spinto, in certo qual modo, alcune comunità di ricerca ad ideare reti di sensori in un'ottica sottomarina.

Volendo fornire una panoramica sulle reti sottomarine si può iniziare ad elencare quelli che potrebbero essere le entità sottomarine che possono costituire una scenario di questo tipo; per tanto dalla Figura 1.7 si possono individuare elementi come:

Sensori sottomarini: questi dispositivi hanno il compito di monitorare e comunicare i dati rilevati nell'ambiente subacqueo; per questo scopo sono costituiti da un modem acustico e da sensori per il rilevamento di grandezze prefissate (come ad esempio salinità dell'acqua, temperatura, pressione, etc.). Dispositivi del genere vendono installati e mantengono una posizione fissa all'interno dello scenario underwater.

Veicoli sottomarini autonomi (AUV): questi sistemi, di natura più sofisticata dei precedenti, hanno la possibilità di spostarsi e quindi ricoprire la funzione di un sensore mobile. L'utilità di questi dispositivi è sicuramente quella di poter gestire ad esempio un monitoraggio seguendo una certo percorso prefissato.

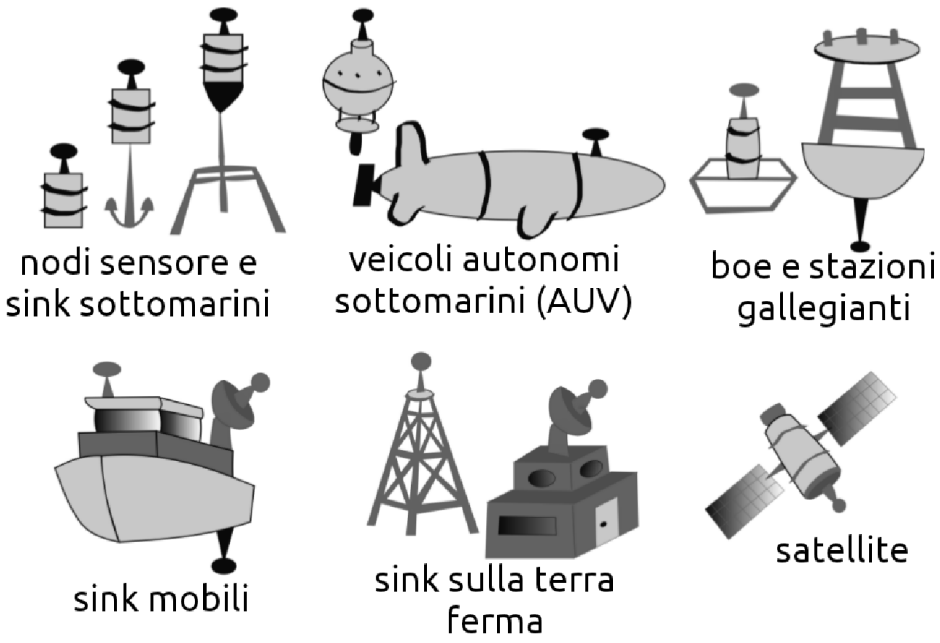


Figura 1.7: In questa immagine sono presenti in grendi linee le entità fondamentali che costituiscono una rete di comunicazione sottomarina

Sink sottomarini: il compito dei sink è di fondamentale importanza se si pensa a una rete in larga scala; di fatto questi elementi hanno il compito di raccogliere le informazioni dei nodi sensori per poi ritrasmetterle ai nodi di superficie.

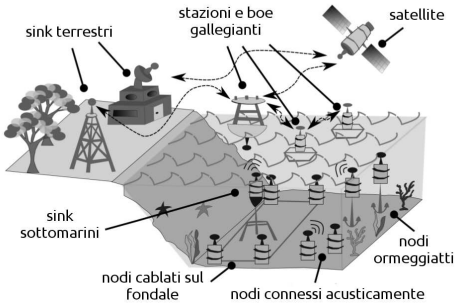
Boe e stazioni galleggianti questi elementi hanno il compito di interfacciare comunicazione acustiche sottomarine con comunicazioni via etere; in questo modo se la rete di sensori è delocalizzata rispetto alla stazione base è possibile far recapitare con un bit rate più elevato i dati raccolti ad un centro di calcolo posizionato sulla terra ferma.

Sink gallegianti mobili: questi punti di raccolta hanno le stesse caratteristiche di quanto visto prima, anche se in aggiunta hanno la possibilità di spostarsi e quindi ricoprire aree più ampie. Inoltre essendo galleggianti hanno anche la possibilità di comunicare direttamente, via etere, con altri dispositivi di superficie.

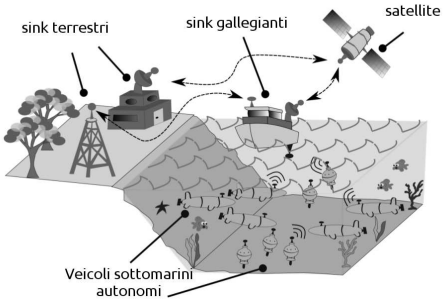
Sink installati sulla terra ferma: i sink installati a terra sono sostanzialmente dei dispositivi di supporto in quanto gestirebbero eventualmente solo comunicazioni terrestri e satellitari

Satelliti: infine si hanno i satelliti i quali oltre ad dare un supporto come ponte radio, possono fornire ai dispositivi che lo richiedono (ad esempio per la localizzazione degli AUV) la loro posizione geografica.

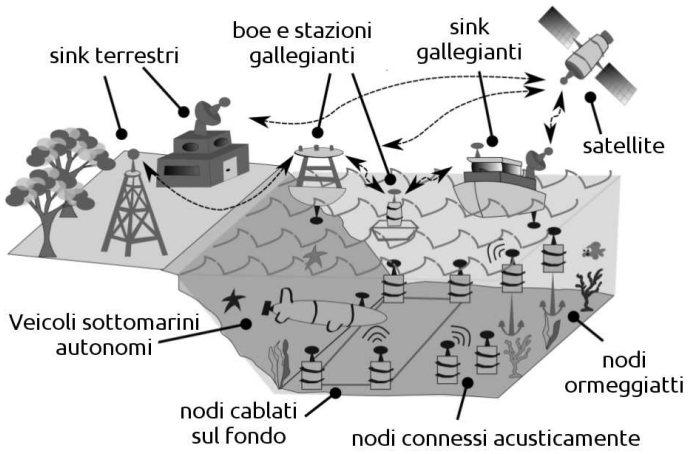
Volendo illustrare uno scenario prototipo si può fare attraverso le figure:



(a) architettura statica di uno scenario sottomarino



(b) architettura mobile di uno scenario sottomarino



(c) architettura ibrida di uno scenario sottomarino

Figura 1.8: Queste immagini servono per dare un'idea di come può presentarsi uno scenario sottomarino .

Capitolo 2

Simulatori di rete

Nel presente capitolo saranno discussi concetti riguardanti il software e le librerie utilizzate per eseguire le simulazioni di reti sottomarine; pertanto verrà fatta una panoramica del *simulatore di rete* **NS2**, delle librerie *NS-Miracle* sviluppate all'interno del laboratorio SIGNET[1] e dei *moduli DESERT* idealizzati e sviluppati per il progetto Nautilus sempre all'interno del medesimo laboratorio.

2.1 Introduzione a NS2

NS2 è un simulatore di reti basato su eventi discreti. Questo simulatore fornisce un supporto sostanziale per la simulazione del protocollo TCP, routing e protocolli multicast per reti sia cablate che wireless a tutte le comunità di ricerca incentrate sul networking. NS è nato come variante del simulatore di reti REAL nel 1989; è un strumento originariamente pensato per studiare le dinamiche di flusso e gli schemi di controllo nelle reti dati a commutazione di pacchetto. Nel 1995 lo sviluppo di NS è stato supportato finanziariamente dal DARPA attraverso il progetto VINT[2] nei laboratori Xerox PARC, LBL, UC Berkeley e USC/ISI. Al giorno d'oggi, invece, lo sviluppo di questo simulatore è affidato alla DARPA col progetto SAMAN e dal NSF con progetto CONSER, anche se entrambi sono in collaborazione con altri ricercatori tra cui ACIRI[3]; in definitiva si può asserire che il punto di forza di questo simulatore è sicuramente da attribuire al fatto che diverse comunità di ricerca, nei diversi anni, hanno contribuito a mantenere vivo il suo sviluppo; come ad esempio l'inclusione del modulo `dei80211mr` all'interno delle release ufficiali di NS2.

Il simulatore NS2 per descrivere la topologia delle reti, gli elementi coinvolti e le loro caratteristiche si affida fondamentalmente al linguaggio di program-

[1] Il laboratorio SIGNET è un gruppo si ricerca e sviluppo costituito all'interno del dipartimento di Ingegneria dell'informazione dell'università degli studi di Padova

[2] Virtual InterNetwork Testbed

[3] AT&T Center for Internet Research at ICISI

mazione C++ e agli script OTcl. Il linguaggio C++ ha un profilo back-end in quanto si occupa

- della definizione dei nuovi agent, dei protocolli e dei framework
- della manipolazione dei pacchetti a livello di byte/bit
- e della gestione di strutture dati complesse o voluminose

Per quanto riguarda, invece, lo script OTcl (Object-Oriented Tool Command Language) possiamo dire che, a differenza del linguaggio C++, ha un profilo front-end; di fatto se lo scopo della simulazione è quello di definire una determinata topologia di una rete, utilizzando elementi standard dello stack protocollare, l'unica cosa che il programmatore deve gestire è la scrittura di uno script OTcl. Anche in questo caso, possiamo schematizzare tre punti caratteristici dello script OTcl; e quindi affermare che:

- descrive lo scenario di simulazione
- definisce le variabili temporanee
- definisce il codice che necessita di essere modificato frequentemente

Volendo fornire uno schema a blocchi per rappresentare in modo semplice quanto descritto si può fare riferimento alla Figura 2.1

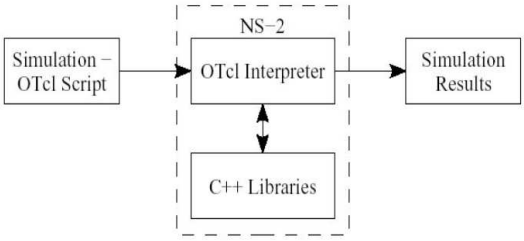


Figura 2.1: Questa figura ha il compito di illustrare in modo semplificato la struttura di NS2 dal punto di vista dei due linguaggi di programmazione

2.2 NS-Miracle

NS-Miracle è un'estensione per il simulatore **NS2** sviluppato dal Dipartimento di Ingegneria dell'Informazione dell'Università di Padova. L'acronimo sta per *Multi InteRfAce Cross Layer Extension* e descrive una delle sue funzionalità più importanti introdotta dalla modularità di questa patch. Di fatto, usando ns-miracle si possono gestire i nodi in modo cross layer in aggiunta al supporto multi-tecnology[4] nello stesso nodo.

[4]Il supporto multi-tecnology permette al generico nodo di gestire ad esempio più interfacce di rete; questa caratteristica non è supportata da NS2 puro

In NS-Miracle ogni modifica (o funzionalità) può essere descritta da una libreria a sé stante; questo aspetto molto importante permette al programmatore di integrare tale modifica all'interno di una simulazione usando semplicemente il comando di caricamento nello script TCL[5]. In questo modo le modifiche apportate ai moduli possono essere intercambiate caricando la libreria appropriata. Infine, altra caratteristica, da non sottovalutare, è che le modifiche apportate ad una libreria richiedono solo di ricompilare il codice associato a quest'ultima; in questo modo, trattandosi di operazioni poco dispendiose in ordine di tempo, si ha la possibilità di dedicare maggiori risorse al debugging e alle simulazioni.

La motivazione che ha spinto il laboratorio SIGNET a creare l'estensione NS-Miracle è stata quella di colmare il gap tra NS2 e la simulazione di ambienti a multi-tecnologia, e di implementare, come anticipato prima, la gestione della comunicazione tra layer. In questo modo, NS-Miracle fornisce un approccio generico e indipendente delle interfacce di rete ad uno dei più avanzati campi di ricerca, rendendolo quindi un valido strumento di simulazione nel campo open-source.

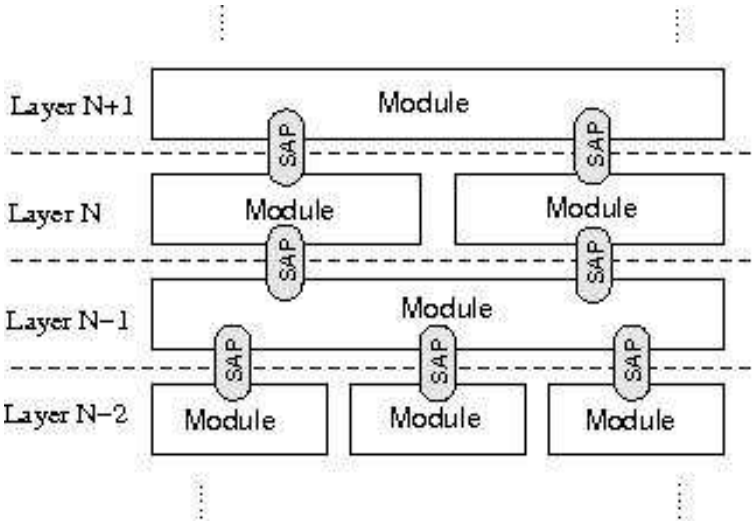


Figura 2.2: In questa figura vengono rappresentati i moduli nei diversi layer; si può notare come ogni layer può ospitare più moduli e come i moduli riescono a cominicare attraversi il SAP(service access point)

Tornando all'aspetto implementativo, un nodo in Ns-Miracle è basato su generiche entità connesse tra di loro: lo stack protocollare è implementato attraverso blocchi chiamati *moduli*, come mostrato in Figura 2.2. L'idea innovativa, inoltre, consiste nel permettere la presenza di più moduli all'interno di un singolo layer; questo aspetto da' la possibilità di ottimizzare le risorse e di

[5]La possibilità di caricare dinamicamente le librerie è stata integrata nel simulatore NS2 a partire dalla versione 2.33.

garantire un'organizzazione delle funzionalità molto più chiara (sia in fase di implementazione che di debugging). Come si vede dalla Figura 2.3 ogni modulo è connesso ad un'altra entità chiamata Core-Node il quale prescinde dallo schema in cui sono ordinati i layer e ha il compito di coordinare lo scambio di messaggi tra i moduli; inoltre conserva l'attuale posizione del nodo, nel campo di simulazione, in maniera da permettere calcoli in funzione della posizione come quelli relativi a modelli di propagazione e modelli di interferenza.

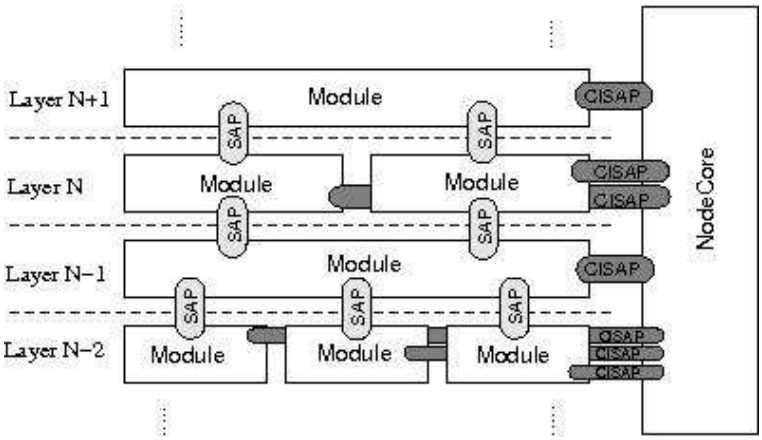


Figura 2.3: In questa figura, rispetto alla 2.2 abbiamo l'aggiunta dell'entità PlugIng; come descritto nel testo questo blocco garantisce l'interconumicazione tra i moduli

In aggiunta a quanto detto sopra, ci sono altre entità connesse al Core-Node, chiamate PlugIn; come mostrato in Figura 2.4) si può notare che anche questi ultimi non dipendono dalla classificazione dei vari layer. A causa della loro indipendenza dallo stack ISO/OSI, i PlugIn possono essere sfruttati per funzionalità di coordinamento dei nodi e per un'intelligente comunicazione tra layer.

La comunicazione tra differenti layer è fornita dal *Service Access Points* (SAP), in accordo con la struttura ISO/OSI (cioè ogni layer comunica solo con i relativi adiacenti). Come in NS2, le connessioni tra moduli, Node-Core e Plugin sono eseguite dai connettori; ma questi ultimi, con l'introduzione di NS-Miracle, sono stati completamente riprogettati in modo da introdurre il tracciamento dei pacchetti scambiati tra i layer: questa ultima funzionalità è a discrezione del programmatore.

2.3 DESERT Underwater

DESERT Underwater (DEsign, Simulate, Emulate and Realize Test-beds for Underwater network protocols) è un set completo di librerie pubbliche **C**/**C**++

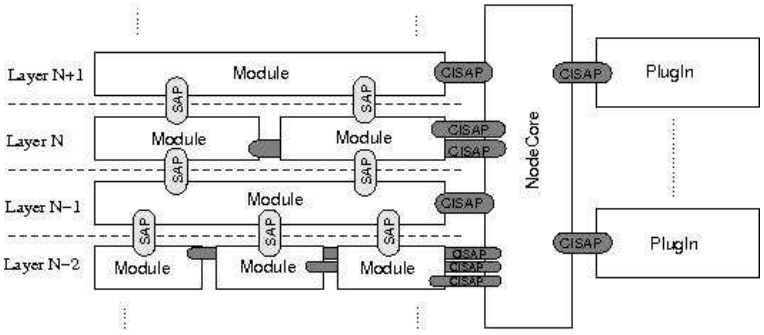


Figura 2.4: In questa ultima figura invece si ha l'introduzione di un'altro blocco che fa sotto il nome di PlugIn; tale entità coordina i nodi e gestisce in modo intelligente la comunicazione tra layer

di supporto alla progettazione e all'implementazione di protocolli sottomarini basato sul simulatore di reti NS2 e sull'appena discusso NS-Miracle. La sua implementazione deriva dalla volontà di spingere gli studi sul *underwater networking* oltre le simulazioni. L'implementazione di soluzioni teoriche su dispositivi attuali, infatti, è la chiave per realizzare un'architettura di comunicazione e di rete che permetta a nodi eterogenei di comunicare in modo affidabile in ambiente sottomarino.

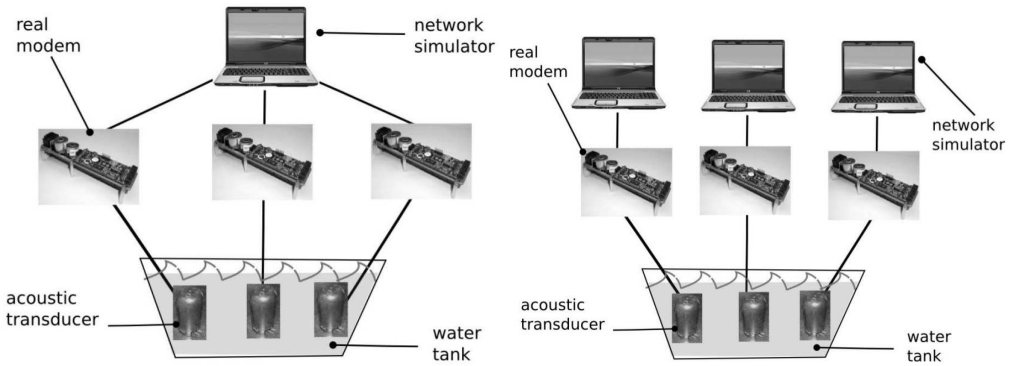
Nel dettaglio, le librerie DESERT Underwater, renderanno possibile il passaggio dalla pura simulazione confinata in un calcolatore alla realizzazione di prototipi attraverso l'utilizzo di reali modem acustici all'interno di ns-miracle stesso; pertanto l'idea è di racchiudere tutti i comandi necessari per comunicare con i modem in un modulo NS-Miracle. In questo modo lo sviluppatore può basarsi su due impostazioni sperimentali:

1. un "**EMULATION setting**" (in piccola scala) dove più modem acustici sono connessi ad un singolo dispositivo e sono controllati da un singolo processo NS-Miracle come mostrato in Figura 6.1a

2. un "**TEST-BED setting**" dove ogni modem acustico è controllato da un unico dispositivo (o da un unica istanza ns-miracle indipendente dalle altre) come in Figura 6.1b

La prima release delle librerie DESERT Underwater comprende una serie moduli NS-Miracle presentati di seguito mediante un approccio top-down (dall'alto verso il basso) all'interno dello stack TPC/IP. Alla fine, inoltre, verranno illustrati altri quattro moduli aggiuntivi implementati per simulare la mobilità dei nodi in scenari di rete sottomarina.

Per distinguere i moduli che appartengono a DESERT da quelli derivanti da altre librerie è stato aggiunto un prefisso `UW-`, il quale non implica che la



(a) Emulation setting (b) Test-Bed setting

Figura 2.5: La a sinistra rappresenta un EMULATION setting in cui una solo istanza NS controlla i tre modem. Nella figura a destra invece abbiamo la rappresentazione di un TEST-BED setting in cui i tre modem sono controllata da istanze di NS indipendenti

soluzione protocollare, implementata da un dato modulo, sia ottimizzata per una situazione sottomarina ma lo scopo è di avere un tool più omogeneo e proiettato verso sviluppi futuri. Dato che il simulatore di rete NS2, e di conseguenza ns-miracle, sono implementati su due differenti linguaggi (C/C++, TCL/OTcl) generalmente possiamo riferirci ad un dato modulo mediante tre nomi:

1. il nome del modulo che corrisponde al nome della cartella che contiene i file sorgenti C/C++
2. il nome della corrispondente libreria dinamica, che deve essere caricata prima dell'utilizzo del modulo stesso
3. il nome del corrispondente oggetto OTcl (da usare nel file di configurazione dei parametri durante la creazione del modulo stesso)

NB. Si tiene a precisare che dove non altrimenti specificato, tutti i moduli DESERT possono essere utilizzati sia per la simulazione che per i due casi sopracitati (EMULATION setting e TEST-BED setting)

A)moduli per application Layer:

Al momento ci sono due moduli: `uwcbr` e `uwvbr` i quali spediscono pacchetti dummy e servono a generare traffico in accordo ai rispettivi meccanismi CBR

e VBR. Per fare in modo che tutti i moduli del layer applicazione funzionino correttamente è necessario che siano collegati al modulo trasport layer.

B)moduli per il transport layer:

Come nel caso precedente sono presenti due moduli: uno semplice chiamato `uwudp` e uno più sofisticato `uwtp`.

C)moduli per il Network layer:

Il network layer ha il compito di fornire gli strumenti per l'interfaccia di rete e meccanismi per data routing. Sono presenti tre moduli per il routing (`uwstaticrouting`, `uwsun` e `uwicrp`) e un modulo più semplice per gestire gli indirizzi di rete in formato IP chiamato `uwip` .

D)moduli per il Data Link layer:

Il cuore del Data Link layer è il MAC (medium access control) che amministra l'accesso al canale acustico di comunicazione. Le librerie DESERT, pertanto, forniscono 6 moduli che implementano lo stesso numero di tecniche MAC: `uwaloha`, `uwsr`, `uw-csma-aloha`, `uwdacap`, `uwpolling` e `uw-t-lohi`. In più è presente uwmll che mappa gli indirizzi IP ai corrispondenti indirizzi MAC.

E)moduli per il Physical layer(interfacce per reali modem-acustici):

Per ora DESERT supporta tre differenti piattaforme hardware per l'emulazione e realizzazione di test-bed: il modem idroacustico S2C sviluppato da Evologics che sfrutta la tecnologia Sweep-Spread-Carrier per dati telemetrici e comunicazioni sottomarine; i modem FSK e PSK della WHOI basati su un DSP della Texas Instruments. Comunque il modulo `uwmphy_modem` che implementa l'interfaccia modem/Ns-miracle, può essere facilmente adattato per supportare diversi dispositivi. Inoltre sono presenti moduli specifici per l'hardware appena descritto (rispettivamente `ms2c_evologics`, `mfsk_whoi_mm` e `mpsk_whoi_mm`) e un modulo addizionale chiamato `uwmphypatch` che serve a preparare lo script OTcl necessario a settare i parametri dei prototipi di rete da testare.

F)moduli addizionali per simulare la mobilità:

Quando reti sottomarine sono simulate la soluzione testata dovrebbe essere esaminata usando accurati modelli che includano, più di ogni altra cosa, pattern realistici di mobilità. DESERT include quattro moduli per simulare la mobilità dei nodi in scenari in 2D così come in 3D: `uwdriftposition` e `uwgmposition`

sono moduli a sé stanti, mentre sia `wossgmmob3d` e `wossgroupmob3d` richiedono l'istallazione di WOSS[6] per funzionare.

[6]WOSS è l'acronimo di World Ocean Simulation System e definisce una libreria per una più accurata modellizzazione del canale sottomarino. Questa libreria può lavorare in due modalità: (a) una semplificata dove vengono usate equazioni empiriche per calcolare aspetti come il ritardo di propagazione, l'attenuazione e la potenza del rumore; (b) una in cui assume una configurazione molto più potente dove le variabili in gioco sono controllate in modo più accurato attraverso leggi delle fisica e processi multi-threaded

Capitolo 3

Modem acustici

Questo capitolo, sempre di carattere introduttivo, inizia con una panoramica sui *modem acustici* e successivamente entra un po' più nel dettaglio illustrando le caratteristiche dei *WHOI micro-modem* utilizzati nei diversi esperimenti di comunicazione sottomarine effettuati in questo elaborato. Nell'ultima sezione verrà presentato lo standard di comunicazione *NMEA 0183* utilizzato nelle trasmissioni con i micro-modem.

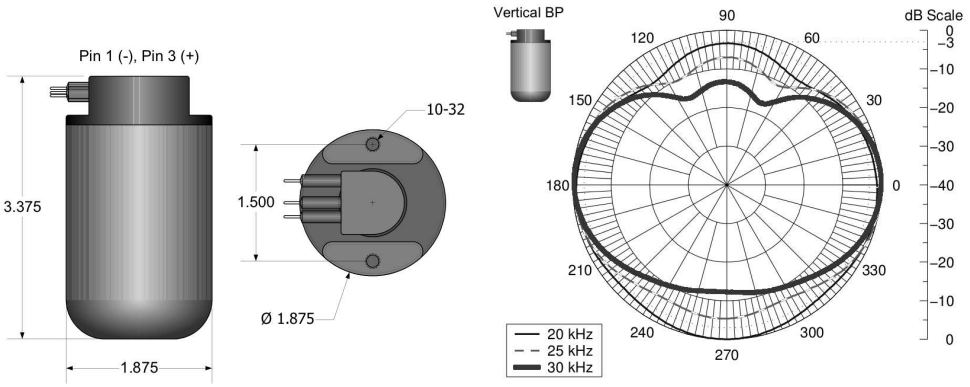
3.1 Introduzione ai modem acustici

Come introdotto nella prefazione a questo capitolo, nel presente paragrafo si andrà ad illustrare in grandi linee la funzionalità dei modem acustici. Tornando a quanto detto nel capitolo precedente, si può dire che questi dispositivi ricoprono, nel progetto Nautilus, un ruolo molto importante in quanto, grazie al loro operato, è possibile mettere in atto l'idea di base del progetto appena citato. Di fatto, connettendo un modem acustico ad un personal computer (e in alcune fasi dei test alle PandaBoard) è stato possibile trasmette i pacchetti, generati all'interno del simulatore di rete, direttamente in acqua attraverso reali segnali acustici. Per tradurre i segnali elettrici del modem in segnali acustici è stato usato un trasduttore elettro-acustico.

La maggior parte dei modem acustici attualmente disponibili per applicazioni sottomarine non sono riconfigurabili; infatti, i loro algoritmi a livello fisico e il formato del flusso di bit sono solitamente hard-coded nel firmware. Oltre a questo aspetto, alcune volte, si aggiunge anche il fatto che protocolli di comunicazione tra coppie di modem include parametri che non sono né controllabili né riconfigurabili dall'utente. Il vantaggio di questi dispositivi è che possono essere utilizzati in maniera del tutto immediata tenendo in considerazione solo lo standard di comunicazione tra modem e simulatore di rete. Di contro però, se si ha la necessità di apportare delle modifiche al livello fisico, questa caratteristica, potrebbe risultare limitativa in abito di ricerca. Per rimediare a questa mancanza esistono in commercio modem riconfigurabili capaci di garantire diversi

livelli di flessibilità nelle configurazioni: dai protocolli di comunicazione utilizzabili a parametri per la gestione della potenza del segnale. La nota dolente che potrebbe interessare le piccole comunità di ricerca è che questi modem vengono venduti in larga scala gravando, in questo modo, su fondi disponibili all'attività di ricerca.

Negli EMULATION setting e nei TEST-BED setting sono stati adoperati dei trasduttori acustici della **BTech Acoustics, LLC**; il trasduttore si basa su attuatori di tipo piezo-elettrici disposti all'interno di un cilindro di gomma il quale, oltre ad isolare a tenuta stagna i componenti elettrici contenuti nel trasduttore, funge anche da membrana acustica nell'ambiente sottomarino. In figura 3.1a e 3.1b vengono riportati, rispettivamente, un'immagine tecnica del BTech's Model BT-2RCL e il diagramma di radiazione dello stesso in cui si può notare come all'aumentare delle frequenze la radiazione si avvicina all'ideale omnidirezionalità.



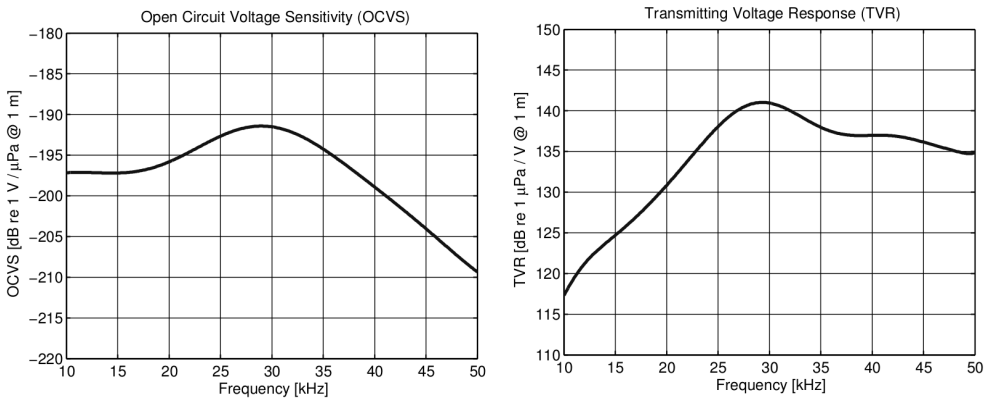
(a) immagine prospettica (b) diagramma di radiazione verticale

Figura 3.1: L'immagine a sinistra riporta la vista laterale e superiore del trasduttore con le relative dimensioni. Nella figura a destra invece viene riportato il diagramma di radiazione a tre frequenze campione.

Soffermandosi ancora per un istante su questo elemento si possono analizzare due distinte risposte in funzione della frequenza attraverso le figure 3.2a e 3.2b

3.2 WHOI micro-modem

Per quanto riguarda il progetto Nautilus si è pensato, almeno in un primo momento, di adoperare modem non configurabili in quanto, altre a garantire un utilizzo immediato nelle simulazioni si è potuta fare la scelta di acquistare un numero limitato di nodi. Tra i diversi modem disponibili in commercio si è optato per il micro-modem del WHOI (Woods Hole Oceanographic Institution) il



(a) Open Circuit Voltage Sensitivity (b) Trasmitting Voltage Response

Figura 3.2: L'immagine a sinistra riporta la risposta del dispositivo in ricezione in condizione di circuito aperto ad un metro dalla sorgente; si nota come l'andamento risulta avere una linearità tra i 10 e i $16KHz$ e un massimo alla frequenza $f_r = 28KHz$. Nel grafico a destra, invece, si analizza il trasduttore come trasmettitore; l'andamento risulta essere diverso dal caso precedente ma presenta sempre un picco di risonanza al $28KHz$

quale per l'interfacciamento con i trasduttori necessita di una scheda di amplificazione fornita sempre dagli stessi produttori; Di seguito andremo ad analizzare questi due elementi.

scheda principale

La scheda principale del Micro-modem è fisicamente molto compatta e, grazie anche al basso consumo energetico, consente implementazioni embedded in sistemi sottomarini (come boe AUV e ROV). Il micro-modem, fornendo ingressi e uscite analogiche unite ad un'elevata potenza di calcolo, si presta molto bene all'elaborazione di segnali acustici orientati alla comunicazione e applicazioni subacquee. Il controllo del dispositivo è reso possibile tramite una porta seriale con standard RS-232, e in aggiunta a questa ne è presenta una seconda per consentire al Micro-modem il controllo di strumenti aggiuntivi.

La logica di controllo del dispositivo è affidata al un Digital Signal Process della Texas Instrument (il C5416) capace di fornire 160 MIPS; la frequenza del clock interna è configurabile via software, in questo modo si ha la possibilità di passare in modalità "basso consumo" non appena, mediante sistemi di controllo, si rileva una modalità idle. Tornando al sistema input/output del modem possiamo dire che l'ingresso è dotato di un stadio di amplificazione, anch'esso programmabile, e di un filtro passa-basso per evitare disturbi di aliasing dal mo-

Frequenza di risonanza (f_r)	$28KHz$
Range di frequenze consigliate	$20 - 40KHz$
TVR alla frequenza (f_r)	141 dB re 1 μPa/V
OCVR alla frequenza (f_r)	-191 dB re 1 V/μPa
Radiazione orizzontale	omnidirezionale
Radiazione verticale	toroidale

Tabella 3.1: In questa tabella sono riportate le caratteristiche nominali del trasduttore BTech's Model BT-2RCL

mento che subito dopo il blocco di ingresso è installato un convertitore A/D a 12-bit e campionamento a 80KHz. In modo duale in uscita il micro-modem presenta un convertitore D/A a 12bit e un filtro di ricostruzione. Anche se il micro-modem della WHOI è classificato tra i modem non programmabili è comunque possibile dietro licenza della casa produttrice aggiornare il firmware, scritto su di una memoria flash, attraverso una connessione con standard JTAG; grazie a quest'ultimo è possibile effettuare anche una sorta di debug del sistema in fase di test. Di seguito vengono schematizzate le caratteristiche del micro-modem:

Range di frequenze in cui opera: $3 - 30KHz$

Modulazione: FSK e PSK

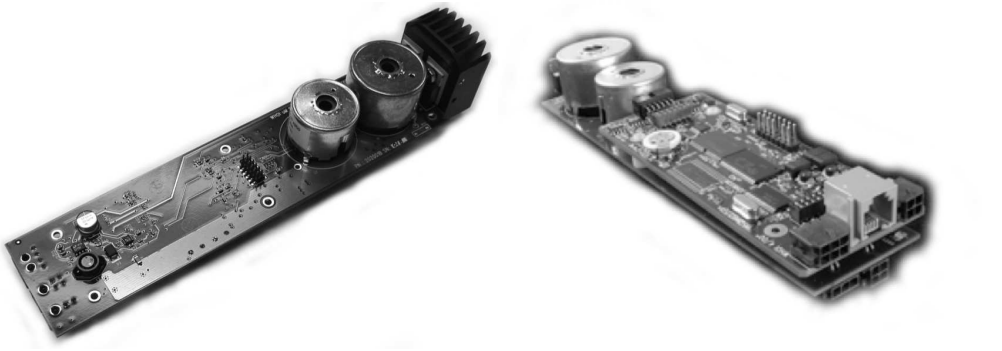
Data Rate: $80 - 5400$ bit/s

Potenze: $<$ 50W o $<$ 100w (in modalità trasmissiva), 158 mW @ 12 V - 211 mW @ 24 V - 230 mW @ 36V (in modalità ricevente), 5.8 mW @ 12 V - 19.3 mW @ 24V - 39.6 mW @ 36 V (in modalità di riposo);

amplificatore di potenza

Il Micro-modem, come anticipato sopra, per poter operare col trasduttore acustico si serve di un'altra scheda, la quale mette a disposizione un unico canale di uscita caratterizzato da un amplificatore di potenza di classe D. L'amplificatore di potenza, inoltre, è stato progettato per essere efficiente e facilmente adattabile a sistemi sottomarini con differenti alimentazioni, a trasduttori diversi e a differenti rapporti frequenza su banda di utilizzo del segnale trasmesso. La scelta di un amplificatore classe D è stata dettata dalle sue caratteristiche di efficienza, e questo implica che per una data applicazione la potenza di uscita è fissata. Il linea di massima una tipica applicazione richiede, durante la trasmissione acustica, una potenza di 50W per ottenere 190dB su 1μPa e un diagramma di radiazione omnidirezionale. L'amplificatore di potenza comprende inoltre un ricevitore a singolo canale grazie a un commutatore di rete per gestire la trasmissione e la

ricezione. Per la quota parte del ricevitore, si ha la fortuna di avere un preamplificatore a basso rumore su larga banda; questo aspetto permette di arrivare ai convertitori A/D con un migliore rapporto segnale-rumore.



(a) scheda di amplificazione

(b) sistema completo (scheda principale + amplificatore)

Figura 3.3: In questa figura vengono riportate due immagine che rappresentano il micro-mode utilizzato dei test; a sinistra si ha solo la scheda di amplificazione, la quale si occupa fondamentalmente di elaborare il segnale a bassa potenza proveniente dalla scheda principale per inviarlo secondo determinate specifiche al trasduttore BTech's Model BT-2RCL.

3.3 Lo standard NMEA

NMEA 0183 (o più comunemente NMEA) è uno standard di comunicazione di dati utilizzato soprattutto in nautica e nella comunicazione di dati satellitari GPS. L'ente che gestisce e sviluppa il protocollo è la National Marine Electronics Association (NMEA). Questo protocollo si basa sul principio che la fonte, detta "talker", può soltanto inviare i dati (sentences) e la ricevente, detta "listener", può soltanto riceverli. Il Micro-modem fornisce un'interfaccia utente attraverso una delle due porte seriali RS-232 disponibili. Il set di comandi per l'interfacciamento è stato sviluppato seguendo, per l'appunto, lo standard NMEA 0183. Di fatto solo alcuni comandi sono necessari per iniziare a trasmettere e ricevere dati tramite il modem acustico; pertanto si tiene a precisare che, anche se molti messaggi sono utilizzati per fornire informazioni sullo stato del sistema, la maggior parte di essi possono essere disattivati se lo si desidera. Siccome l'interfaccia è progettata per essere NMEA compatibile, i comandi di interfaccia e i messaggi di stato sono tutti di tipo ASCII; quindi, tutti i dati, che effettivamente vengono trasmessi in modo acustico, sono inviati in forma binaria in pacchetti di lunghezza fissa.

Capitolo 4

Sistemi Embedded

Il quarto capitolo introduce i concetti base che hanno spinto, nelle fasi di ideazione del progetto Nautilus, a dare spazio a sistemi di questo tipo. Il paragrafo "**Introduzione ai sistemi embedded**" ha il compito di dare una visione globale di questi sistemi integrati e introdurre i paragrafi successivi, dove la descrizione riprenderà un carattere più tecnico. A tal proposito seguirà, quindi, la trattazione delle architetture ARM (di gran lunga utilizzate in sistemi integrati) e un'illustrazione dei sistemi embedded considerati all'interno di questo progetto.

4.1 Introduzione ai sistemi embedded

La maggior parte dei sistemi di elaborazione non è costituita da personal computer, bensì da dispositivi in stretta relazione con l'ambiente in cui operano (special purpose), che in genere hanno una funzione prefissata e, per questo motivo, non richiedono di caricare programmi e, in molti casi, neppure la necessità di avere interfacce tradizionali come tastiera e monitor. Si parla, pertanto, di sistemi invisibili all'utilizzatore ma che al tempo stesso danno un importante contributo ad applicazioni informatiche utilizzate nella vita quotidiana. Questi sistemi, detti *dedicati* o embedded, dominano numericamente il mercato, anche se in modo non conclamato, rispetto ai più noti e visibili sistemi *general purpose* come i personal computer. Si ritiene infatti che attualmente un cittadino del mondo occidentale entri in contatto giornalmente con circa 100 sistemi dedicati. Per convincersi di ciò basti pensare che ciascuno dei seguenti apparati contiene normalmente sistemi elettronici dotati di almeno un microprocessore: telefono cellulare, apri cancello, bancomat, testina di una stampante a getto d'inchiostro, lavatrice, lavapiatti, serratura elettronica, navigatore, carta di credito e così via, fino all'automobile dove si contano addirittura decine di microprocessori. La presenza di tali sistemi in una varietà di prodotti non ha solamente lo scopo di realizzare le funzionalità desiderate ma sempre più spesso diviene il veicolo per introdurre innovazione. L'architettura di un sistema dedicato ricorda quella di un generico sistema di calcolo, con sezioni di elaborazione, comunicazione

e memorizzazione. I requisiti operativi di tali sistemi molto spesso li rendono talmente particolari da richiedere metodologie di progetto, e un approccio alla ricerca di soluzioni tecnologiche, tali da giustificare la nascita di una vera e propria nuova disciplina dell'ingegneria. Per la maggior parte dei sistemi embedded le prestazioni richieste possono essere soddisfatte con una combinazione di hardware dedicato e una quantità limitata di software ottimizzato. A titolo di esempio, basti pensare ad un decoder per una televisione satellitare. Nonostante un sistema come questo debba processare decine di megabit di dati al secondo, la maggior parte del lavoro è svolta da hardware dedicato che separa, regola e decodifica il flusso digitale multicanale in un'uscita video.

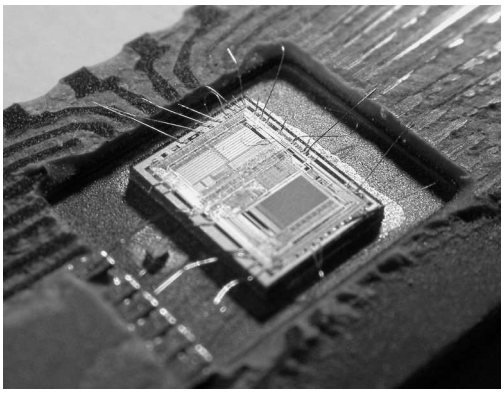


Figura 4.1: Microcontrollore per sistemi embedded senza plastica di protezione (Intel 8742)

Alla CPU embedded spetta determinare i percorsi dei dati nel sistema, o gestire gli interrupt, generare e disegnare la grafica, e così via. Spesso, quindi, gran parte dell'hardware di un sistema embedded deve sottostare a requisiti di prestazione molto meno severi di quelli che, invece, deve rispettare l'hardware primario del sistema stesso. Questo permette all'architettura di un sistema embedded di essere semplificata rispetto a quella di un computer generico che deve eseguire le stesse operazioni usando, ad esempio, una CPU più economica che, tutto sommato, si comporta discretamente bene anche per queste funzioni secondarie. Una caratteristica comune dei sistemi embedded è che spesso richiedono di essere attivi in modo continuato per diversi anni senza generare errori di sistema; pertanto il software ed il firmware di tali sistemi sono progettati e testati con molti più vincoli rispetto ai software dei comuni personal computer dove, su base statistica, sono previsti mediamente due riavvii al giorno. Molti sistemi embedded, infatti, evitano di incorporare componenti con parti meccaniche in movimento (come gli hard disk), poiché meno affidabili rispetto a componenti allo stato solido come le memorie Flash. In aggiunta, i sistemi embedded possono essere fisicamente inaccessibili (come per le trivelle dei pozzi di petrolio,

oppure i componenti lanciati nello spazio), pertanto i sistemi che li contengono devono essere capaci di resettarsi autonomamente in caso di perdita o corruzione dei dati, e nella maggior parte dei casi permettere una connessione remota per l'analisi del sistema o, in caso di operazioni real-time, avere il trasferimento in tempo reale dei dati elaborati. L'affidabilità in termini di operatività full-time è molto spesso ottenuta con l'inserimento di un componente elettronico chiamato watchdog che ripristina il processore se il programma presente sullo stesso non azzera con una certa frequenza il timer interno del componente.

4.2 Architettura ARM

L'architettura ARM identifica una famiglia di microprocessori e microcontrollori di tipo RISC, progettati da *ARM Holdings*[1]. La reale produzione delle CPU basata su questa architettura non è svolta effettivamente da ARM Ltd ma, sotto particolari licenze, viene data la possibilità ad altre aziende di realizzare CPU basate su core ARM. La serie di licenze fornite da ARM Ltd variano a seconda del processore, delle personalizzazioni e del numero di pezzi prodotti. Tutte le licenze ARM prevedono una descrizione hardware dei core e il set completo di strumenti di sviluppo per la realizzazione del software per i processori. Pertanto le aziende acquirenti, dopo aver acquistato la licenza da ARM, si rivolgono a una fonderia di silicio che provvede a realizzare fisicamente il processore basandosi sulla descrizione fornita da ARM Ldt; se tale azienda lo richiede, ARM Ldt fornisce anche i simulatori dell'hardware delle CPU in modo da permettere all'acquirente di modificare la CPU aggiungendovi funzionalità e poi di testarne l'effettiva efficienza senza dover realizzare fisicamente il chip. Grazie alle caratteristiche di basso consumo (in rapportato alle prestazioni) l'architettura ARM domina il settore dei dispositivi mobili[2] dove il risparmio energetico delle batterie è fondamentale.

Attualmente l'architettura ARM copre il 75% del mercato mondiale dei processori per applicazioni embedded, ed è una delle più diffuse architetture a 32 bit del mondo. I processori ARM vengono utilizzati in PDA, cellulari, tablet, lettori multimediali, videogiochi portatili e periferiche per computer (come router, hard disk di rete ecc). Importanti rami della famiglia ARM sono i processori **XScale**

[1] ARM Holdings (London Stock Exchange: ARM, NASDAQ: ARMH) è una società di alta tecnologia con sede a Cambridge, Regno Unito. La società è nota principalmente per la sua linea di processori sebbene sviluppi e venda system-on-a-chip, piattaforme hardware, infrastrutture e software sotto i marchi RealView e KEIL. La società venne fondata come joint venture tra la Acorn Computers, la Apple Computer e la VLSI Technology al fine di sviluppare una linea di microprocessori RISC, inizialmente utilizzati nei computer Acorn Archimedes e nei palmari Apple Newton. In seguito i processori sono stati utilizzati da molti produttori terzi per dispositivi portabili e per applicazioni application-specific integrated circuit (ASIC). La società domina il settore dei microprocessori per telefoni cellulari.

[2] Per dispositivi mobili si intendono tutti quei sistemi di controllo ed elaborazioni dati che non hanno la garanzia di un'alimentazione illimitata nel tempo

e i processori **OMAP** prodotti da Texas Instruments , introdotti nei paragrafi successivi.

Le origini della tecnologia ARM vanno ricondotte alla inglese *Acorn Computers Ldt*[3]. All'inizio degli anni '80 Acorn stipulò un accordo con *British Broadcasting Corporation* (BBC) per lo sviluppo di un nuovo microprocessore per il progetto BBC Computer Literacy. Il successo di questo accordo permise alla Acorn di continuare nello sviluppo e progettare il primo processore RISC commerciale, l'**Acorn RISC Machine** (ARM). La prima versione, ARM1, divenne operativa nel 1985 e fu utilizzata per ricerche interne e sviluppo, oltre ad essere utilizzata come coprocessore nel progetto per la BBC. Sempre nel 1985, Acorn rilasciò l'ARM2, un processore delle stesse dimensioni del precedente, ma più veloce e con più funzioni. Ulteriori migliorie furono apportate con il processore ARM3 (1989). Durante questo periodo Acorn affidò alla società VLSI Technology la fabbricazione dei suoi chip. VLSI poteva commercializzare i chip, ed ebbe un grande successo nel convincere altre aziende a montare processori ARM nei loro prodotti, in particolare come processori embedded.

Il progetto ARM andava incontro alla domanda commerciale di processori per applicazioni embedded ad alte prestazioni, bassi consumi, piccole dimensioni e costi contenuti, ma ulteriori sviluppi andavano oltre le capacità di Acorn. Fu così creata una nuova società, la ARM Ltd., che vedeva la partecipazione di Acorn, VLSI e Apple Computer. Acorn RISC Machine fu rinominata **Advanced RISC Machine**[4]. Il primo prodotto offerto dalla nuova società fu ARM6, una versione migliorata di ARM3. In seguito, la società introdusse diverse nuova famiglie di processori, con sempre più numerose funzionalità e migliori prestazioni. La Tabella 4.1 mostra alcune caratteristiche delle diverse famiglie di architetture ARM; si tiene a precisare che i valori riportati in tabella sono indicativi in quanto quelli reali variano a seconda delle differenti implementazioni. In fine riportando quanto descritto sul sito web di ARM (www.arm.com), i processori ARM sono progettati per soddisfare tre diverse categorie di sistemi:

Sistemi embedded real-time: sistemi di memorizzazione, comparto automobilistico e industriale, applicazioni di rete;

Piattaforme applicative: dispositivi che utilizzano sistemi operativi come Linux, Palm OS, Symbian OS, e Windows CE per applicazioni wireless e di intrattenimento digitale;

[3] Acorn Computers era una compagnia inglese con sede a Cambridge, Inghilterra, nata nel 1978. La compagnia produsse computer che divennero popolari soprattutto in Inghilterra. Tra questi, l'Acorn Electron, il BBC Micro e l'Acorn Archimedes. Il BBC Micro dominò il mercato dei computer nelle scuole negli anni '80 e inizi '90.Nel 2000 la compagnia fu divisa in diverse aziende, anche se lasciò in eredità moltissimi progetti, quali lo sviluppo dei computer basati sui processori RISC. Alcune di queste aziende, come la ARM Holdings, sono tutt'oggi operative.

[4] La società abbandonò la denominazione Advanced RISC Machine alla fine degli anni '90, di fatto al giorno d'oggi tale architettura è conosciuta semplicemente come ARM

Famiglia	Funzionalità rilevanti	Cache	MIPS@MHz
ARM1	32-bit RISC	NO	
ARM2	Istruzioni di moltiplicazione e scambio; unità di gestione della memoria, grafica e processore I/O integrati	NO	7 MIPS @ 12 MHz
ARM3	Primo utilizzo della cache del processore	4 KB unificata	12 MIPS @ 25 MHz
ARM6	Primo a supportare indirizzi a 32 bit; unità floating point (virgola mobile)	4 KB unificata	28 MIPS @ 33 MHz
ARM7	SoC (System on a Chip) integrato	8 KB unificata	60 MIPS @ 60 MHz
ARM8	Pipeline a 5 stadi; predizione statica dei salti	8 KB unificata	84 MIPS @ 72 MHz
ARM9		16 KB / 16 KB	300 MIPS @ 300 MHz
ARM9E	Istruzioni DSP migliorate	16 KB / 16 KB	220 MIPS @ 200 MHz
ARM10E	Pipeline a 6 stadi	32 KB / 32 KB	
ARM11	Pipeline a 9 stadi	Variabile	740 MIPS @ 665 MHz
Cortex	Pipeline superscalare a 13 stadi	Variabile	2000 MIPS @ 1 GHz
XScale	Processore per le applicazioni: pipeline a 7 stadi	32 KB / 32 KB L1 512 KB L2	1000 MIPS @ 1,25 GHz

Tabella 4.1: In questa tabella viene riportata una sorta di evoluzione dell'architettura ARM; si tiene a precisare che i valori numerici riportati sono indicativi in quanto una rappresentazione più dettagliata occuperebbe molto più spazio e, al tempo stesso, si discosterebbe dalle tematiche di questo elaborato

Applicazioni per la sicurezza: smart card, SIM card e terminali Bancomat.

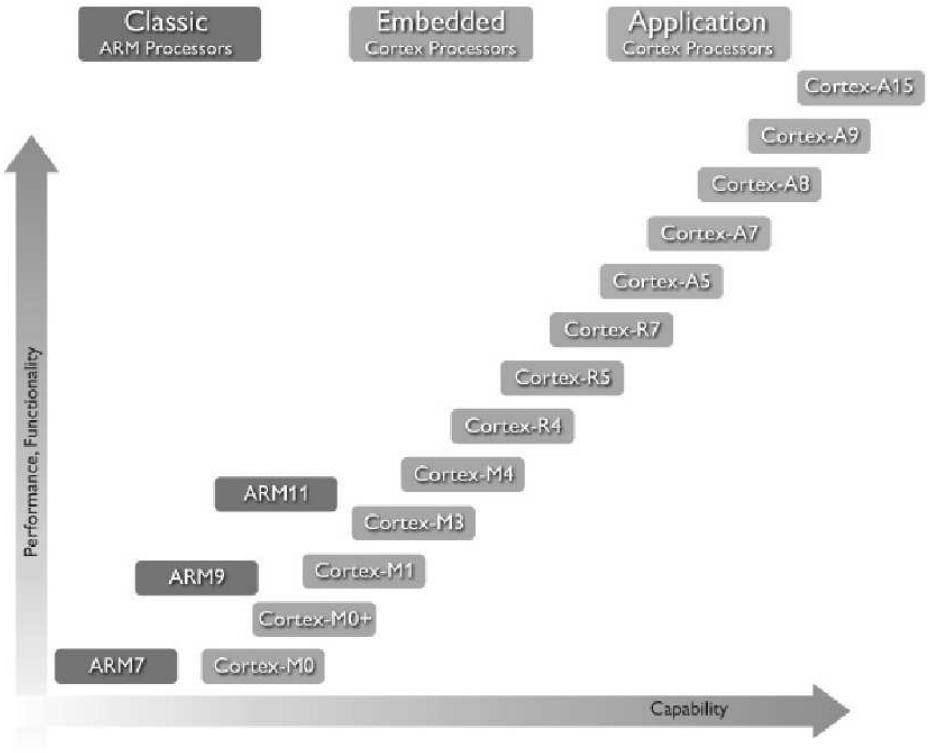


Figura 4.2: In questo grafico vengono riportate le prestazioni e le funzionalità delle diverse famiglie ARM (Marzo 2012, da www.arm.com)

4.2.1 OMAP

La tecnologia OMAP, acronimo di Open Multimedia Application Platform (Piattaforma per applicazioni multimediali aperte), sviluppata da Texas Instruments è una categoria di sistema proprietario su chip (SoC) per le applicazioni multimediali portatili e mobili. I dispositivi OMAP in linea di massima includono un'architettura general-purpose basata su processori ARM (sotto licenza della ARM Ldt.) sia a singolo core che multi core.

Stando alle informazioni reperite dalle documentazioni fornite da Texas Instruments la famiglia OMAP può essere suddivisa in 3 differenti gruppi sia per prestazioni che per applicazioni, pertanto abbiamo:

- Processori per applicazioni ad alte prestazioni
- Processori per applicazioni multimediali di base

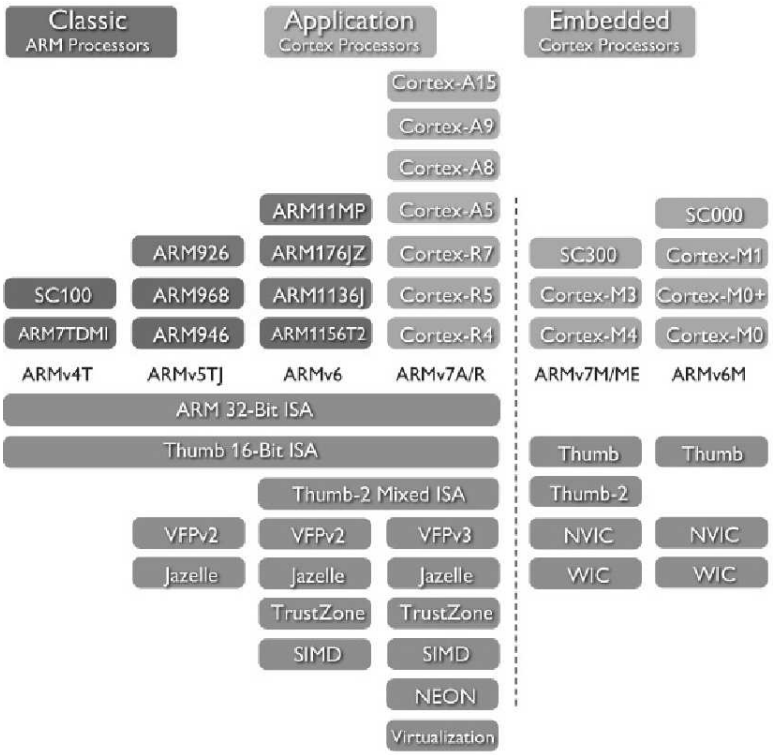


Figura 4.3: In questa figura viene fatta una panoramica sulle diverse architetture ARM (Febbraio 2012, da www.arm.com)

- Processori per applicazioni come Modem integrato

I **Processori per applicazioni ad alte prestazioni** in origine erano destinate ad applicazioni come *smartphone* in cui si ha la necessità di avere una notevole potenza di calcolo, rispetto ai comuni cellulari, per poter garantire agli utenti la funzionalità di sistemi operativi fluidi e reattività delle diverse applicazioni software installate. I processori ARM facenti parte a questa categoria sono i seguenti:

- OMAP 1
  La famiglia OMAP-1 ha origine dal potenziamento di un core ARM già sviluppato dalla Texas Instruments, e poi, in seguito, standardizzato con il nome di ARM926. Al giorno d'oggi la famiglia OMAP1710 è ancora disponibile per le aziende che sviluppano dispositivi cellulari. In effetti i processori OMAP-1 vengono utilizzati in centinaia di modelli di telefoni cellulari come ad esempio l'internet tablet della nokia (il Nokia 770).

- OMAP 2
  Questa seconda famiglia è fondamentalmente una evoluzione dell'OMAP-1; in effetti questi processori, destinati sempre al mercato cellulare, risultano avere frequenze di clock maggiori rispetto alle versioni precedenti.
- OMAP 3
  Il vero successo di questa tecnologia lo si vede con l'OMAP-3. Di fatto, per soddisfare diverse esigenze, questa serie di processori ARM è stata suddivisa in tre gruppi l'OMAP34x, l'OMAP35x, e l'OMAP36x. La tecnologia video contenuta nell'OMAP 3 deriva in parte dal prodotto DaVinci il quale è stato il primo ad avere il supporto del DSP TMS320C64x; una lista dei processori ARM basati su questa tecnologia è riportata in tabella 4.2
- OMAP 4
  La quarta generazione degli OMAP: OMAP 4430, 4460 (in precedenza denominata 4440), e 4470 implementano tutti un ARM Cortex-A9 dual-core. Il 4470 contiene inoltre due Cortex-M3 funzionanti a 266 MHz per sgravare il Cortex-A9 da compiti computazionalmente meno intensivi; questo accorgimento aiuta ad aumentare l'efficienza energetica. Gli OMAP 4430 e 4460 utilizzano un PowerVR SGX540 integrato, il quale si occupa dell'accelerazione grafica 3D con una frequenza di clock di 304 MHz e 384 MHz rispettivamente. Il 4470 ospita un PowerVR SGX544 GPU capace di supportare le DirectX 9 e quindi consetire l'utilizzo di Windows 8 in dispositivi aventi questo tipo di processore. Tutti gli OMAP 4 sono dotati di un acceleratore hardware IVA3 multimediale grazie ad un DSP programmabile che permette di avere un Full HD 1080p.
- OMAP 5
  L'OMAP di quinta generazione usa: un dual-core ARM Cortex-A15 con due ulteriori core Cortex-M4 per la minimizzazione dell'energia, due core grafici (PowerVR SGX544MP) e un acceleratore grafico. Altre caratteristiche molto importanti sono espresse: dal supporto fino a 8 GB di memoria dual channel DDR3, da un'uscita video HDMI 1.4, 3 porte USB 2.0, una porta 3.0 e un controller SATA 2.0. Attualmente l'OMAP-5 non è ancora in commercio ma da alcuni test effettuati da Texas Instruments si possono apprezzare delle grandi potenzialità, per alcuni aspetti anche maggiori del potentissimo Tegra3[5] dell'Nvidia

[5] I processori Tegra sono dei System-on-a-chip (SoC) dalle dimensioni molto contenute (il primo esponente è più piccolo di una moneta da 10 centesimi), ovvero contengono all'interno

modello	set di istruzioni CPU	CPU	Dispositivi
OMAP3430	ARMv7	600 MHz ARM Cortex-A8	Motorola Droid/Milestone, Nokia N900, Palm Pre, Samsung i8910, Sony Ericsson Satio
OMAP3440	ARMv7	800 MHz ARM Cortex-A8	Archos 5 (Gen 7), Motorola Milestone XT720, Motorola Titanium XT800,[citation needed] Samsung Galaxy A (SHW-M100S)
OMAP3503	ARMv7	600 MHz ARM Cortex-A8	Gumstix Overo Earth
OMAP3530	ARMv7	720 MHz ARM Cortex-A8	Alico's Kinetic 3500, Always Innovating Touch Book, BeagleBoard, Embest DevKit8000, Gumstix Overo Water, IGEPv2, OpenSourceMID K7 MID, Oswald, Overo Water, Pandora
OMAP3630	ARMv7	600 MHz 1.2 GHz ARM Cortex-A8	Motorola Bravo, Motorola Defy, Archos 28, Archos 32, Archos 43, Archos 70, Archos 101, LG Optimus Black, LG Optimus Bright, Motorola Cliq 2, Motorola Droid 2/Milestone 2, Motorola Droid X, Motorola Defy, Motorola Defy+, Nokia N9, Nokia N950, Palm Pre 2, Panasonic P-07C, Samsung Galaxy S LCD

Tabella 4.2: OMAP3

modello	set di istruzioni CPU	CPU	Dispositivi
OMAP4430	ARMv7	1-1.2 GHz dual-core ARM Cortex-A9	BlackBerry Fujitsu Arrows X LTE F-05D, Fujitsu Arrows Z ISW11F, Panasonic Lumix Phone 101P, Panasonic Lumix Phone P-02D, LG Prada 3.0, LG Optimus 3D P920, LG Optimus 3D Max, LG Optimus L7, Motorola Droid RAZR, Motorola Xyboard, PandaBoard, Samsung Galaxy S II (GT-I9100G), Sharp Aquos Phone 102SH, TianyeIT CIP411, Toshiba AT200 Excite, Amazon Kindle Fire, Archos 80 (Gen 9), Archos 101 (Gen 9),
OMAP4460	ARMv7	1.2-1.5 GHz dual-core ARM Cortex-A9	Archos 80 Turbo (Gen 9), Archos 101 Turbo (Gen 9), Samsung Galaxy Nexus, Huawei Ascend D1, Huawei Ascend P1/P1S, Pandaboard ES, Sharp Aquos Phone 104SH, Variscite VAR-SOM-OM44
OMAP4470	ARMv7	1.5-1.8 GHz dual-core ARM Cortex-A9	

Tabella 4.3: OMAP4

modello	set di istruzioni CPU	CPU	GPU
OMAP5430	ARMv7	2 GHz dual-core ARM Cortex-A15	PowerVR SGX544MP2 (dual-core) + dedicated 2D graphics accelerator
OMAP5432	ARMv7	2 GHz dual-core ARM Cortex-A15	PowerVR SGX544MP2 (dual-core) + dedicated 2D graphics accelerator

Tabella 4.4: OMAP5

4.2.2 XScale

XScale è una famiglia di microprocessori sviluppata da Intel a partire dalla quinta generazione di processori ARM. La famiglia XScale è divisa nelle sottofamiglie

- IXP
- IXC
- IOP
- PXA

L'architettura XScale si basa sul set di istruzioni ARM v5TE con l'aggiunta di istruzioni in virgola mobile. Come anticipato nella tabella 4.1 l'XScale utilizza un'architettura RISC a pipeline con 7 stadi per gli interi e 8 per la memoria. I processori sono i successori degli StrongARM che Intel acquisì dalla DEC quando, per porre fine a una causa legale che DEC aveva intentato verso la società, Intel acquistò l'intera divisione dedicata allo sviluppo dei semiconduttori. Pertanto Intel utilizzò lo StrongARM per rimpiazzare le sue precedenti linee di processori RISC i860 e i960. La pipeline dei processori XScale a differenza della maggior parte dei processori RISC è una singola pipeline che dopo aver decodificato l'istruzione si divide in tre sotto pipeline differenti. Le istruzioni a seconda della loro tipologia vengono eseguite da una delle tre pipeline e le tre pipeline possono eseguire le istruzioni in parallelo dato che non sono bloccante; questo implica che il processore esegua le istruzioni fuori ordine. Nel giugno del 2006 la famiglia PXA è stata venduta alla Marvell Technology Group la quale ha continuato a credere sulla tecnologia XScale mettendo subito in commercio, nel 2007, la nuova famiglia di processori PXA3xx.

4.3 PandaBoard

La PandaBoard è definita una Single-Board Computer (SBC); questa scheda caratterizzata da un costo e da consumi, in termini di energia, relativamente bassi è sviluppata e prodotta direttamente dalle Texas Instruments. Le caratteristiche di questa board sono:

- un processore dual-core ARM Cortex-A9 da 1GHz (OMAP4430),
- una GPU PowerVR SGX540 da 304 MHz
- un Digital Signal Process C64x DSP per l'accelerazione hardware,

di un unico package sia la CPU sia la circuiteria tipica dei chipset e i controller di I/O e della RAM. L'architettura Tegra è di tipo eterogeneo; essa infatti è formata da più processori, ognuno ottimizzato per uno specifico ambito di impiego

- 1 GB di SDRAM DDR2.
- uno slot per schede Secure Digital di tipo SDHC fino a 32 GB
- un connettore Ethernet 10/100
- connettività wireless 802.11g e Bluetooth
- due uscite video: DVI e HDMI.
- un connettore autdio da 3.5 mm.
- due connettore USB 2.0 host
- e un connettore USB On-The-Go

La PandaBoard, grazie alle sue caratteristiche molto prestanti, ha dato, all'interno di questo elaborato, la possibilità di realizzare alcuni aspetti di base; come la compilazione dei sorgenti direttamente sul dispositivo embedded.

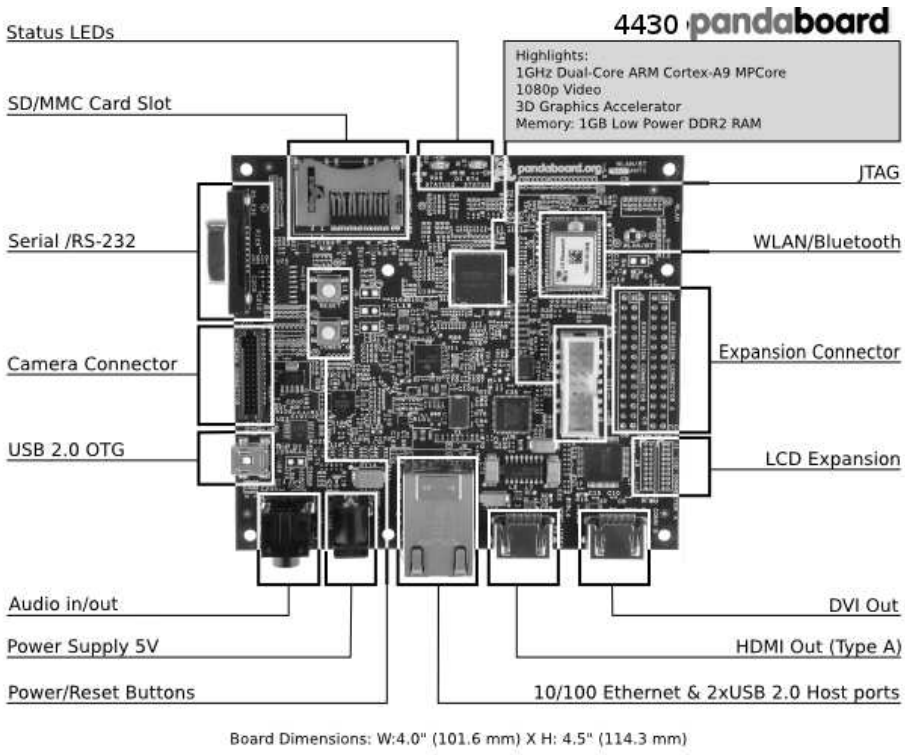


Figura 4.4: In questa immagine viene riportata una foto della PandaBoard utilizzata in questo elaborato.

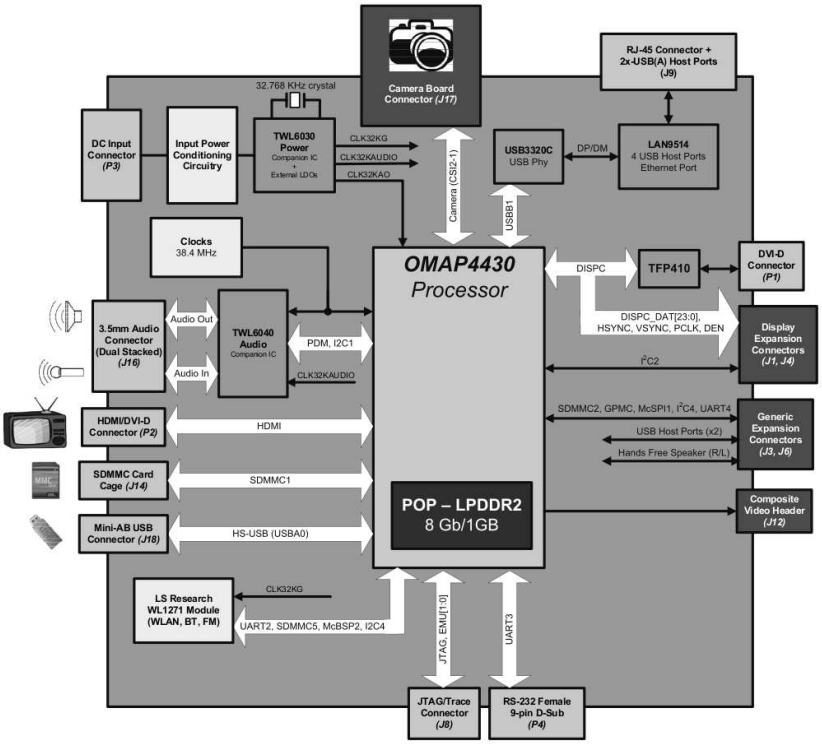


Figura 4.5: In questa immagine viene riportato lo schema ablocchi della PandaBoard di Fig4.4.

4.4 Gumstix

Gumstix è una società fondata nel 2003 da Gordon Kruberg che produce piccoli single-board computer. Il nome Gumstix si riferisce alla dimensione del primo mini-computer in quanto aveva le dimensioni approssimative di una gomma da masticare. A differenza della scheda principale le schede di espansione sono sotto una licenza Creative Commons Share-Alike.

A differenza della PandaBoard, la Gumstix segue due linee di produzione: una serie che va sotto il nome di Overo è basata su un processore OMAP35xx della Texas Instruments e un'altra serie che prende il nome di Verdex Pro, basata su processori XScale PXA270 della Marvell. La differenza tra i due prodotti consiste, fondamentalmente, nella potenza di calcolo dove il processore OMAP ha sicuramente la meglio; con ciò non si vuole alludere al fatto che un prodotto sia migliore di un altro, ma soltanto che le due soluzioni sono destinate ad applicazioni differenti in cui la discriminante è data dal rapporto prestazioni/consumo energetico dove in alcuni casi la necessità di avere dispositivi che consumano poca potenza è più importante della potenza di calcolo.

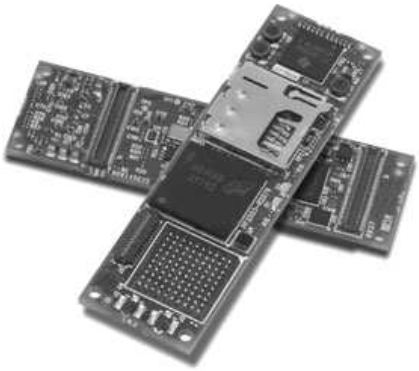


Figura 4.6: In questa immagine viene riportato la foto della Gumstix

4.5 NetDCU 5.2

Come NetDCU4, NetDCU5.2 utilizza CPU Intel XScale PXA255. Al giorno d'oggi la potenza di calcolo di questa scheda risulta essere al di sotto della media, ma le dimensioni compatte e una varietà di interfacce basilari la rendono ancora un dispositivo utilizzabile. La NetDCU5.2 è utilizzabile in applicazioni in cui si ha la necessità di un display a risoluzione SVGA, XGA e SXGA e quando si ha bisogno di trasferimento dati via Ethernet o WLAN. Date le basse prestazioni del dispositivo, il sistema operativo installato sulla NetDCU5.2 è costruito su misura; questo aspetto implica la necessità di avere un software di supporto per lo sviluppo della applicazioni da far girare sul dispositivo. La gamma di applicazioni per le quali la NetDCU5.2 è destinata sono oramai limitate; dettato anche dal fatto che il processore PXA255 è fuori produzione, le applicazioni commerciali per questo dispositivo scalano col passare degli anni. Ciò nonostante l'interesse nato, in questo elaborato, nei confronti del dispositivo in questione sono dettati dal fatto che in alcuni sistemi sottomarini sono presenti dispositivi NetDCU5.2. Pertanto la possibilità di riuscire a testare il pacchetto NS2/NS-Miracle/DESERT su un dispositivo di questo genere ha spinto l'analisi dei dispositivi embedded anche in questa direzione. Nella tabella 4.5 sono riportate le caratteristiche di questo dispositivo embedded.

Alimentazione	+5VDC/±5%
Corrente assorbita	<1,2A
Interfaccia seriale	3x RS232
Ethernet	10/100 Mbit basara su cavo incrociato
Touch-Screen	connessione diretta a un Touch-Panel resistivo
ingressi e uscite digitali	max. 21 porte I/O
ingressi e uscite analogiche	max. 4 ingressi, 10 bit, Audio-Codec AC97
PCMCIA	1 slot
USB	2 porte USB-Host 1.1
interfaccia LCD	CSTN: 800 x 600 Pixel, 256 colori TFT: 1280 x 1024 Pixel, 65536 colori
Memoria programmi	32 MByte
RAM	SDRAM da 32 MByte
Processore	PXA255 (da 400MHz)
Range di Temperatura	0°C ... 70°C
Dimenzioni	100 x 80 x 10 mm
Peso	60 g

Tabella 4.5: In questa tabella vengono riportate le caratteristiche della NetDCU 5.2

Capitolo 5

Implementazione di sistemi embedded per la prototipazione di reti sottomarine

Il presente capitolo esporrà quello che è stato il lavoro svolto in questi mesi. Si partirà da una introduzione al progetto Nautilus per arrivare, all'ultima sezione, in cui l'intero sistema (hardware-software) è pronto per effettuare i test presentati nel capitolo successivo

5.1 Il progetto Nautilus

Come accennato nell'introduzione, il progetto Nautilus nasce dall'idea ingegneristica presente nel gruppo del laboratorio SIGNET. Questo progetto, sovvenzionato dalla Italian Istitute of Tecnology (IIT), fonda le basi su concetti di trasmissione a pacchetti già conosciuti in letteratura, ma mira con ambizione a ridefinire protocolli di comunicazioni adatti alle trasmissioni radio sottomarine. Il progetto Nautilus intuisce che le esplorazioni sottomarine permettono di scoprire nuove forme di vita come specie animali e botaniche. Il monitoraggio subacqueo può aiutare gli scienziati a comprendere meglio questo ambiente in modo da capire i meccanismi che regolano la vita del nostro pianeta e, al tempo stesso, permette a noi di proteggerlo. Tutto ciò fa capire che queste attività possono aiutare altri gruppi di ricerca a rilevare movimenti tettonici, tsunami in arrivo, inquinamento dell'acqua, riscaldamento globale e molti altri fattori che di riflesso influenzano la nostra vita. Detto ciò, il monitoraggio del mondo sottomarino, al giorno d'oggi, è un compito molto impegnativo a causa della sua complessità in termini di dimensioni e disomogeneità del mezzo; a questo si aggiunge anche la limitata tecnologia usata per il monitoraggio stesso. Di fatto, le varie tecniche si appoggiano su impianti costosi e molto spesso connessi via cavo; per analisi in cui è prevista un'attività di monitoraggio a lungo termine

si fa in modo che personale specializzato vada a posizionare i sensori per poi, in un secondo momento, andare a recuperare le informazioni acquisite. Queste escursioni non sempre sono di facile realizzazione, infatti in alcuni casi possono risultare anche pericolose. Pertanto il progetto Nautilus, considera la possibilità di ridisegnare l'aspetto del monitoraggio sottomarino basandosi su una rete di sensori distribuita. Di fatto, questo progetto immagina un ambiente in cui si potrebbe semplicemente diffondere un numero di nodi sensore in acqua: alcuni sul fondo del mare, altri galleggianti a diverse profondità, ed altri ancora capaci di muoversi. Questi dispositivi, inoltre, dovranno essere in grado di parlare gli uni agli altri e, in modo autonomo, organizzarsi in una rete gestendo lo scambio dati, ed eventualmente trasferire tutte le informazioni ad un punto di raccolta dove l'utente specializzato potrà scaricare le informazioni raccolte in maniera più facile e meno costosa. L'interazione dei vari sensori dovrà basarsi sulla versatilità dei nodi; questo fa si che la rete acquisisca una topologia dinamica di modo che, se alcuni sensori dovessero guastarsi o essere momentaneamente oscurati, la rete sarà in grado di reagire provvedendo ad una nuova configurazione topologica. Come accennato in precedenza, il progetto Nautilus prevede dei veicoli sottomarini autonomi (AUV) i quali possono muoversi liberamente nello scenario subacqueo. Questi dispositivi, pertanto, possono passare in rassegna i vari sensori in modo da scaricare i dati e riportarli ai punti di raccolta citati prima. Considerando una gestione più articolata dei AUV, è possibile considerare missioni in cui i veicoli sottomarini autonomi lavorano in cooperazione. In definitiva, anche se alcune di queste idee potrebbero sembrare fantascientifiche, le tecnologie moderne possono fare in modo che ciò possa essere realizzabile. L'elemento chiave, che attualmente manca e, che il progetto Nautilus vuole fornire per trasformare l'idea teorica in realtà, è la disponibilità di un'infrastruttura di comunicazione efficace e flessibile che renda possibile a tutti i dispositivi la capacità di collaborare e di agire in modo coordinato. Lo sviluppo di questo progetto potrà avere un grande impatto sulle comunità di ricerca nel settore delle comunicazioni e del networking, così come nella robotica e nelle scienze sottomarine. A tal proposito il risultato conclusivo del progetto Nautilus sarà basato su comunicazioni innovative e soluzioni di protocollo avanzate, nonché su una profonda comprensione delle esigenze applicative e delle caratteristiche in termini di missioni collaborative. Le principali aree di ricerca si occuperanno di: modelli di canale sottomarini, comunicazioni acustiche efficienti e protocolli di rete (accesso multiplo, routing, trasporto).

5.2 Contributo della tesi

Trattandosi di un lavoro complesso e articolato, il progetto Nautilus, parte cercando di pianificare la ricerca in modo da avere sempre risultati il più possibile attendibili; Il primo ostacolo è stato il fatto che la prototipazione dei protocolli per le comunicazioni sottomarine soffre della mancanza di un'affidabile modello

di canale subacqueo. Non potendosi affidare a delle simulazioni di questo tipo, si è pensato di lavorare in ambiente simulato (NS2) per buona parte dello stack protocollare e, per il layer riguardante il Livello Fisico[1] si è fatto in modo che il simulatore si interfacciasse con i modem acustici; in questo modo il canale trasmissivo è diventato un canale a tutti gli effetti reale.

Il Contributo principale di questo elaborato è stato quello di fornire un dispositivo embedded capace di ospitare il simulatore di rete (NS2), e allo stesso tempo interfacciare questo tool di simulazione con i modem acustici attraverso il protocollo RS232[2].

La scelta di iniziare a lavorare con un dispositivo come la Pandaboard è stata dettata dal fatto che si è pensato di andare subito a sperimentare se un dispositivo embedded fosse capace di gestire simulazioni di rete. Di fatto si è optato per un hardware molto prestante e con una notevole potenza di calcolo per essere un sistema embedded. Nei requisiti del simulatore di rete c'è la necessità di avere una distribuzione linux come sistema operativo. Restando sempre sull'idea di avere un feedback immediato nel corso del progetto e, quindi, non trovarsi a dover prendere delle decisioni e aspettare del tempo prima di capire se è stata la scelta giusta, si è pensato di usare una distribuzione molto stabile e testata già da molte persone; pertanto la distribuzione considerata in questo elaborato è stata *ubuntu 11.10 server*. In questo modo, anche se in una prima fase non è stato definito un sistema operativo ottimizzato per il dispositivo embedded in questione, si è potuto in tempi relativamente brevi avere a disposizione lo strumento fondamentale di questo elaborato. Lo step intermedio, antecedente ai primi test in acqua, è stato quello di disporre di cavi per interfacciare elettricamente la PandaBoard col MicroModem e il MicroModem con il trasduttore elettro-acustico. Presa visione dei data-sheet[3] dei tre dispositivi, sono stati acquistati degli appositi connettori si è proceduto nell'assemblaggio dei cavi utilizzando un saldatore a stagno. Dopo i primi beta-test, appurato che il lavoro fino a quel punto procedeva nella direzione corretta, si è passati alla fase di analisi del dispositivo, non ché alla ricerca di ottimizzazioni del sistema. A tal proposito è stato prodotto un programma, scritto in linguaggio C++, per il controllo della RAM durante le simulazioni in acqua(in definitiva è stato fatto un benchmark[4]). Da questi test si evince, che l'occupazione della RAM è molto dipendente dalla simulazione che viene fatta eseguire; di fatto l'utilizzo della memoria segue temporalmente l'esecuzione dello script TCL[5]. Dai grafici

[1] Il livello fisico corrisponde al primo livello dello stack ISO/OSI, detto anche Phisical Layer

[2] EIA RS-232 (Recommended Standard 232) è uno standard EIA equivalente allo standard europeo CCITT V21/V24, che definisce un'interfaccia seriale a bassa velocità di trasmissione per lo scambio di dati tra dispositivi digitali.

[3] Con il termine data-sheet si indica la documentazione che riassume le caratteristiche di un componente

[4] Con il termine benchmark si intende un insieme di test software volti a fornire una misura delle prestazioni di un computer per quanto riguarda diverse operazioni.

[5] Tcl (acronimo di Tool Command Language), è un linguaggio scripting creato da John

si può vedere come la maggior parte dalla memoria allocata è concentrata nelle prime fasi dell'esecuzione in cui vengono caricate librerie di supporto alla simulazione e allocare variabili di ambiente. Un altro studio, volto all'ottimizzazione, è stato quello di analizzare i sorgenti del simulatore in modo da andare ad eliminare tutto il codice non necessario alla produzione del solo codice necessario alla prototipazione sottomarina; questo aspetto verrà discusso nel paragrafo 5.4

5.3 Linux su PandaBoard

La PandaBoard di fondo può ospitare diversi sistemi operativi[6]; anche se, come già detto in precedenza, si è pensato di utilizzare Ubuntu, in quanto possiede un forte supporto tecnico e questo aspetto sarebbe potuto essere di supporto.

5.3.1 Download dell'immagine

Detto ciò, il primo passo per l'installazione di Ubuntu sulla PandaBoard è stato scaricare dai repository di Ubuntu la distribuzione adatta per architettura ARM-OMAP4[7]; il link di riferimento per scaricare l'immagine della distribuzione è il seguente:

http://cdimage.ubuntu.com/releases/11.10/release/

Nella pagina web di riferimento si hanno due possibilità di scelta: **Preinstalled desktop image** e **Preinstalled server image**. Nel caso in questione, si è scelto di usare la seconda versione, dal momento che lo scopo è quello di avere almeno tre PandaBoard controllate da remoto. Il link preso in considerazione è, pertanto, il seguente:

http://cdimage.ubuntu.com/releases/11.10/release/ubuntu-11.10-preinstalled-server-armel+omap4.img.gz

---

la **versione desktop** prevede un'istallazione completa focalizzata all'uso della pandaboard fatta attraverso interfaccia grafica; quindi oltre ad installare l'ambiente grafico, prevede tutta una serie di configurazioni rivolte a questo utilizzo. E' da tenere in considerazione che

Ousterhout generalmente considerato di facile apprendimento (rispetto ai linguaggi della sua generazione) ma allo stesso tempo potente. Viene comunemente usato per prototipizzare rapidamente e testare applicazioni interpretate, anche dotate di GUI. La pronuncia suggerita per l'acronimo Tcl è tickle

[6] Al momento i sistemi operativi supportati dalla pandaboard sono: Ubuntu, Android ed Angstrom

[7] Gli OMAP, acronimo di Open Multimedia Application Platform (Piattaforma per applicazioni multimediali aperte), sono dei processori RISC specificatamente progettati per applicazioni multimediali, costruiti dalla Texas Instruments sotto licenza ARM. Molti telefoni cellulari e smartphone (per esempio i nokia Symbian) utilizzano questa tecnologia.

già dalle prime fasi di installazione è necessario collegare un monitor (con ingresso HDMI o DVI/D) direttamente alla PandaBoard.

la **versione server**, rispetto alla precedente, ha il vantaggio di essere molto più leggera, sia in termini di spazio occupato che di Ram utilizzata. Questa versione pertanto risulta più adatta a dispositivi embedded controllati da remoto, ad esempio attraverso una connessione ssh(Secure SHell). In questo caso, per quanto riguarda l'istallazione, è necessario avere una connesione seriale (RS232).

5.3.2 Scrittura dell'immagine su di una memoria SD

In questa sezione si andrà ad illustrare come è stata effettuate la copia, bit a bit dell'immagine appena scaricata, su di una SD card.

1. E' stata inserita la Secure Digital nell'apposito lettore di un normale personal computer.

2. Siccome con le ultime distribuzioni gli archivi di massa vengono montati in automatico è stato necessario smontare le eventuali partizioni contenute nella SD. Pertanto con il comando:

   ```
   mount -l
   ```

   è stato possibile avere una lista delle partizioni montate. Se vengono, ad esempio, individuate partizioni montate sotto il nome di sdd1, oppure in generale con sddN (con N numero intero), allora è possibile smontare la partizione attraverso il seguente comando:

   ```
   umount /dev/sdd1
   ```

3. A questo punto è stato considerato il "raw device name"[8]. Questo perché è necessario prendere in considerazione tutti i settori dalla SD, compresi i primi 512byte riservati al *Master Boot Record*[9] A tal proposito è stato lanciato il seguente comando, naturalmente quest'ultimo ha richiesto come parametro il percorso della cartella che contiene l'immagine, scaricata nei passi precedenti.

   - (per la versione **Preinstalled desktop image**)

     ```
     sudo sh -c 'zcat ./ubuntu-11.10-preinstalled-desktop-armel+omap4.img.gz
     |dd bs=4M of=/dev/sdd ; sync'
     ```

[8] Per raw device name si intende **sdd** che si riferisce all'intero store-device (dal primo all'ultimo settore); **sdd1**, **sdd2**, ..., **sddN** si riferisce, invece, all'n-esima partizione.

[9] Il master boot record (MBR), nell'architettura dei PC IBM contiene la sequenza di comandi necessaria per l'avvio del sistema operativo oltre ad includere la tabella della partizioni.

- (per la versione **Preinstalled server image**)

```
    sudo sh -c 'zcat ./ubuntu-11.10-preinstalled-server-armel+omap4.img.gz
|dd bs=4M of=/dev/sdd ; sync'
```

5.3.3 Installazione attraverso RS232

Come già anticipato in alcune note precedenti, per l'installazione della versione Preinstalled server image, è stato necessario instaurare una connessione, attraverso il protocollo RS232, tra PC e PandaBoard. Pertanto, prima di dare il via all'istallazione si è proseguito facendo partire il programma *minicom*[10] in modalità settaggio, pertanto:

1. `sudo minicom -s`

   You'll get a screen as follows:

```
          +-----[configuration]------+
          | Filenames and paths      |
          | File transfer protocols  |
          | Serial port setup        |
          | Modem and dialing        |
          | Screen and keyboard      |
          | Save setup as dfl        |
          | Save setup as..          |
          | Exit                     |
          | Exit from Minicom        |
          +--------------------------+
```

2. selezionando "`Serial port setup`" si è ottenuta seguente schermata:

```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/ttyS1                                |
| B - Lockfile Location     : /var/lock                                 |
| C -   Callin Program      :                                           |
| D -  Callout Program      :                                           |
| E -    Bps/Par/Bits       : 9600 8N1                                  |
| F - Hardware Flow Control : Yes                                       |
| G - Software Flow Control : No                                        |
|                                                                       |
|    Change which setting?                                              |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

3. digitando il tasto "`E`" è stata modificata la velocità di trasmissione, portandola a 115200 con l'opzione 8N1:

```
                +-----------[Comm Parameters]------------+
+---------------|                                        |--------------+
```

[10] Minicom è un programma di emulazione terminale(emula il terminale ANSI e il terminale VT102) per sistemi operativi Unix, originariamente scritto da Miquel van Smoorenburg. The reference link is: http://alioth.debian.org/projects/minicom/

```
| A -    Serial | Current: 115200 8N1                    |              |
| B - Lockfile L|                                        |              |
| C -   Callin P|   Speed          Parity          Data  |              |
| D -  Callout P|                                        |              |
| E -    Bps/Par| A: 300           L: None         S: 5  |              |
| F - Hardware F| B: 1200          M: Even         T: 6  |              |
| G - Software F| C: 2400          N: Odd          U: 7  |              |
|               | D: 4800          O: Mark         V: 8  |              |
|    Change whic| E: 9600          P: Space              |              |
+---------------| F: 19200                      Stopbits |--------------+
        | Screen| G: 38400                        W: 1   |
        | Save s| H: 57600                        X: 2   |
        | Save s| I: 115200        Q: 8-N-1               |
        | Exit  | J: 230400        R: 7-E-1               |
        | Exit f|                                         |
        +-------|                                         |
                | Choice, or <Enter> to exit?             |
                +-----------------------------------------+
```

Questo settaggio è stato possibile andando a selezionare le voci giuste attraverso la tastiera.

4. una volta impostati i giusti parametri del punto 2, si è tornati alla schermata precedente:

```
+-----------------------------------------------------------------------+
| A -    Serial Device      : /dev/ttyS1                                |
| B - Lockfile Location     : /var/lock                                 |
| C -   Callin Program      :                                           |
| D -  Callout Program      :                                           |
| E -    Bps/Par/Bits       : 115200 8N1                                |
| F - Hardware Flow Control : No                                        |
| G - Software Flow Control : No                                        |
|                                                                       |
|    Change which setting?                                              |
+-----------------------------------------------------------------------+
        | Screen and keyboard      |
        | Save setup as dfl        |
        | Save setup as..          |
        | Exit                     |
        | Exit from Minicom        |
        +--------------------------+
```

dove è stato necessario modificare il controllo del flusso hardware portandolo a `no` attraverso il tasto `F`. Accettando le modifiche effettuate si è selezionata la voce `Exit`, in questo modo si è usciti dalla modalità settaggio di minicom e si entrati in modalità ascolto dalla porta seriale. Questo lo ha dimostrato il seguente output:

```
sudo minicom

Welcome to minicom 2.5

OPTIONS: I18n
Compiled on Feb  5 2011, 06:31:35.
Port /dev/ttyS0

Press CTRL-A Z for help on special keys
```

Con questo ultimo punto è stato completato il setting di minicom e quindi si è partiti con la fase di installazione. Pertanto, inserendo la SD card nell'apposito host device della PandaBoard e collegando quest'ultima al Personal Computer, al primo avvio del dispositivo embedded si è ottenuto il seguente output.

```
Texas Instruments X-Loader 1.5.0 (Apr 11 2011 - 09:48:22)
Reading boot sector
Loading u-boot.bin from mmc

U-Boot 2011.03 (Apr 20 2011 - 07:37:43)

CPU  : OMAP4430
Board: OMAP4 Panda
I2C:   ready
DRAM:  1 GiB
MMC:   OMAP SD/MMC: 0
Using default environment

In:    serial
Out:   serial
Err:   serial
Hit any key to stop autoboot:  0
reading boot.scr

373 bytes read
Running bootscript from mmc0 ...
## Executing script at 82000000
reading uImage
```

Dal quale poi in modo automatico è partita la fase di installazione guidata.

Ad installazione completata, sempre nel terminale, in cui si è svolta l'installazione, è comparsa la richiesta di inserimento di login e password[11]; in questo modo è stata completata la fase di installazione del sistema operativo.

5.3.4 Update e Upgrade del sistema operativo

Terminata la sezione precedente, il passo successivo è stato quello di effettuare un aggiornamento[12] dell'intero sistema attraverso i seguenti comandi:

```
sudo apt-get update
sudo apt-get upgrade
```
[13]

E' stato necessario tenere presente che questi due comandi hanno necessitato di una connessione ad internet, pertanto visto che in fase di installazione non è stato effettuato il settaggio della scheda di rete è stato fatto in questa occasione.

[11] queste credenziali sono state definite nella procedura di installazione

[12] Gli aggiornamenti vengono rilasciati dalla Canonical Community mediamente ogni giorno; the reference link is: http://www.canonical.com/about-canonical/overview

[13] update e upgrate sono rispettivamente l'aggiornamento dei repository e l'istallazione delle modifiche apportate attraverso i repository.

Nei test svolti sulla schede di rete si sono riscontrati problemi[14] con il dispositivo WiFi, integrato nella PandaBoard, per tanto il settaggio della connessione internet è stato svolto sull'interfaccia ethernet (eth0).

5.3.5 Settaggio della scheda di rete ethernet

Per quanto riguarda il settaggio della scheda di rete, prima di procedere alla configurazione, è stato necessario controllare se l'interfaccia `eth0` fosse spenta. La procedura è iniziata lanciando da terminale il seguente comando:

```
ifconfig
```

attraverso il quale si è ottenuto in output il seguente risultato:

```
eth0      Link encap:Ethernet HWaddr xx:xx:xx:xx:xx:xx
          inet6 addr: xxxx::xxxx:xxxx:xxxx:xxxx/xx Scope:Link
          UP BROADCAST RUNNING MULTICAST MTU:1492 Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 MB) TX bytes:0 (0.0 KB)
```

Questo log è stato il segno che la scheda ethernet fosse accesa, pertanto si è provveduto a spegnerla con il seguente comando:

```
sudo ifconfig eth0 down
```

A questo punto si è proceduto effettivamente nella configurazione dell'interfaccia.

- Configurazione dell'indirizzo IP e della subnetmask:

  ```
  ifconfig eth0 192.168.xxx.xxx netmask 255.255.255.0
  ```

- Riattivazione dell'interfaccia di rete:

  ```
  ifconfig eth0 up
  ```

- Configurazione dell'indirizzo di Gateway[15]:

  ```
  route add default gw 192.168.xxx.xxx
  ```

[14] Si è notato che nella versione server i driver per la gestione del WiFi è come se non fossero presenti; questo non accade nella versione Desktop, nella quale il modulo wireless, nonostante sia munito di una piccola antenna riesce a coprire perfettamente una raggio di 15-20 metri.

[15] Il Gateway è un termine generico che indica il servizio di inoltro dei pacchetti verso l'esterno; il dispositivo hardware che porterà a termine questo compito è tipicamente un router. Nelle reti più semplici è presente un solo gateway che inoltra tutto il traffico diretto all'esterno verso la rete internet. In reti più complesse in cui sono presenti parecchie subnet, ognuna di queste fa riferimento ad un gateway che si occuperà di instradare il traffico dati verso le altre sottoreti o a rimbalzarlo ad altri gateway.

- Scelta del server DNS[16] a cui fare riferimento:

```
echo nameserver [IP address of DNS server] » /etc/resolv.conf
```

giunti a questo punto è stato possibile fare delle prove di connessione e appurare che effettivamente la PandaBoard fosse collega ad una rete internet; tali prove sono state effettuate lanciando dei ping ad un qualsiasi sito internet, come per esempio:

```
ping www.google.it
```

5.3.6 Installazione di un server SSH

[17] Come anticipato in uno dei paragrafi precedenti, avendo istallato una distribuzione di tipo server, è risultato molto utile avere a disposizione un server SSH installato sulla PandaBoard; questo perché sistemi del genere molto spesso sono integrati in impianti non sempre accessibili e soprattutto lontani dall'operatore; come del resto nello scenario sottomarino disegnato dal progetto NAUTILUS. Pertanto in questo paragrafo si andrà ad illustrare come è stato installato e configurato un server ssh. Il primo passo è stato, quindi, quello di installare il pacchetto dato con il seguente comando:

```
sudo apt-get install openssh-server
```

dopo l'installazione si è proceduto con la configurazione del server:

```
sudo cp /etc/ssh/sshd_config /etc/ssh/sshd_config.original
```

il comando riportato sopra è stato utile per avere una copia della configurazione originale; in questo modo si ci è assicurati che se nel caso la nostra configurazione non fosse stata corretta si sarebbero potute ripristinare le impostazioni di default. In ogni caso, procedendo come riportato di seguito si è riusciti ad effettuare una giusta configurazione del server.

```
sudo chmod a-w /etc/ssh/sshd_config
sudo nano /etc/ssh/sshd_config
```

Attraverso questi due comandi si è riusciti a modificare i diritti di scrittura sul file di configurazione mediante `chmod a-w`[18] e ad aprire un editor di testo per il

[16] Il sistema dei nomi a dominio, in inglese Domain Name System (spesso indicato con DNS), è un sistema utilizzato per la risoluzione di nomi dei nodi della rete (in inglese host) in indirizzi IP e viceversa. Il servizio è realizzato tramite un database distribuito, costituito dai server DNS.

[17] In informatica e telecomunicazioni SSH (Secure SHell, shell sicura) è un protocollo di rete che permette di stabilire una sessione remota cifrata tramite interfaccia a riga di comando con un altro host di una rete informatica. È il protocollo che ha sostituito l'analogo ma insicuro Telnet.

[18] Con il comando `chmod a-w` si è fatto in modo che il file di configurazione non venisse scritto dall'utente: user,group e other

settaggio dei parametri di configurazione. Per personalizzare la porta di accesso al server ssh è stato modificata la riga:

```
Port 22
```

in

```
Port 222
```

questa modifica è stato un piccolo contributo alla security system. Per consentire che il server ssh utilizzasse una chiave pubblica, è stata modificata la relativa voce nel seguente modo:

```
PubkeyAuthentication yes
```

In aggiunta alla chiave pubblica infine è stata abilitata la richiesta di password per accedere al servizio Security Shell:

```
PasswordAuthentication yes
```

Infine per far si che il server SSH, appena un utente si connetta da remoto, mostri un banner di pre-login contenuto del file /etc/issue.net, è stata aggiunta la seguente riga di configurazione:

```
Banner /etc/issue.net
```

Giunti a questo punto è stata completata la configurazione; per renderla funzionante, dopo aver salvato il file /etc/ssh/sshd_config è stato lanciato il seguente comando:

```
sudo /etc/init.d/ssh restart
```

Per quanto riguarda il test della configurazione appena eseguita è bastato collegare con un cavo ethernet incrociato[19] la PandaBoard e il personal computer usato fino a questo punto come supporto alle diverse operazioni. Utilizzando un terminale è stato eseguito il seguente comando:

```
ssh -p 222 loginPandaBoard@ipaddress
```

dove `loginPandaBoard` è il nome utente scelto nelle fasi di installazione e `ipaddress` invece è l'indirizzo IP settato nel sotto-paragrafo 5.3.5.

[19] Un Cavo incrociato ethernet o crossover è un tipo di cavo di rete usato per connettere assieme dei dispositivi di computer direttamente dove invece vengono normalmente connessi mediante uno switch di rete, hub o router. Per esempio, si può utilizzare un cavo crossover per connettere direttamente due personal computer mediante i loro adattatori di rete. I connettori normalmente utilizzati prendono il nome di RJ-45

5.4 Ottimizzazione di NS2/NS-miracle per reti sottomarine

In questo paragrafo si procede, in un primo momento, con l'esposizione di come è stato installato per intero tutto il pacchetto NS/NS-miracle e poi, in una fase successiva, l'ottimizzazione volta alle reti sottomarine. Un'altra cosa importante da considerare è che tutte le operazioni svolte si riferiscono al sistema operativo presente sulla PandaBoard; pertanto si rimarca il fatto che non è stata usata nessuna cross-compilazione per generare gli eseguibili di NS2.

5.4.1 Installazione di NS2/NS-miracle sulla PandaBoard

I requisiti necessari per la compilazione ed esecuzione di NS2 sono stati risolti installando le seguenti dipendenze:

```
build-essential; make; tcl8.5-dev; tk8.5-dev; autoconf; automake;
libxmu-dev;
xorg-dev; g++; xgraph;
```

a tal proposito, come già visto in precedenza, si è proceduto nel seguente modo da terminale:

```
sudo apt-get install build-essential; make; tcl8.5-dev; tk8.5-dev; autoconf; automake;
libxmu-dev;
sudo apt-get install xorg-dev; g++; xgraph;
```

Per quanto riguarda i sorgenti di NS2, si è scaricato dal seguente link:

http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.34/

la versione 2.34 all-in-one. Questa versione contiene tutti i moduli di supporto all'engine di NS2 ed un unico file di compilazione in modo che, con una sola operazione, il simulatore di rete viene compilato e installato. A questo punto, per scelta di progetto, si è pensato di organizzare l'albero delle cartelle in modo da tenere ben separati gli eseguibili di NS2, le librerie di NS-Miracle e i moduli DESERT, pertanto si è definita la seguente suddivisione:

```
/home/uw/.ns2
          |
          +-- DESERT
          |
          +-- ns-allinone-2.34
          |
          +-- ns-miracle
```

In questo modo è stato decompresso il file `ns-allinone-2.34.tar.gz` nella relativa cartella e prima di passare all'effettiva installazione si sono dovute apportare delle modifiche al codice sorgente in quanto alcune *chiamate a funzioni* sono risultate non compatibili con il compilatore. Detto ciò le modifiche effettuate sono state le seguenti:

- nel file `configure` contenuto nella cartella `otcl-1.13` alla riga 6304 è stato sostituita la riga:

  ```
  SHLIB_LD=gcc -shared
  ```

  con

  ```
  SHLIB_LD=ld -shared
  ```

- al file `ranvar.cc` contenuto nella cartella `ns-2.34/tools` alla riga 219 è stata modificata la firma della funzione:

  ```
  return GammaRandomVariable::GammaRandomVariable(1.0 + alpha_,
  beta_).value() * pow (u, 1.0 / alpha_);
  ```

  con la seguente

  ```
  return GammaRandomVariable(1.0 + alpha_, beta_).value()
  * pow (u, 1.0 / alpha_);
  ```

- nel file, raggiungibile da questo percorso `.ns2/ns-allinone-2.34/ns-2.34 /mobile/nakagami.cc` le righe di codice 183 e 185 sono state modificate rispettivamente in:

  ```
  resultPower = ErlangRandomVariable(Pr/m, int_m).value();
  ```

  e

  ```
  resultPower = GammaRandomVariable(m, Pr/m).value();
  ```

- infine, l'ultima modifica è stata apportata al file `mac-802_11Ext.h` contenuto nella cartella `ns-2.34/mac/` alla riga 64 con l'aggiunta dell'inclusione dell'header *stddef.h*

  ```
  #include <stddef.h>
  ```

A questo punto dalla cartella `.ns2/ns-allinone-2.34` è stato possibile lanciare l'installazione attraverso il seguente comando:

```
./install
```

Dopo aver completato l'installazione è stato fatto un primo test sulla funzionalità del simulatore, di fatto seguendo il percorso `.ns2/ns-allinone-2.34/ns-2.34` e lanciando il comando:

```
./ns
```

è stato possibile appurare che NS2 è stato installato correttamente; infatti visualizzato il Prompt dei comandi di ns è stata fatta partire una simulazione di prova, che è andata naturalmente a buon fine.

A questo punto per richiamare il simulatore di rete da ogni punto dell'albero di sistema è stato necessario aggiungere le seguenti righe di configurazione al file `.bashrc`[20] contenuto in `/home/uw`:

```
# LD_LIBRARY_PATH
OTCL_LIB=/home/UTENTE/.ns2/ns-allinone-2.34/otcl-1.13
NS2_LIB=/home/UTENTE/.ns2/ns-allinone-2.34/lib
USR_LOCAL_LIB=/usr/local/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$USR_LOCAL_LIB

# TCL_LIBRARY
TCL_LIB=/home/UTENTE/.ns2/ns-allinone-2.34/tcl8.4.18/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB

# PATH
XGRAPH=/home/UTENTE/.ns2/ns-allinone-2.34/bin:/home/UTENTE/.ns2/ns-allinone-2.34/
        tcl8.4.18/unix:/home/UTENTE/.ns2/ns-allinone-2.34/tk8.4.18/unix
NS=/home/UTENTE/.ns2/ns-allinone-2.34/ns-2.34/
NAM=/home/UTENET/.ns2/ns-allinone-2.34/nam-1.14/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

Per rendere queste configurazioni attive fin da subito, evitando di riavviare il dispositivo embedded, è bastato lanciare il seguente comando:

```
source  /.bashrc
```

Procedendo con l'installazione di NS-Miracle, sempre in modalità non ottimizzata, si è proseguito nel seguente modo:

- Ci si è posizionati nella sotto directory `/.ns2/ns-miracle` ed è stato fatto partire il download di NS-Miracle direttamente dal server del dipartimento attraverso svn[21]:

  ```
  svn co --username nsmiracle-dev-guest
         --password nsmiracleguest
         https://telecom.dei.unipd.it:/tlcrepos/nsmiracle-dev/trunk}
  ```

- una volta scaricato l'intero pacchetto è stata completata la procedura di installazione nel seguente modo:

  ```
  ./autogen.sh
  ./configure --with-ns-allinone=/home/UTENTE/.ns2/ns-allinone-2.34
              --prefix=/home/UTENTE/.ns2/ns-allinone-2.34
  ```

[20] E' un file di configurazione locale che contiene direttive di configurazione come gli alias, o funzioni definite dall'utente. Il file viene letto ed eseguito successivamente a quelli globali, gli alias o le funzioni saranno specifici dell'utente e non influenzeranno nessun altro utente. E' il corrispondente locale di /etc/bashrc.

[21] Subversion (noto anche come svn, che è il nome del suo client a riga di comando) è un sistema di controllo versione progettato da CollabNet Inc. con lo scopo di essere il naturale successore di CVS, oramai considerato superato.

```
            --disable-static
            --with-dei80211mr=/home/UTENTE/.ns2/ns-allinone-2.34/dei80211mr-1.1.4
make
sudo make install
```

A questo punto sono state fatte nuovamente delle prove[22] per testare che tutto funzionasse nel modo corretto. Queste prove si sono svolte utilizzando anche le librerie DESERT, di cui è stato discusso nel paragrafo 2.3.

Appurato che tutto funzionasse nel migliore dei modi si è passati alla fase di ottimizzazione; questa fase di studio è iniziata con lo studio dei manuali di ns2 in cui si sono rilevati, effettivamente, pacchetti non necessari alla prototipazione di reti acustiche sottomarine. La lista dei moduli contenuti in *ns-allinone* è la seguente:

- Tcl (release 8.5.10)
- Tk (release 8.5.10)
- Otcl (release 1.14)
- TclCL (release 1.20)
- Ns (release 2.35)
- Nam (release 1.15)
- Xgraph (version 12)
- CWeb (version 3.4g)
- SGB (version 1.0)
- Gt-itm
- sgb2ns 1.1
- Zlib (version 1.2.3)
- dei80211mr-1.1.4

Analizzando tali moduli si è valutato che buona parte di questi componenti è risultata non necessaria alle trasmissioni sottomarine; di fatto le seguenti estensioni non sono state compilate:

Nam. Questo modulo è uno strumento di animazione basato su Tcl e Tk. In sostanza questo tool, siccome supporta il layout topologico della rete, l'animazione dei pacchetti e vari strumenti di controllo dati, è utilizzato per la visualizzazione delle tracce di simulazione. Lo sviluppo di Nam è partito da LBL con il forte contributo dal progetto VINT.

[22] I test effettuati, in questo punto dell'elaborato, hanno previsto un setting di sistema in cui la Pandaboard è collegata al micromodem; quindi un test in cui il canale subacqueo è quello reale

XGraph. Tale routine è un programma di plotting a supporto di NS2 attraverso il quale è possibile creare rappresentazioni grafiche dei risultati delle simulazioni utilizzando i file *.tr* generati come output dei file tcl

Cweb. La filosofia alla base di questo modulo è quella di dare ai programmatori di NS2, che vogliono fornire un'ottima documentazione per i loro programmi, un tool in grado di combinare la formattazione LaTeXcon il linguaggio **C** per la programmazione. Questo perché nessuno dei due tipi di linguaggio è in grado di fornire da solo la migliore documentazione per l'ambiente NS2.

SGB. (Stanford GraphBase) contiene i sorgenti per la compilazione della libreria sgblib necessarie all'utilizzo di GT-ITM

Gt-itm. Questo è un insieme di routine per generare e analizzare grafici usando un'ampia varietà di modelli per la topologia internetwork. I grafici sono generati in formato SGB Don Knuth; ma grazie ad una routine, fornita anch'essa con questo set di applicazioni, è possibile convertire i grafici di cui si parlava sopra in un formato alternativo più facile da analizzare.

sgb2ns. Questo programma è usato per convertire l'output del generatore di topologia GT-ITM nel formato ns2; Praticamente è possibile creare graficamente la topologia della rete che si vogliono simulare attraverso le routine contenute nella GT-ITM e poi convertirle nel formato ns2 per la simulazione effettiva.

Zlib. Zlib è una libreria general purpose per la compressione dei dati; il set di funzioni messe a disposizione da Zlib possono essere usate direttamente nei file Tcl per la manipolazione di file compressi.

dei80211mr-1.1.4 La libreria dei80211mr offre un'implementazione del protocollo 802.11 per NS2. Questa implementazione deriva dal modulo 802.11 incluso già nella versione 2.29 del Simulatore di Rete (NS) e mira a risolvere alcuni bug noti, oltre a fornire funzionalità avanzate. La libreria dei80211mr utilizza la patch *Dynamic Lybrary*; grazie alle funzionalità introdotte da questa patch, dei80211mr può essere utilizzato con diverse versioni di ns2. Di fatto tale libreria e stata testata con ns-2.29, ns-2.31 e ns-2.34.

Per quanto riguarda invece dei sorgenti contenuti in: Tcl, Tk, Otcl, TclCL ed Ns è stato fatto un lavoro di ottimizzazione diverso. In pratica, dato che questi cinque componenti sono effettivamente il motore di tutto il simulatore, è necessario procedere con un'ottimizzazione mirata; nel senso che prima di compilare i sorgenti è stato necessario analizzare i codici e capire quali routine non sono necessarie per simulare trasmissioni in scenari sottomarini. La necessità di effettuare dei tagli sul codice prima della compilazione comporta il vantaggio che

nelle librerie e negli eseguibili generati dalla compilazione non vengono inglobate delle funzioni che inevitabilmente renderebbero il file più pesante dal punto di vista dello spazio occupato. Di contro è stato necessario prestare molta attenzione a ciò che è stato eliminato e provvedere alla modifica dei codici sorgente che nativamente prevedono degli include a librerie o sorgenti non più esistenti. Dopo la compilazione dei sorgenti, un ulteriore step di ottimizzazione dal punto di vista dello spazio occupato è stato quello di ripulire le cartelle di ns-allinone da tutti i file sorgenti; in questa fase sono state eliminati anche molti altri file e cartelle di supporto come documentazioni ai sorgenti e file di tipo evaluation-test per l'analisi di una corretta installazione. Cosa molto importante per questa operazione è stata l'analisi dei vari Makefile[23] utilizzati per la compilazione.

Per quanto riguarda la procedura di ottimizzazione per il modulo Ns-Miracle è stata utilizzata la stessa procedure vista sopra; di fatto dall'analisi di questo modulo sono state riscontrate le seguenti funzionalità:

- applicazione CBR[24]
- una routine Cross-layer tracers, utilizzata per tracciare il flusso dati tra i diversi layer del simulatore di rete.
- funzionalità per il protocollo IP (ad esempio la possibilità di fare routing).
- un wrapper per il modulo TCP del simulatore di rete.
- una routine Miracle PHY per l'implementazione di tecnologie wireless a livello fisico.
- una libreria per il multiplexer delle porte.

[23] L'utility Make è usata soprattutto per la compilazione di codice sorgente in codice oggetto, unendo e poi linkando il codice oggetto in programmi eseguibili o in librerie. Esso usa file chiamati makefile per determinare il grafo delle dipendenze per un particolare output, e gli script necessari per la compilazione da passare alla shell. Pertanto un makefile consiste in linee di testo che definiscono un file (o un gruppo di file) oppure il nome di una regola dipendente dal gruppo di file.

[24] La categoria di servizio CBR viene utilizzato per le connessioni in cui il traffico dati ha un bit-rate costante e dove c'è la necessità di avere una sincronizzazione del traffico in termini di tempo tra sorgente e destinatario. Queste applicazioni includono servizi come la videoconferenza, la telefonia (servizi voce) o qualsiasi tipo di servizio on-demand, come la voce interattiva e audio. Per le applicazioni di telefonia vocale CBR offre bassa latenza del traffico e caratteristiche di erogazione prevedibili; per tanto in genere il servizio CBR è utilizzato per l'emulazione di reti.

- un set di librerie per la gestione dei moduli: UMTS[25], MAC[26] e RLC[27]
- un wrapper per il modulo 802.11 come nel caso del protocollo TCP.
- un insieme di strumenti per le librerie dei80211mr usare in nsmiracle.
- e infine un set di modelli per la gestione del flusso dati (ad esempio come il Gauss-Markov mobility).

Pertanto come nel caso precedente, considerando trasmissioni di tipo sottomarine, tutte le routine utilizzate per il protocollo 802.11 non sono state compilate.

In definitiva valutando l'ottimizzazione in termini percentuali si può dire che l'operazione esposta in questo paragrafo ha portato una compressione del pacchetto NS2,Ns-miracle di circa il 20%. Si tiene a precisare che questa procedura di ottimizzazione non è stata molto spinta in quanto, utilizzando per la PandaBoard una Secure Digital di tipo SDHC da 4 Gb, si è pensato di concentrare il lavoro di questa tesi su altri aspetti. Ciò nonostante, attraverso questa attività si è potuta valutare l'effettiva possibilità di ottimizzare il pacchetto NS2-NsMiracle-DESERT con vincoli più stringenti; in effetti come ripreso del capitolo 7 dato che si pensa di testare anche dispositivi embedded con prestazioni inferiori alla PandaBoard l'idea di avere un pacchetto che occupa il meno possibile potrebbe essere uno degli aspetti principali; pertanto, avere un simulatore di rete ridotto al minimo indispensabile, potrebbe risultare tra le attività future del progetto Nautilus.

5.5 Installazione delle librerie di DESERT Underwater

L'installazione delle librerie DESERT Underwater è fondamentalmente simile alle procedure viste per l'installazione di NS-miracle; di fatto anche se le librerie DESERT sono composte da diversi moduli compilabili in modo separato si è

[25] Universal Mobile Telecommunications System, è uno standard di telefonia mobile cellulare 3G, evoluzione del GSM. Tale tecnologia ha la peculiarità di impiegare lo standard base W-CDMA più evoluto come interfaccia di trasmissione nell'accesso radio al sistema, è compatibile con lo standard 3GPP e rappresenta la risposta europea al sistema ITU di telefonia cellulare 3G.

[26] In telecomunicazioni nell'ambito delle reti di calcolatori MAC (acronimo di Media Access Control) è il nome di uno strato del modello architetturale standardizzato ISO-OSI, definito nello standard IEEE 802 e che contiene funzionalità di controllo dell'accesso al mezzo fisico per canali broadcast, funzionalità di framing e controllo di errore. Fa parte del livello datalink di cui rappresenta il sottolivello inferiore sovrastato dal sottolivello LLC e limitato inferiormente dal livello fisico

[27] Radio Link Protocol (RLP) è una frammentazione del protocollo ARQ utilizzato su una rete wireless (tipicamente cellulare). Diverse interfacce wireless sono settate in modo da avere una perdita di pacchetti dell'1%. Tuttavia, la perdita di pacchetti fissata all'1% è intollerabile per tutte le varianti di TCP; pertanto in questo caso è necessario migliorare l'affidabilità per le reti che trasportano i dati in modalità TCP/IP.

pensato anche ah una soluzione "*all in one*" in modo da avere un unico file capace di garantire la compilazione completa di tutti i sorgenti in un unico step. Pertanto posizionandoci nella cartella DESERT è stato preso in considerazione il file autogen e da terminale è stato lanciato il seguente comando:

```
./autogen.sh
```
[28]

fatto ciò, il comando successivo è stato quello di configurazione per la creazione del makefile:

```
./configure --with-ns-allinone=<ns2-allinone_path>
            --with-nsmiracle=<ns-miracle_path>
            --prefix=<path_where_libraries_will_be_installed>
```

5.6 Benchmark della RAM

Il controllo della RAM, utilizzata durante le simulazioni, è affidato ad un piccolo programma scritto in linguaggio **C++**. Tale programma, che va sotto il nome di **infoRAM**, prevede una interfaccia grafica di tipo testuale[29] direttamente dal terminale in cui si lancia il programma. A tal proposito per effettuare i test di benchmark sono stati eseguiti i seguenti passi:

- sono stati aperti due terminali; uno destinato al programma infoRAM e l'altro invece utilizzato per eseguire le simulazioni in NS2
- per utilizzare il programma di banchmarck è stato necessario entra nella cartella del programma e lanciare l'eseguibile con il seguente comando

  ```
  ./infoRAM
  ```

  in questo modo, nello stesso terminale, si avvia l'interfaccia utente per la gestione del programma. La prima richiesta del programma è stata:

  ```
  Write the name of the process that you want to monitor:
  ```

  come in Figura 5.1

  pertanto prima di scrivere il nome del processo da monitorare (in questo caso è risultato essere **ns**) è stato necessario far partire il simulatore di rete.

- quindi a questo punto nell'altro terminale è stato lanciato il seguente comando:

[28] autogen.sh fornisce automaticamente il sistema di compilazione è generalmente molto utile per progetti che utilizzano la GNU build sistem(cioè lo GNU autotools: autoconf, automake, e libtool).

[29] l'interfaccia grafica è garantita grazie all'utilizzo delle librerie *ncurses*

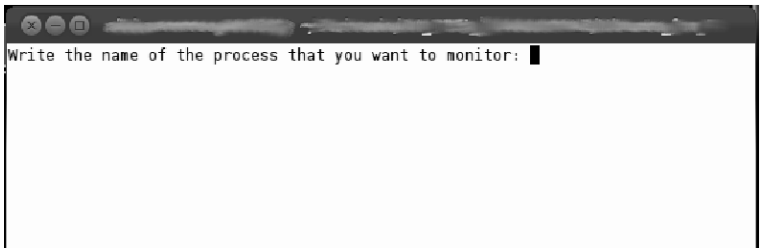


Figura 5.1: In questa figura viene mostrata la prima schermata del programma infoRAM in cui è richiesto il nome del processo da monitorare: nel caso in questione è stato digitato "ns"

```
ns
```

in questo modo è comparso il prompt dei comandi del simulatore di rete.

- dopo che il simulatore di rete è stato messo in esecuzione si è tornati sull'interfaccia utente del programma di benchmark ed è stato digitato il nome del processo; dando l'invio a questa richiesta del programma è iniziato il monitoraggio della ram utilizzata del simulatore di rete, come in Figura 5.2.

- a questo punto per monitorare effettivamente una simulazione si è ritornati al prompt dei comandi di ns ed è stata lanciata la prima simulazione attraverso il comando:

```
source <nome-simulazione>.tcl
```

- completata l'esecuzione dello script .tcl è stato bloccato anche il monitoraggio della ram. In questa fase il programma restituisce un file di output con il nome di *log-infoRAM.txt* in cui sono stati riportati tutti i valori della ram rilevati durante l'analisi.

Una nota molto importante per l'effettiva esecuzione del benchmark è stata quella di inserire delle righe di codice nei file .tcl in modo da avere a fine simulazione un file di log, chiamato *log-infoRAM.txt*, contenete tutte le operazioni monitorate e i corrispondenti tempi in cui sono stati eseguite. In Figura 5.3 viene riportato un grafico per mostrare effettivamente come evolve il consumo di ram durante l'esecuzione delle simulazioni.

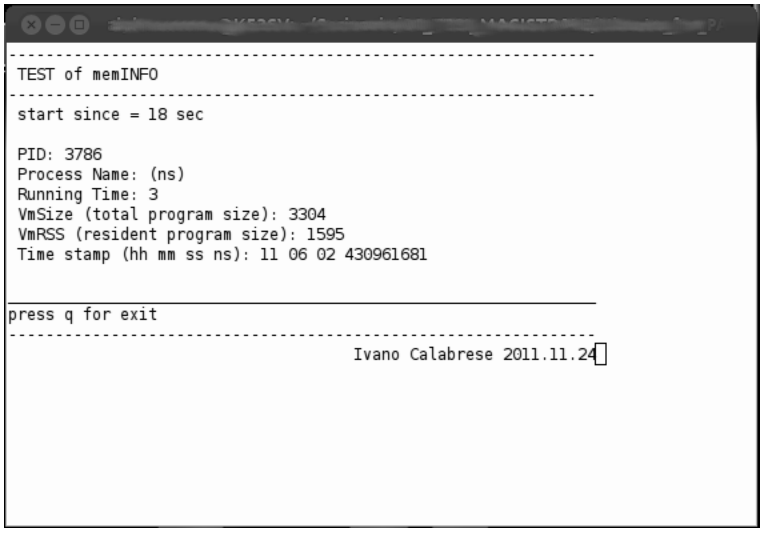


Figura 5.2: In questa figura viene mostrata la schermata del programma infoRAM in cui vengono riportati tutti i dati di controllo. Le voci di fatto più importanti sono: "VmSize" e "VmRSS". VmSize è la quantità totale di memoria richiesta dal programma monitorato e VmRSS è la "Resident Set Size" cioè l'utilizzo effettivamente della memoria all'atto del monitoraggio. Questo perché parte della memoria ram potrebbe essere trascritta su memoria Flash

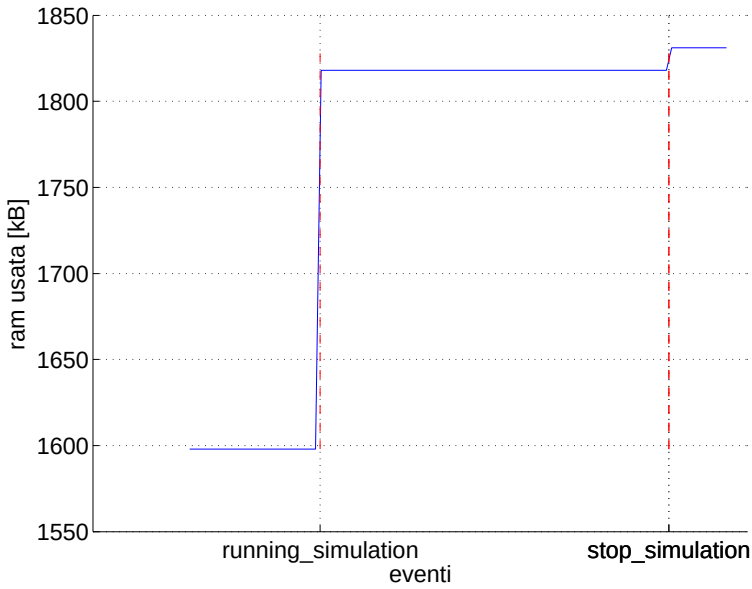


Figura 5.3: In figura viene riportato un andamento del monitoraggio della ram

Capitolo 6

Test preliminari

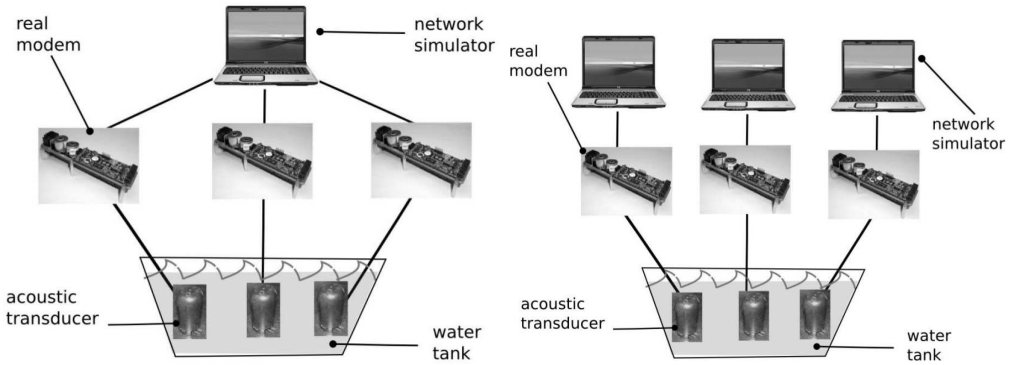
In questo penultimo capitolo saranno illustrati i test effettuati sia durante le varie fasi di sviluppo in laboratorio di questo progetto , sia quelli realizzati direttamente in loco in Piovego e a La Spezia nella parte conclusiva dei lavori. Nel primo paragrafo sarà presendata l'interfaccia *NS-Miracle - PandaBoard*; tale software ha permesso al simulatore di rete NS2 di interagire con i micro-modem. Il paragrafo successivo, invece, descriverà i due SetUp utilizzati per i test fuori porta: Piovego e La Spezia. Infine, negli ultimi due paragrafi saranno illutrati gli scenari che abbiamo testato, i problemi che si sono riscontrati e gli accorgimenti da prendere eventualmente nei test successivi.

6.1 SetUp emulativo/test-bed

Le prove di trasmissione effettuate hanno previsto un "setUp emulativo" e un "setUp test-bed"; siccome di questi due aspetti se ne è parlato già nel paragrafo 2.3 si preferisce passare direttamente ad una descrizione dello scenario costruito per i test. Il setting delle prove ha previsto i seguenti dispositivi:

- Un personal computer per la gestione dei diversi processi NS2
- tre micro-modem (descritti nel paragrafo 3.2)
- tre cavi a tenuta stagno (di lunghezza 17m circa) utilizzati per connettere i micro-modem ai trasduttori immersi in acqua
- tre trasduttori BTech's Model BT-2RCL (descritti nel paragrafo **??**)
- dei tubi in PVC per garantire un'immersione più controllata dei trasduttori
- tre alimentatori da 12V ciascuno per il funzionamento dei micro-modem
- e in fine tre adattatori USB/RS232 per la comunicazione tra PC e micro-modem

Riportando un'immagine (Figura 6.1) già introdotta in uno dei capitolo precedenti si può illustrare lo schema dei setting adoperati.



(a) Emulation setting (b) Test-Bed setting

Figura 6.1: La a sinistra rappresenta un EMULATION setting in cui una solo istanza NS controlla i tre modem. Nella figura a destra invece abbiamo la rappresentazione di un TEST-BED setting in cui i tre modem sono controllata da istanze di NS indipendenti

6.2 Prove in Piovego

I primi test in campo sono stati quelli effettuati lungo la riva del Piovego[1]; queste prove sono state efettuati fondamentalmente come test preliminari per i test effettuati in mare a largo dell'arsenale di La Spezia.

Le prove effettuate sulla riva del Piovego, anche se prevedevano l'utilizzo di tre nodi, sono state effettuate con solo due micro-modem in quanto uno dei tre ha smesso di funzionare improvvisamente e non si è potuto rimediare in tempi sufficienti brevi. Di fatto le prove svolte in questa prima fase hanno dato la possibilità di effettuare una semplice caratterizzazione del canale; ciò nonostante, essendo stata la prima prova su campo, ha contribuito in positivo ad acquisire nuove conoscenze sull'aspetto organizzativo-logistico delle prove.

La caratterizzazione del canale è stata effettuata su una distanza massima di 23m in passi da 3m; ad ogni step sono stati inviati in sequenza 50 mini-packet[2] in modo da avere una discreta affidabilià sulla statistica riporta in Figura 6.4. Si tiene a precisare, inoltre, che i trasmettitori sono stati posti a circa 2m dalla riva e ad una profondità di circa un metro.

[1] Il Piòvego è un canale artificiale emissario del Bacchiglione, lungo 10,17 km; nei secoli passati fu un'importante via di comunicazione tra Padova e Venezia. Il nome deriva dal latino publicum, ed indica quindi un canale pubblico utilizzato per la navigazione.

[2] Per mini-packet si intende un pacchetto in cui il payload è di solo 13bit

Figura 6.2: In questa immagine viene riportata un'immagine aerea del sito in cui sono stati effettuati i test preliminari. L'area utilizzata per i test è quella riportata nel riquadro rosso.

A posteriori, per fruttare a pieno i dati ricavati dai test, si è pensato di analizzare il tempo impiegato dal simulatore da quando genera il mini-packet a quando viene spedito all'intefaccia tra micro-modem e NS2; a tal proposito si analizzi la Figura 6.5

6.3 Prove a La Spezia

Nei test effettuati a La Spezia si è riusciti ad operare con i tre micro-modem in quanto in fase di valutazione del mal funzionamento del dispositivo non utilizzato, nei test sul Piovego, si è riusciti ad individuare la causa della rottura del dispositivo; pertanto, riuscendo a sostituire il componente rotto, i test effettuati in mare sono stati caratterizzati dall'uso di tre nodi.

Il luogo dei test effettuati in mare è mostrato in Figura 6.7; i test si sono svolti a bordo di una imbarcazione (un rimorchiatore di costruzione americana, mostrato in Figura 6.8) messa a disposizione da un gruppo di ricerca dell'azienda WASS[3].

I test si sono divisi in due fasi: nella prima è stata ripetuta la caratterizzazione del canale come nei test fatti nel Piovego, con la differenza che in questo

[3]Whitehead Alenia Sistemi Subacquei (WASS) è azienda leader a livello mondiale nel settore dei Sistemi Subacquei, riconosciuta per la sua eccellenza nell'Ingegneria dei Sistemi Integrati.

caso si sono scelti solo tre punti date le dimensioni ridotte dell'imbarcazione; nella Figura 6.9 viene riporta la vista laterale del Manning[4] e le relative posizioni dei nodi. I tre link sono stati testati in entrambe le direzioni, questo approccio ha messo in risalto come le comunicazioni sottomarine possano essere influenzate, oltre che dalla distanza, anche dal percorso che esse effettuano (si osservi la Figura 6.10a). Volendo in questo caso particolare dare una giustificazione, si può asserire dicendo che la causa di tale fenomeno è da attribuire alla scafo della barca (molto vicine ai trasduttori) unita probabilmente al moto delle onde che presentavano un movimento da poppa verso prua. Per una lettura più chiara dei grafici, con la lettera "a" il verso di percorrenza dal nodo più vicino a poppa al nodo verso prua[5]; nel vero contrario il collegamento è identificato con la lettera "b".

Nella seconda fase del test è stato introdotto il protocollo ALOHA-puro[6] e il protocollo ALOHA-CSMA[7]. In questi test si è usata solo la configurazione del link 2 (pertanto con i due nodi ad una distanza di 5.7m). Le tracce ricavate da questo test hanno permesso, a posteriori, di fare un'analisi della Packet Error Rate dei due protocolli di comunicazione (Figura 6.10b) e di ricavare il throughput dei pacchetti ricevuti dai nodi (Figura 6.10c). Il parametro variabile di questi test è stato il periodo con cui si spedivano i pacchetti (ogni 4sec, 6sec e 8sec per un totale di 50 pacchetti)

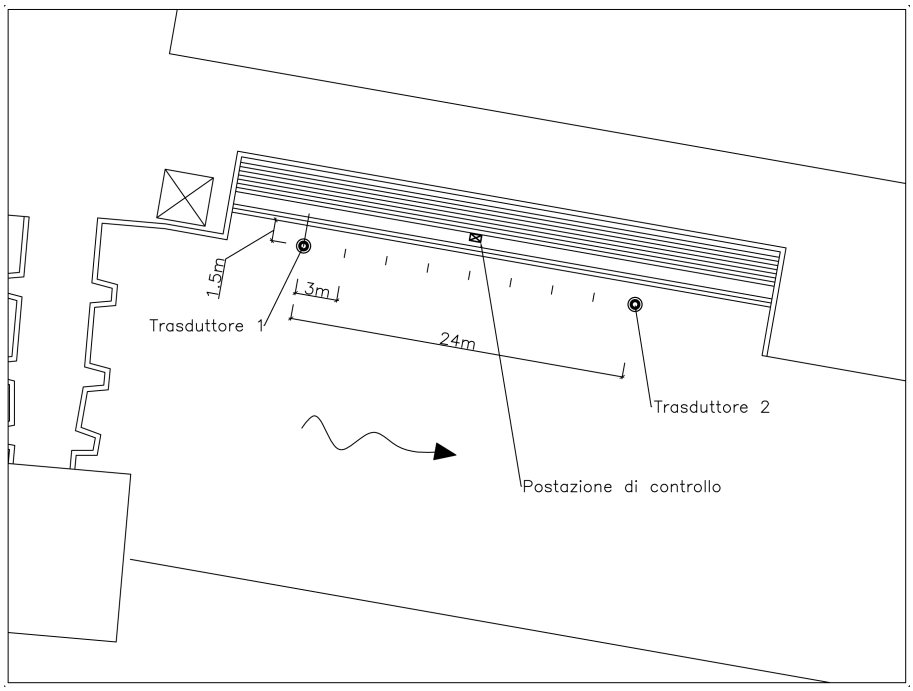
Grazie ad un idrofono si è potuto infine registrare le tracce audio di alcune trasmissioni fatte durante i test; questa analisi è stata fatto solo a titolo informativo in quanto per gli argomenti trattati in questo elaborato non avrebbero apportato informazioni rilevanti. In Figura 6.11 viene riportata una traccia che rappresenta lo spettrogramma della registrazione effettuata durante una trasmissione di pacchetti.

[4]Manning è il nome dell'imbarcazione messa a nostra disposizione per i test in mare (FIG.6.8)

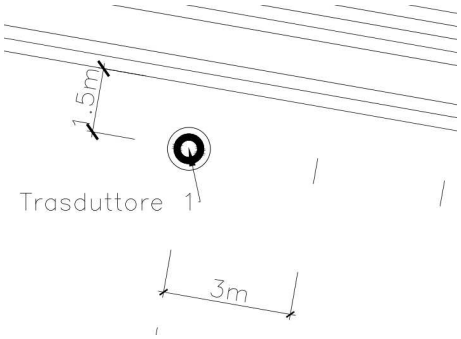
[5]Ad asempio il una trasmissione dal trasduttore 1 al trasduttore 2 è un collegamento identificato con la lettera "a"

[6]Il protocollo ALOHA puro (pure Aloha o semplicemente ALOHA), non prevede vincoli all'invio di dati e quindi all'occupazione della banda. Quando una postazione ha dati da trasmettere, li trasmette. Poiché ogni stazione agisce indipendentemente dalle altre, il successo è determinato unicamente dalla mancata collisione con altre trasmissioni da parte di altre stazioni.

[7]Nelle telecomunicazioni CSMA, (acronimo inglese di Carrier Sense Multiple Access traducibile come: protocollo ad accesso multiplo con rilevamento della portante) indica una tecnica di trasmissione dati che si basa sull'accesso multiplo tramite rilevamento della portante. Il protocollo implementa la direttiva: Ascolta prima di trasmettere. Se trovi il canale occupato aspetta e riprova più tardi secondo una modalità di ritrasmissione stabilita.



(a) visione in pianta



(b) particolare del trasmettitore 1

(c) particolare del banco di lavoro

Figura 6.3: In questa immagine viene riportata un'immagine in pianta del sito in cui sono stati effettuati i test preliminari.

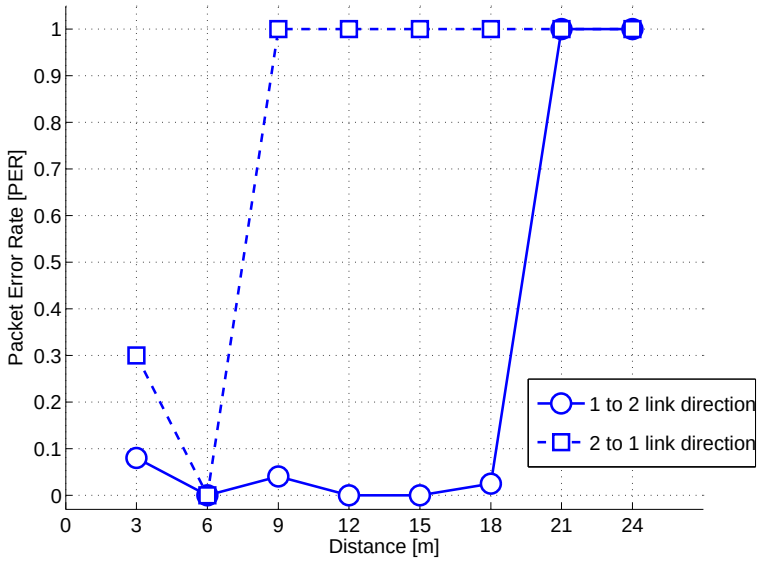


Figura 6.4: Nella presente figura viene riportata la statistica del PER in funzione della distanza tra trasmettitore e ricevitore. Le trasmissioni sono state effettuate in entrambi i versi è questo ha esaltato la forte dipendenza di quest'ultime in condizione di bassi fondali (very shallow-water)

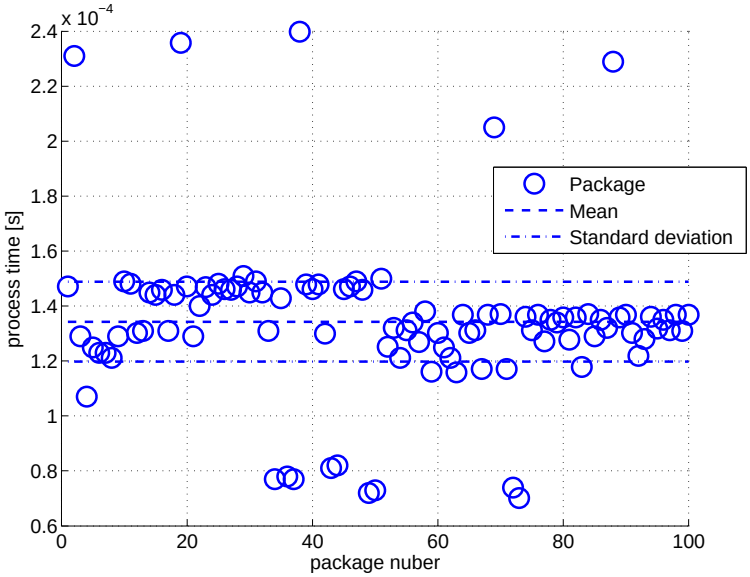
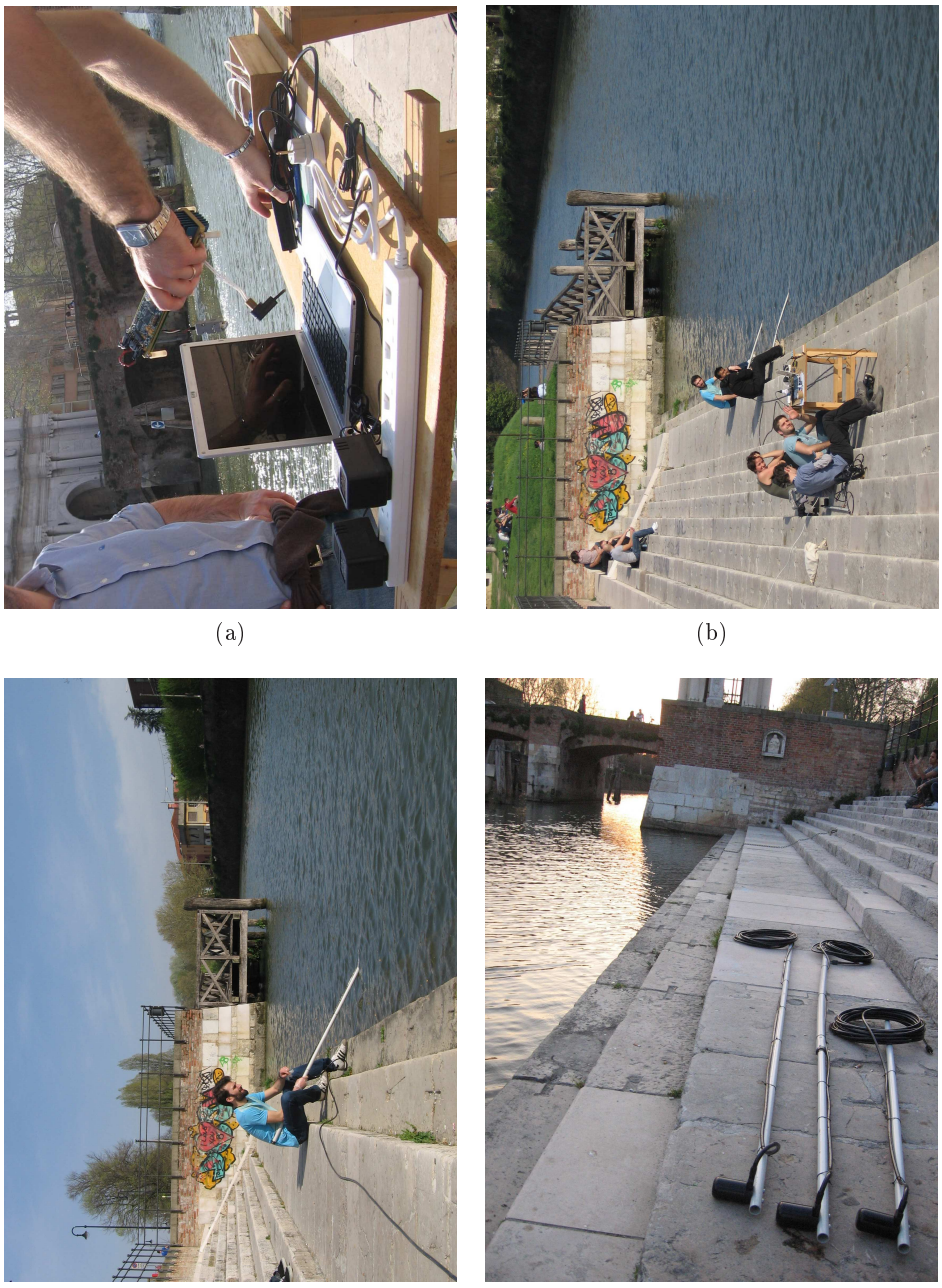


Figura 6.5: Nel seguente grafico viene riportato il tempo di processamento da quando il simulatore genera il pacchetto a quando viene inviato al software di interfacciamento con il micro-modem



(a) (b)

(c) (d)

Figura 6.6: Foto dei test sulla riva del Piovego.

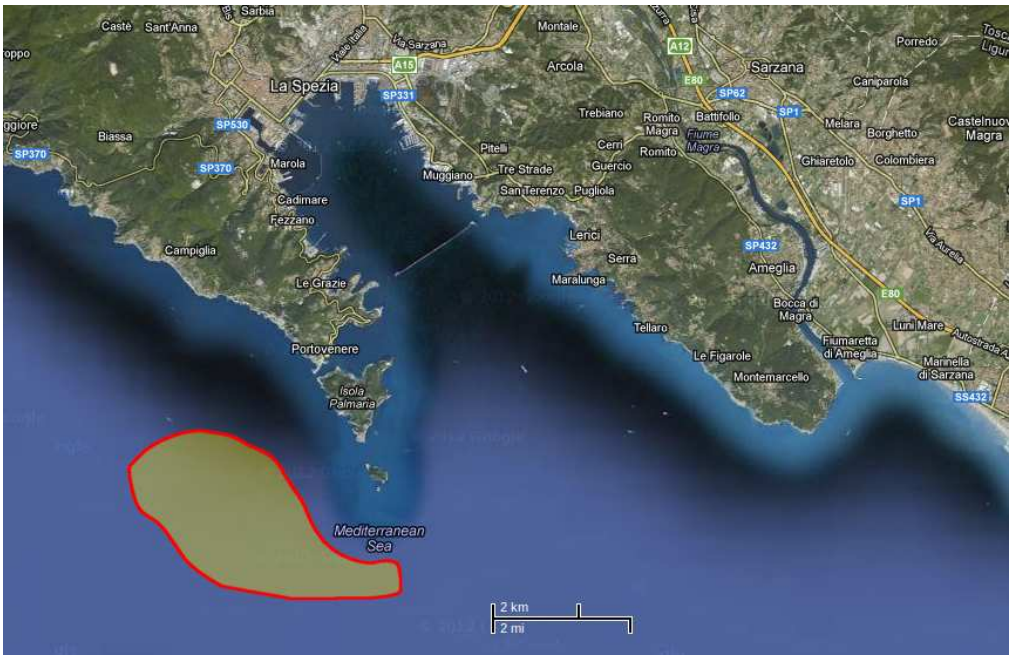


Figura 6.7: In questa immagine viene riportata una foto aerea del luogo dove sono stati effettuati i test a La Spezia; l'area dei test è contenuta nel contorno rosso, tale zona si trova a pochi miglia dall'arsenale navale.

Figura 6.8: La foto illustra l'imbarcazione usata nei test effettuati a La Spezia

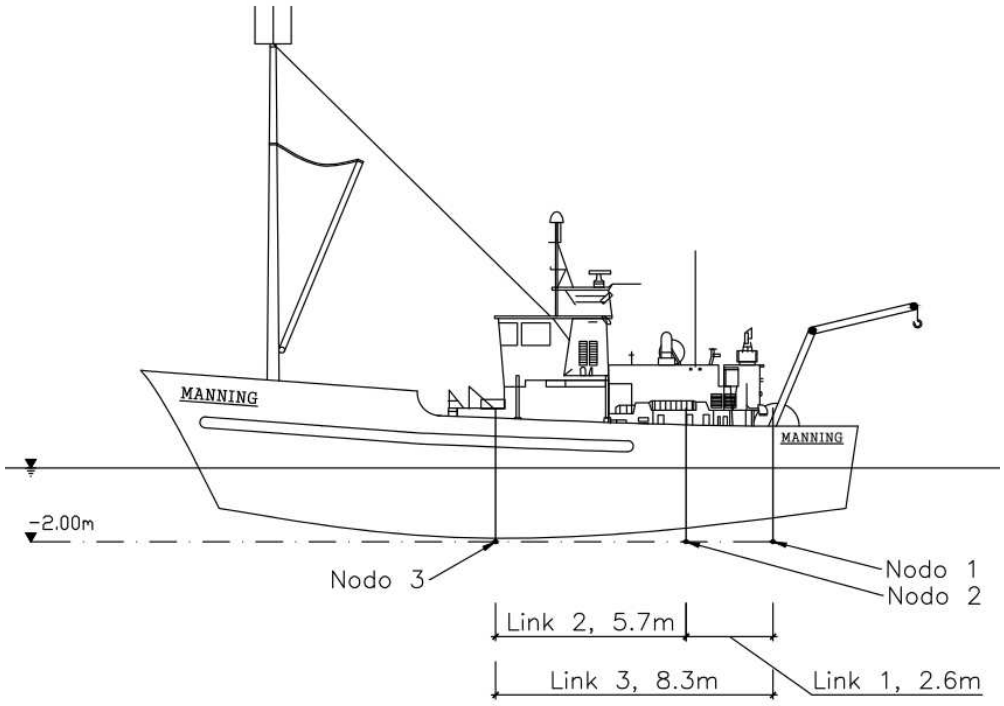
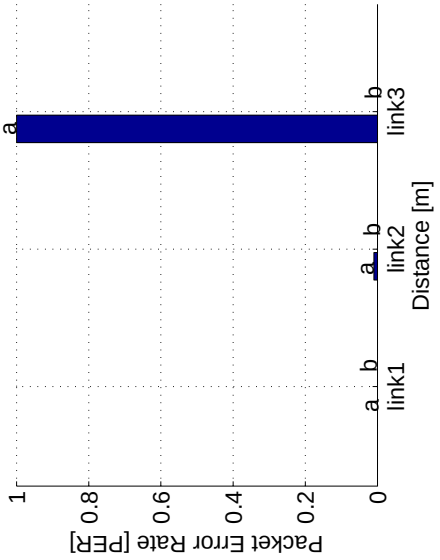
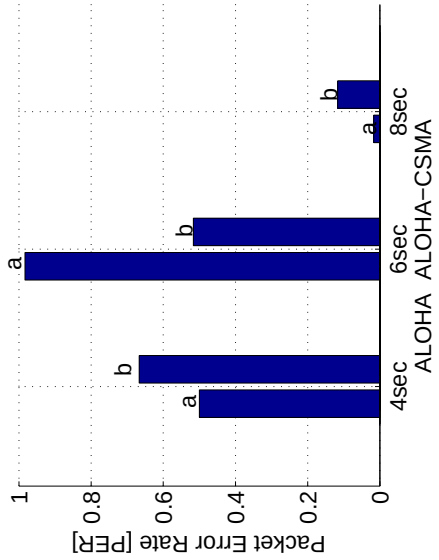


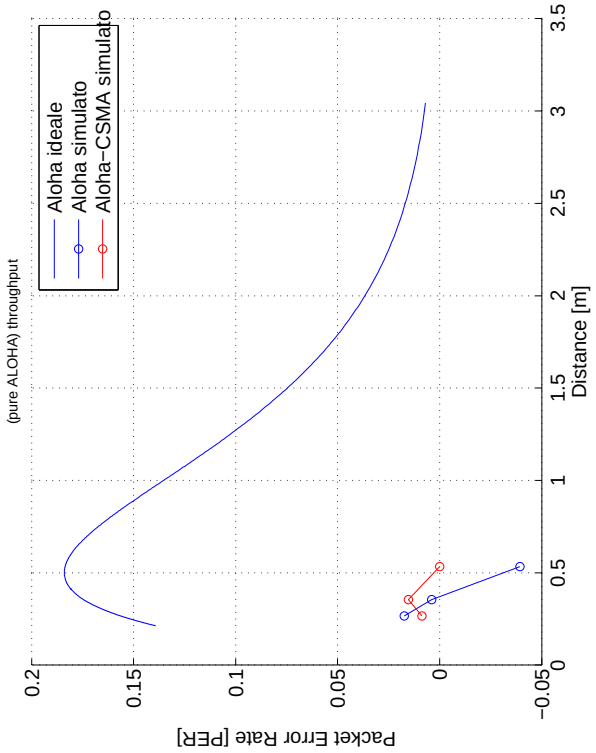
Figura 6.9: Nel disegno è riportata la vista laterale del Manning con le relative disposizioni dei nodi; si tiene a precisare che la profondita del mare nella zona in cui sono stati effettuati i test è stata stimata, dall'eco-scandaglio dell'imbarcazione, essere di circa 40m



(a) Packet Error Rate del canale marino nelle condizioni descritte in questo paragrafo



(b) Packet Error Rate utilizzando ALOHA puro e ALOHA-CSMA su una distanza fissa



(c) throughput utilizzando ALOHA puro e ALOHA-CSMA

Figura 6.10: In questo set di grafici vengono riportati i risultati dei test effettuati in mare a La Spezia. Nella (c) viene riportato il throughput dei pacchetti, partendo da sinistra i tre cerchietti rappresentano trasmissioni di pacchetto ongi 4,6 8 secondi. Si può notare come l'aloha-csma dava meglio per periodi alti.

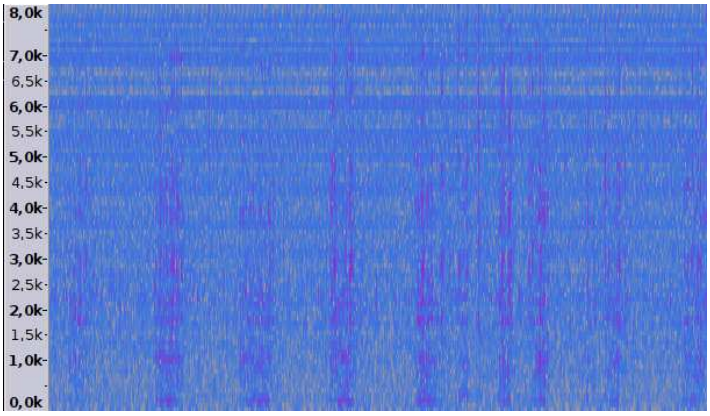


Figura 6.11: In questa immagine è riportato lo spettrogramma delle frequenze di una registrazione; i colori più caldi rappresentano una potenza acustica maggiore identificando il segnale trasmesso.

(a) banco di lavoro

(b) banco di lavoro

(c) trasduttore immerso in acqua

(d) trasduttori

Figura 6.12: Foto dei test effettuati a La Spezia.

Capitolo 7

Attività future

Questo capitolo, posto prima delle conclusione, ha il compito di esporre tutte le idee che sono nate durante questi mesi di lavoro. Di fatto verranno esposte considerazioni generali focalizzate a pianificare nuove attività di ricerca parallele a quanto fatto in questa tesi; come ad esempio l'installazioni del pacchetto NS2/NS-Miracle/DESERT su dispositi embedded a bassissime prestazioni ma che, al tempo stesso, sono presenti in larga scala su sistemi già funzionanti.

7.1 Considerazioni generali

Le attività svolte in questa tesi hanno dato vita a nuove idee sviluppabili in tempi pressoché consecutivi a questo elaborato. Lo studio che interesserà, di sicuro, le settimane successive alla presentazione di questo elaborato sarà l'analisi del malfunzionamento dell'interfacciamento DESERT/PandaBoard, questo aspetto verrà svolto seguendo per passi cercando di confinare in modo intelligente l'errore. Fatto ciò e come già anticipato in precedenza, considerando sistemi embedded tipo NetDCU5.2, in cui le prestazioni sono nettamente inferiori alle media degli attuali dispositivi commercializzati, si potrebbe provare a ridurre al minimo il numero di librerie usate da NS2/NS-Miracle/DESERT in modo da poterle installare in tale dispositivo e provare a sostituire la PandaBoard, per l'appunto, con la NetDCU5.2. Questo "downgrade" darebbe il vantaggio di testare l'idea del progetto Nautilus su sistemi sottomarini già esistenti ma che utilizzano un approccio diverso alla prototipazione esposta in questa tesi. In prima istanza, questo tipo di attività è già stata avviata producendo buoni presupposti per un'effettivo studio in questa direzione.

Un'altra idea, che nelle prime fasi potrebbe sembrare trasversale a questo progetto, è la realizzazione di un AUV[1] con la possibilità di essere controllato

[1] Autonomous Underwater Vehicle (veicolo sottomarino autonomo) è un robot che opera in acqua e che è in grado di portare a termine delle missioni in maniera autonoma

da remoto, come se fosse un ROV[2]. Questo dispositivo, pertanto, avrebbe la possibilità di acquisire delle missioni, come ad esempio scansione dell'ambiente sottomarino secondo determinati target, e al tempo stesso essere in grado di trasmettere o ricevere dati da una postazione fissa (come potrebbe essere una boa[3] o semplicemente un operatore che acquisisce, da remoto, i dati raccolti dall'AUV/ROV) attraverso i micro-modem usati in questa tesi. Una prima idea, potrebbe essere quella di integrare tutto il software sullo stesso dispositivo embedded; questo significherebbe avere, ad esempio, una pandaboard che controlla sia l'ambiente di sviluppo NS2/NS-Miracle/DESERT che il controllo dei motori del veicolo sottomarino. In ogni caso, questa ultima idea dovrà essere studiata e progettata fissando subito dei target di progetto, perchè dovranno essere curati diversi aspetti che erroneamente potrebbero essere messi in secondo piano. Ad esempio, la gestione dell'autonomia delle batterie potrebbe prevedere dei cicli autonomi di carica attraverso dei pannelli solari; in questo caso il supporto di una rete di boe potrebbe pilotare l'AUV/ROV in posizioni geografiche in cui la presenza dei raggi solari è più forte.

[2]Remotely Operated underwater Vehicle (sottomarino a comando remoto) è un robot teleoperato usato in applicazioni sottomarine.

[3]Una boa è un dispositivo che funge da interfaccia tra il canale sottomarino e il canale via etere

Conclusioni

Giunti a questo punto è necessario trarre delle conclusioni. Il lavoro di questi mesi si può ritenere essere stato molto produttivo in quanto gli obiettivi principali:

- scelta del sistema operativo linux da installare sulla PandaBoard cercando un buon compromesso tra configurabilità e livelli prestazionali del software.
- customizzazione del sistema operativo secondo le necessità di progetto.
- installazione del Network Simulator (NS2) con i moduli NS-Miracle
- prove di tipo evaluation-test in modo da fare un primo confronto tra Personal-compure e PandaBoard in termini di simulazione.
- analisi dello spazio occupato dal Network Simulator e tracking dell'utilizzo della RAM in fase di simulazione.
- customizzazione di NS2 in modo da avere un network simulator ottimizzato per la prototipazione di protocolli UnderWater.
- prove di simulazione effettuare direttamente in acque marine.

fissati nelle prime fasi del progetto, sono stati portati a termine con buona soddisfazione.

Per quanto riguarda l'utilizzo delle librerie DESERT, messa in evidenza tra i punti presentati nell'introduzione, e dei dei relativi test (EMULATION setting e TEST-BED setting) effettuati con la PandaBoard si sono avuti dei problemi di interfacciamento; tali ostacoli non hanno compromesso l'intero lavoro in quanto utilizzando un personal computer si è riusciti ad effettuare dei test simulativi sicuramente utilizzabili nelle fasi successive di questo progetto per capire la natura del problema che risiede tra librerie DESERT e PandaBoard.

Proseguendo con l'analisi degli obiettivi citati sopra, si può dire che la scelta del sistema operativo Linux ha portato ad utilizzare una distribuzione Ubuntu[4] di tipo server, in quanto diverse configurazioni di preset si avvicinavano già alle

[4]Ubuntu è un sistema operativo GNU/Linux nato nel 2004,Finanziato dalla società Canonical Ltd (registrata nell'Isola di Man), questo sistema è rilasciato come software libero sotto licenza GNU GPL ed è gratuito e liberamente modificabile

nostre esigenze; questo ha permesso di facilitare la castomizzazione del sistema oparativo. La fortuna di aver avuto un dispositivo embedded molto potente, a livello hardware, ha dato la possibilità di poter lavorare direttamente on-board, evitanto così di usare ambienti di virtualizzazione per architetture ARM[5] su dispositivi di tipo X86[6]; di fatto, riscontrando che i tempi di compilazione si sono revelati accettabili, non è stato necessario effettuare cross-compilazione. Per quanto riguarda l'occupazione su disco si può dire che in fase di compilazione tutto ciò che riguardava l'ambiente grafico non è stato preso in considerazione, dato che dal punto di vista applicativo non è stato necessario avere un'interfaccia grafica. La riduzione dello spazio non è stata molto spinta in quanto, utilizzando una SDHC[7] di classe 10, come archivio di massa, lo spazio a disposizione è stato più che sufficiente. Naturalmente, questo punto, dovrà essere rivisitato in visione di dispositivi con archivio di massa limitati come nel caso della NetDCU5.2.

Spendendo ancora qualche parola sui test effettuati in campo si può mettere in risalto come l'introduzione di "nodi ripetitori" possa contribuire ad avere una rete di comunicazione molto più solida. Di fatto di è potuto provare che con l'introduzione di un terzo nodo tra un semplice link tra due entità ha contribuito a migliorare la Packet Error Rate (PER).

In definitiva, e come anticipato nel capitolo 7, si può concludere che il lavoro svolto in questa tesi ha dato la possibilità di fare una prima analisi sull'utilizzo dei sistemi embedded in simulazioni reali. Anche se tutto non ha funzionato al meglio ci si ritiene molto soddisfatti del lavoro in quanto i feedback ricevuti dalle diverse prove fanno pensare che la strada presa è quella giusta e si allinea perfettamente con le aspettative del progetto NAUTILUS

[5] Advanced RISC Machine,indica una famiglia di microprocessori RISC a 32-bit sviluppata da ARM Holdings e utilizzata in una moltitudine di sistemi embedded. Grazie alle sue caratteristiche di basso consumo l'architettura ARM domina il settore dei dispositivi mobili dove il risparmio energetico delle batterie è fondamentale.

[6] l'architettura x86 è quella utilizzata nei comuni personal computer

[7] Le SDHC (Secure Digital High Capacity, Secure Digital ad alta capacità) sono le schede di memoria più diffuse per dispositivi elettronici; le SDHC sono un'evoluzione delle SD(Secure Digital) e rispetto a quest'ultime si differenziano per una maggiore capacità massima, in termini di MB, e una maggiore velocità di lettura/scrittura

Bibliografia

[1] N. Benvenuto, M. Zorzi, *Principles of Communication Networks and Systems.* John Wiley and Sons Ltd., 1nd ed., December 13 2011. ISBN: 978-0-470-74431-4.

[2] R. Masiero, P. Casari, M. Zorzi, "The nautilus project: Physical parameters, architectures and network scenarios," *MTS/IEEE Oceans*, 2011.

[3] "sito web del progetto NAUTILUS." http://nautilus.dei.unipd.it/.

[4] UC Berkeley, LBL, USC/ISI, and Xerox PARC, *The ns Manual*, Novembre 4 2011. http://www.isi.edu/nsnam/ns/ns-documentation.html.

[5] N. Baldo, M. Miozzo, F. Guerra, M. Rossi, M. Zorzi, "Miracle: The multi-interface cross-layer extension of ns2," *EURASIP Journal on Wireless Communications and Networking*, 2010.

[6] L. Freitag, M. Grund, S. Singh, J. Partan, P. Koski, and K. Ball, "The WHOI Micro-Modem: An Acoustic Communications and Navigation System for Multiple Platforms," in *Proceedings of MTS/IEEE OCEANS 2005*, vol. 2, pp. 1086–1092, Sept. 2005.

[7] Acoustic Communication Group, *Micro-Modem Software Interface Guide.* Copyright 2002-2010, http://acomms.whoi.edu/documents/uModem Software Interface Guide.pdf.

[8] "sito web di riferimento UBUNTU." http://www.ubuntu.com/.

[9] "sito web di riferimento Canonical Foundation." http://www.canonical.com/.

[10] L. Lamport, *LaTeX: A Document Preparation System.* Addison Wesley Professional, 2nd ed., June 1994. ISBN: 0-201-52983-1.

[11] A. Baudoin, "Impara LaTeX!." documentazione online di Latex, 1998.

[12] T. Oetiker, "The not so short introduction to LaTeX 2_ε." documentazione online di Latex, 1999.

[13] Latex project team, "LaTeX 2_ε for authors." online documentation, 1999. directory: /usr/share/texmf/doc/latex/base/, file: userguide.dvi.

[14] "BibConverter – a tool for converting citations." Online available http://www.bibconverter.net/, 2007.

[15] "BibConverter – a tool for converting citations: demo page." Online available http://www.bibconverter.net/demo/bmdemo.htm, 2007.

Elenco delle tabelle

Elenco delle figure

Indice analitico

Ringraziamenti

Il ringraziamenti piú grande va sicuramente ai miei genitori, se oggi sono arrivato fin qua é sicuramente grazie al loro supporto in questi anni. L'ambizione a non mollare mai e sicuramente grazie ai loro insegnamenti.
Dal punto di vista accademico non posso che dire grazie mille al mio relatore Lorenzo Vangelista; per avermi fornito la grande possibilitá di approfondire argomenti di mio grande interesse e per avermi dato importanti consigli in questo periodo. Infine, un grazie altrettanto importante va Riccardo Masiero e a Paolo Casari per la loro disponibilitá in questi mesi di lavoro, il loro aiuto é stato fondamentale in questo progetto.