



Ricerca di Sistema elettrico

Sviluppo di metodologie e strumenti di analisi delle prestazioni di reti di smart meter

C. Landi, D. Gallo, M. Luiso



D^{III} SUN
Dipartimento di Ingegneria
Industriale e dell'Informazione

SVILUPPO DI METODOLOGIE E STRUMENTI DI ANALISI DELLE PRESTAZIONI DI RETI DI SMART METER

C. Landi, D. Gallo, M. Luiso (Seconda Università degli Studi Di Napoli, Dipartimento di ingegneria Industriale e dell'Informazione)

Settembre 2014

Report Ricerca di Sistema Elettrico

Accordo di Programma Ministero dello Sviluppo Economico – ENEA

Piano Annuale di Realizzazione 2013

Area: Razionalizzazione e risparmio nell'uso dell'energia elettrica

Progetto: Risparmio di energia elettrica nei settori: civile, industria e servizi

Obiettivo: Reti di poligenerazione distribuita

Responsabile del Progetto: Ilaria Bertini, ENEA

Il presente documento descrive le attività di ricerca svolte all'interno dell'Accordo di collaborazione "Metriche Di Misura Avanzate Ed Architetture Di Concentratori Dati Intelligenti Per Reti Integrate Di Servizi"

Responsabile scientifico ENEA: Giorgio Graditi

Responsabile scientifico SUN: Carmine Landi

Indice

1	Sommario	5
2	Panoramica sulle Smart Grid e reti di Smart Meter	5
2.1	Smart Grid	5
2.2	Smart Meter	6
2.3	Concentratore dati	7
3	Hardware utilizzato per la realizzazione della rete di Smart Meter	8
3.1	Single-board computer - Raspberry Pi	8
3.2	Microprocessori ARM.....	11
3.3	Microcontrollori STM32	15
3.4	Modem Powerline	17
4	Tecnologie informatiche per lo sviluppo del Software	19
4.1	Python	19
4.2	MySQL	23
4.3	PHP	25
4.4	HTML	27
5	Realizzazione del concentratore dati embedded	30
5.1	Fase preliminare di progettazione e scelta dei parametri di comunicazione	30
5.2	Scelta dell'architettura della rete di Smart Meter	32
5.3	Codice dell'applicazione concentratore di dati	35
5.4	Script per la visualizzazione in locale o in remoto.....	39
5.5	Creazione de database MySql	44
6	Implementazione e test sperimentali	45
6.1	Implementazione e configurazione della rete di Smart Meter	45
6.2	Test della parte software	46
6.3	Test della tecnologia di comunicazione	48
6.4	L'ambiente Labview.....	54
6.5	Calcolo Del Bit Error Rate	56
6.6	L'analizzatore Di Spettro Agilent E74002a	61

6.7	Caratterizzazione della Linea Elettrica	66
7	Riferimenti bibliografici	71
8	Curriculum scientifico del gruppo di lavoro impegnato nell'attività	73

1 Sommario

Il Ministero dello Sviluppo Economico ed ENEA hanno stipulato in data 4 marzo 2013 un Accordo di Programma in base al quale è concesso il contributo finanziario per l'esecuzione delle linee di attività del Piano Triennale della Ricerca e Sviluppo di Interesse Generale per il Sistema Elettrico Nazionale.

In continuità con quanto sviluppato nella precedente annualità e con quanto pianificato nel piano triennale 2012-2014, i temi sviluppati nell'ambito del presente accordo di collaborazione tra ENEA e Seconda Università degli studi di Napoli (SUN) riguardano lo sviluppo di metodologie di misura avanzate per reti di distribuzione asservite all'alimentazione di reti integrate di servizi per distretti energetici e la definizione delle specifiche dell'architettura di "concentratori dati intelligenti" che consentono l'interazione tra generazione e utilizzazione e prevedono accesso da remoto e pubblicazione dati. In tale contesto si è proceduto allo studio delle possibili soluzioni progettuali e gestionali che consentano l'implementazione di una rete di smart multi-metering includendo, anche, la sua caratterizzazione metrologica e la validazione funzionale.

Obiettivo del contratto è stato, quindi, la realizzazione e caratterizzazione di un prototipo di concentratore dati che garantisca la ricezione dei dati utilizzando moderni approcci per la trasmissione in sicurezza e la verifica di accuratezza ed affidabilità del dato attraverso l'analisi combinata delle letture in differenti punti della rete. Il prototipo realizzato ha consentito di individuare e testare metodologie e strumenti di analisi delle prestazioni di reti di smart meter che utilizzano comunicazioni PLC per la trasmissione dei dati al concentratore.

I principali risultati dell'attività in oggetto includono:

1. La realizzazione di un prototipo di concentratore dati per reti di smart meter;
2. La realizzazione di un sistema di trasmissione dati su linea elettrica di potenza utilizzando moderni approcci per la trasmissione in sicurezza;
3. La definizione ed applicazione di metodologie di analisi delle prestazioni del sistema realizzato.

2 Panoramica sulle Smart Grid e reti di Smart Meter

In questo paragrafo sarà fatta una panoramica per quanto riguarda le *Smart Grids* e alcuni suoi componenti, ovvero lo *Smart Meter* e il concentratore di dati. Per quest'ultimo in particolare, oggetto specifico della presente convenzione, si è fatto ricorso ad un PC come il *Raspberry Pi*.

2.1 Smart Grid

Il termine *Smart Grid* indica una rete intelligente, che coinvolge attivamente sia i consumatori che i produttori, al fine di realizzare un sistema di distribuzione dell'energia che presenti le seguenti caratteristiche: efficienza; sostenibilità; economicità; sicurezza.

Infatti una *Smart Grid*, è l'insieme di una rete di informazione e di una rete di distribuzione elettrica, organizzata in modo tale da consentire di gestire la rete elettrica in maniera "intelligente" sotto vari aspetti o funzionalità, ovvero in maniera efficiente per la distribuzione di energia elettrica e per un uso più razionale dell'energia, minimizzando al contempo, eventuali sovraccarichi e variazioni della tensione elettrica intorno al suo valore nominale. Le *Smart Grid* rappresentano un'importante proposta per l'evoluzione della rete elettrica. I siti di controllo di una *Smart Grid*, i sottosistemi, i nodi e le apparecchiature dell'utente finale possono essere suddivisi in quattro categorie, ciascuna delle quali evidenzia vulnerabilità specifiche per quel che riguarda la sicurezza:

- sistemi d'esercizio: generatori, trasformatori, sistemi Scada (*Supervisory Control & Data Acquisition*), sistemi Ems/Dms (*Energy Management Systems/Distribution Management Systems*), sotto-stazioni, gestori dei sistemi di trasmissione e gestori dei sistemi di distribuzione.
- sistemi IT tradizionali: PC, server, mainframe, database, siti web e servizi web.

- protocolli e reti di comunicazione: *Ethernet, Wi-Fi, Prime, Dlms/Cosem, ZigBee*, reti di telefoni mobili.
- punti terminali: Smart Meter, veicoli elettrici, *smartphone* e altri dispositivi mobili.

Le applicazioni e i protocolli di comunicazione utilizzati da questi sistemi fanno uso di linguaggi strutturati e tecnologie di tipo *web-based*, caratteristiche queste che le reti appena sopra descritte sono in grado di supportare. In questo modo la *Smart Grid* può fornire l'accesso in tempo reale alle tariffe elettriche e alle informazioni relative ai consumi al fine di supportare la funzionalità di DR (*Demand-Response*, termine che indica la capacità della rete elettrica di adattarsi dinamicamente alle variazioni di richiesta di energia da parte dei consumatori). In altre parole, un'applicazione dello *smartphone* dell'utente sarà in grado di tracciare automaticamente le tariffe elettriche nell'arco della giornata, in quanto esse aumentano e diminuiscono in base alla domanda e innescare da remoto un segnale di avvio per un elettrodomestico, per mezzo di un comando inviato attraverso Internet e istruito verso lo *Smart Meter* dell'utilizzatore nel momento in cui il prezzo dell'elettricità scende al di sotto di una soglia preimpostata. Tutto ciò va a vantaggio sia del consumatore, che paga un prezzo minore per l'energia, sia della società erogatrice del servizio che, grazie alle fluttuazioni più ridotte della domanda, può generare elettricità in maniera più efficiente ed economica. Sulla base di quanto finora esposto appare chiaro che uno "Smart Meter" è esposto ad altri rischi. Per contrastare i rischi, le aziende di semiconduttori hanno sviluppato tecnologie che assicurano un elevato grado di sicurezza e protezione per i dispositivi che formano una "*Smart Grid*"[3].

2.2 Smart Meter

Uno *Smart Meter* è di solito un misuratore intelligente che registra il consumo di energia elettrica a intervalli di un'ora o meno e comunica tali informazioni, almeno giornalmente, al programma di utilità per il monitoraggio e ai fini della fatturazione. Essi consentono una comunicazione a due vie tra il misuratore e il sistema centrale. A differenza dei contatori di energia a casa, gli *Smart Meter* sono in grado di raccogliere i dati e di trasmetterli a distanza.

Il termine *Smart Meter* spesso si riferisce ad un contatore elettrico, ma può anche riferirsi ad un dispositivo di misurazione di gas naturale o di consumo d'acqua. Questa rappresenta una tecnologia di grande utilità nel campo dell'efficienza energetica perché è lo strumento impiegato per la misurazione dei risparmi conseguibili a seguito di interventi di efficienza. La sua applicazione, infatti, consente di accompagnare ogni intervento di efficienza dalla fase progettuale, con la misurazione e valutazione dei consumi e delle dispersioni di energia di un impianto o di un edificio, prima dell'intervento di riqualificazione, passando per il monitoraggio nel corso della fase di realizzazione e terminare con la misurazione e il controllo in telegestione dei consumi post intervento e dei risparmi conseguiti. Questo sistema di controllo è basato su reti di sensori (wireless, PLC, RS485) per il monitoraggio in tempo reale dei consumi di luce, gas e acqua. Grazie alla possibilità di interfaccia con le tecnologie informatiche e di comunicazione, esso consente di intervenire sugli impianti regolando lo scambio sia di energia che di informazioni sul loro funzionamento, offrendo anche la possibilità di intervenire in caso di problematiche o guasti in modalità immediata, senza dover ricorrere all'intervento sul posto. Le tecnologie di cui si compone, in particolare la sensoristica, sono tecnologie già mature e ampiamente diffuse sul mercato e accessibili a prezzi contenuti. Pertanto il ricorso allo *Smart Metering* è auspicabile ad ogni livello della rete di distribuzione e consumo di energia, dalla centrale, alla rete intelligente fino alla singola unità abitativa, in quanto con un costo contenuto permette da sola di valutare i consumi energetici e alla luce dei risultati riscontrati programmare interventi di efficienza. Lo *Smart Metering* è inoltre strumento indispensabile nell'evoluzione delle reti elettriche tradizionali in *Smart Grid*. [Enea, 2013] In Italia Enel, da diversi anni, ha iniziato la sostituzione dei vecchi contatori elettromeccanici con gli *Smart Meter*. Ad oggi oltre 32 milioni ne sono già stati installati nel paese. L'azienda italiana rende noto, inoltre, che la gestione da remoto di tali contatori consente di evitare l'emissione di 30 mila tonnellate di CO₂ ogni anno. In figura 1 viene illustrato uno *Smart Meter Enel*.



Figura 1 Smart Meter Enel

Lo *Smart Concentrator*, invece, viene installato nelle cabine di trasformazione da media a bassa tensione ed effettua la gestione dei contatori ad esso connessi via PLC o RF. Tale dispositivo periodicamente rileva i dati dagli Smart Meter, li salva e li invia al sistema centrale, dal quale a sua volta riceve i comandi da comunicare ai contatori. A coordinare il tutto vi è un sistema centrale che riceve i dati dai nodi concentratori e li elabora, in modo da gestire il funzionamento dei contatori, le operazioni tariffarie e la qualità del servizio. In figura 2, viene riportato lo schema del servizio di *telegestione* di Enel, realizzato mediante l'interconnessione dei dispositivi precedentemente descritti e che consente al DSO italiano di gestire e monitorare la rete in tempo reale [4].

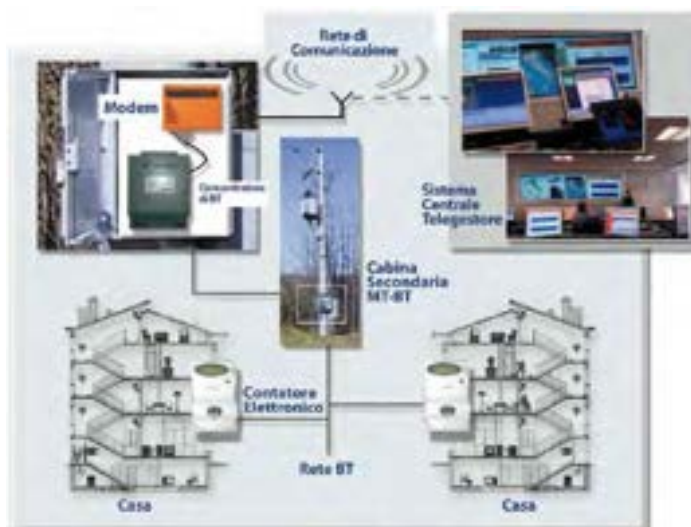


Figura 2 Sistema di telegestione Enel

2.3 Concentratore dati

Il concentratore dati è un componente essenziale di un sistema per la lettura dei contatori. Esso supporta l'infrastruttura collegando più contatori (elettricità, gas, acqua, riscaldamento) a un server centrale dell'azienda di servizio pubblico ed effettuando operazioni di acquisizione e reporting di dati. Il concentratore dati si occupa anche di sincronizzare date e tempi dei contatori con il server centrale e consentire il trasferimento sicuro delle informazioni relative alla cifratura e all'autenticazione dell'utente. Esso permette di avere a disposizione tutti i dati inviati periodicamente dai vari *Smart Meter* della *Smart Grid* attraverso un software di gestione; questo è molto importante perché rende più facile

l'individuazione di eventuali guasti in una rete e con un'azione automatica del software di controllo può prevenire la rottura di dispositivi appartenenti a quel nodo. La visualizzazione dei dati inviati inoltre può avvenire anche tramite Internet, utilizzando le opportune credenziali per accedere al database della pagina Web del concentratore dati. La comunicazione con i contatori avviene attraverso una connessione RF o cablata (*modem powerline* (PLC)) sfruttando protocolli quali *Ethernet*, *Gsm*, *Gprs* o *Pots*. Il concentratore dati di una *Smart Grid* è stato realizzato utilizzando un *Raspberry Pi* come concentratore dati collegato ad uno *Smart Meter* tramite il canale seriale RS232 per monitorare i dati acquisiti e renderli disponibili all'utente. La progettazione e realizzazione dettagliata sarà poi affrontata nei paragrafi seguenti.

3 Hardware utilizzato per la realizzazione della rete di Smart Meter

Nel seguente paragrafo vengono introdotti i componenti hardware utilizzati per lo sviluppo del progetto, in particolare si farà attenzione alla descrizione dettagliata del single-board computer Raspberry Pi e il microcontrollore basato su core Cortex M4.

3.1 Single-board computer - Raspberry Pi

Il Raspberry Pi è un single-board computer (un calcolatore implementato su una sola scheda elettronica) sviluppato nel Regno Unito dalla Raspberry Pi Foundation. Il suo lancio al pubblico è avvenuto alla fine del mese di febbraio 2012. L'idea di base è la realizzazione di un dispositivo economico, concepito per stimolare l'insegnamento di base dell'informatica e della programmazione nelle scuole. Il progetto ruota attorno a un *System-on-a-chip* (SoC) *Broadcom BCM2835*, che incorpora un processore ARM1176JZF-S a 700 MHz, una GPU VideoCore IV, e 256 o 512 MB di memoria. Il progetto non prevede nè *hard disk* nè un'unità a stato solido, affidandosi invece a una scheda SD per il *boot* e per la memoria non volatile. La scheda è stata progettata per ospitare sistemi operativi basati su un *kernel* Linux o RISC OS.

3.1.1 Fondazione

Lo sviluppo del dispositivo è portato avanti dalla Raspberry Pi Foundation, organizzazione di beneficenza registrata presso la *Charity Commission for England and Wales*. La Raspberry Pi Foundation principalmente promuove l'apprendimento del linguaggio di programmazione Python, ma permette anche l'uso del BBC BASIC, del C, del Perl e molti altri linguaggi supportati da Linux e ARM.

3.1.2 Storia del progetto

Le prime concezioni del Raspberry Pi, nel 2006, si basavano sul microcontrollore Atmel ATmega644. L'amministratore Eben Upton mise insieme un gruppo di insegnanti, studenti e appassionati di computer, per concepire un oggetto capace di incoraggiare i bambini, fornendo loro *know-how* e ispirazione. La Raspberry Pi Foundation fu fondata nel maggio 2009, a Caldecote, villaggio del South Cambridgeshire, nel Regno Unito, con lo statuto giuridico di organizzazione caritatevole registrata, regolata dalla Charity Commission for England and Wales. La prima versione del prototipo basata su ARM era montata su una scheda grande grosso modo come una chiave USB, con una porta USB su un lato e una porta HDMI sull'altro. Nel mese di agosto 2011 furono realizzate cinquanta versioni alpha della scheda. Queste schede erano funzionalmente identiche alla struttura progettata per il modello B. Le versioni di prova della scheda mostravano l'ambiente desktop LXDE su Debian, Quake 3 a 1080p, e video Full HD H.264 su HDMI. Nel mese di ottobre 2011 vi è stata la scelta del logo all'interno di un numero di proposte dei membri della comunità, dopo aver stilato una rosa contenente sei proposte, impiegando vari giorni per la scelta finale. Il disegno scelto si basava su una buckyball, ovvero un'immagine 3D composta da 32 facce. Nel mese di dicembre 2011 sono state assemblate e testate un centinaio di versioni Beta del modello B e il layout della componentistica usato in questa versione è lo

stesso di quella destinata alla produzione, salvo un errore di sbrogliatura nel progetto del circuito stampato scoperto e risolto prima di avviare la produzione. Le schede Beta ebbero una dimostrazione pubblica con avvio da Linux; in quell'occasione venne proposta la riproduzione di un trailer a 1080p e l'esecuzione di una demo del benchmark OpenGL ES Samurai, prodotto dalla finlandese Rightware (ex Futuremark). Le prime 10 schede furono messe all'asta su eBay nelle prime settimane del 2012 e una di queste è stata comprata da un anonimo e donata al museo inglese The Centre for Computing History, ubicato nel Suffolk.

3.1.3 Raspberry Pi revisione 2

A settembre 2012 è stata annunciata una revisione del modello B del Raspberry Pi. Novità principali della scheda sono la capacità di ricevere l'alimentazione tramite un hub USB alimentato e la disponibilità del debug via JTAG. Come novità minori si ha la correzione di un difetto di collegamento tramite HDMI (lasciando il Raspberry Pi non alimentato su una catena HDMI, potevano insorgere problemi nell'utilizzo delle funzionalità Consumer Electronics Control per gli altri dispositivi). Il circuito di reset è stato rinnovato, è stato aggiunto un connettore di espansione addizionale, sono stati aggiunti i fori di montaggio e apportate correzioni alla serigrafia. Sono inoltre presenti cambiamenti ai GPIO e ai canali I2C. Si noti che la dicitura "Raspberry 2.0" comparsa in alcuni siti è concettualmente sbagliata, poiché le modifiche apportate comportano unicamente la correzione di alcuni dettagli progettuali e nulla che giustifichi l'incremento nel numero di versione da 1.0 a 2.0, come riportato anche nel sito del produttore. La scheda ha semplicemente subito una revisione al circuito; infatti, è il circuito stampato che è alla seconda versione, non la scheda nel suo complesso che mantiene gli stessi identici componenti.

3.1.4 Caratteristiche

La fondazione distribuisce due modelli, entrambi dotati di 256 MB di RAM: il Model A (figura 3) che ha una singola porta USB ed è privo di un controller Ethernet, e il Model B (figura 4) che è equipaggiato con due porte USB ed un controller Ethernet 10/100. A partire del 15 Ottobre 2012 vi è stato un upgrade e il Model B monta 512 MB di RAM. Sebbene il Model A non abbia una porta Ethernet RJ45, può comunque accedere a una rete attraverso la porta USB, facendo uso di adattatori Ethernet o Wi-Fi con alimentazione autonoma. In maniera analoga ai moderni computer, Raspberry Pi è compatibile con tastiere e mouse generici collegabili tramite porta USB.

Raspberry Pi usa vari sistemi operativi:

- Raspbian ovvero un port di Debian Wheezy
- RaspBMC e OpenELEC basati su XBMC
- Pidora basato su Fedora
- Arch ovvero un port di Archlinux
- RISC OS

Raspberry PI non è fornito di un real time clock, così un sistema operativo deve usare un network time server o chiedere l'ora all'utente al bootstrap per avere accesso a data e ora. Tuttavia è facile aggiungere un real time clock (come il DS1307) con batteria tampone, attraverso l'interfaccia I2C.

Le specifiche dei due modelli di Raspberry Pi sono riportate in tabella 1.



Figura 3

Raspberry Pi Model A

Raspberry Pi Model

Specifiche

	Model A	Model B
Prezzo di offerta:	USD 25 (GBP 16)	USD 35 (GBP 22)
SoC:	Broadcom BCM2835 (CPU + GPU + DSP + SDRAM)	
CPU:	700 MHz ARM1176JZF-S core (famiglia ARM11)	
GPU:	Broadcom VideoCore IV, OpenGL ES 2.0, 1080p30 H.264 high-profile	
Memory (SDRAM):	256 Megabytes(condivisa con	256 o 512 Megabytes(condivisa con
USB 2.0 ports:	1	2 (attraverso un hub USB integrato)
Output video:	Connettore RCA per il video composito, HDMI	
Output audio:	3,5 mm jack, HDMI	
Memoria:	SD / MMC / SDIO card slot	
Collegamenti di rete:	Nessuno	Ethernet 10/100 (RJ-45)
Periferiche di basso livello:	2x13 header pins for GPIO, SPI, I ² C, UART, +3,3 Volt, +5 Volt	
Real-time clock:	No clock or battery	
Corrente (potenza) assorbita:	300 mA, (1,5 W)	700 mA, (3,5 W)
Alimentazione:	5 V via MicroUSB o GPIO header	
Dimensioni:	85,60 mm × 53,98 mm (3.370 inch × 2.125 inch)	
Sistemi operativi supportati:	Debian GNU/Linux, Fedora, Arch Linux e Gentoo	
Sistemi operativi non supportati:	RISC OS (shared source)	

Tabella 1 Specifiche dei due modelli di Raspberry Pi

3.1.5 Software

La Raspberry Pi Foundation ha messo a disposizione per il download un proof of concept di immagine che può essere caricata su SD card per produrre un sistema operativo preliminare. L'immagine si basa su Debian 6.0 (Squeeze), con un ambiente desktop LXDE e un browser Midori, più vari strumenti di programmazione. L'immagine può anche girare sull'emulatore QEMU, permettendo di emulare Raspberry Pi su varie altre piattaforme. La Fondazione ha realizzato una release ottimizzata di Fedora,

raccomandandola come sistema operativo. È disponibile anche una versione di Arch Linux. Esistono distribuzioni per l'utilizzo del Raspberry Pi come Media Center basate su XBMC: OpenELEC, XBian e RaspBMC. Il software di monitoraggio di rete Overlook Fing è stato portato su piattaforma Raspberry Pi rendendo possibile l'installazione di sentinelle di monitoraggio a basso costo in reti remote. Il software open source Aseba per la programmazione semplice ed efficiente di robot è disponibile su Raspberry Pi. Utilizzando il Raspberry PI in unione con Aseba e il robot Thymio II è possibile creare a costi veramente contenuti un vero e proprio laboratorio didattico di Robotica.

Il robot Thymio II è stato sviluppato nell'ambito del programma NCCR Robotics dalla collaborazione tra l'École Polytechnique Fédérale de Lausanne (EPFL) e l'École Cantonale d'Art de Lausanne (ECAL).

3.1.6 Formati supportati

Dato che il progetto ha come obiettivo la riduzione dei costi, la decodifica in hardware di alcuni formati multimediali non è supportata perchè richiede una specifica licenza. Il dispositivo può riprodurre in hardware formati liberi, quali H.264, mentre per riprodurre i formati MPEG-2 e VC-1 è possibile acquistare la relativa licenza abilitando l'hardware alla decodifica.[Wikipedia, 2013].

3.2 Microprocessori ARM

L'architettura ARM (precedentemente Advanced RISC Machine, prima ancora Acorn 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 (rapportato alle prestazioni) l'architettura ARM domina il settore dei dispositivi mobili dove il risparmio energetico delle batterie è fondamentale. Attualmente la famiglia ARM copre il 75% del mercato mondiale dei processori a 32 bit 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.).

3.2.1 Microcontrollori basati su ARM Cortex-M

Il Microcontrollori della famiglia ARM Cortex-M Series sono processori embedded ottimizzati per applicazioni a basso costo. Questi processori supportano il set di istruzioni Thumb-2. L'ARM Cortex-M4 è l'ultimo processore embedded della famiglia ARM Cortex-M, nato come evoluzione del core ARM Cortex™-M3.

Il core M4 offre prestazioni avanzate di controllo digitale e quindi ideale per applicazioni DSC, ovvero applicazioni di controllo ed elaborazione del segnale. Le sue caratteristiche sono basso assorbimento ed alta efficienza, è particolarmente adatto alle applicazioni in real time ed è stato progettato per rispondere alle richieste di soluzioni flessibili specificamente mirate a campi come il motor control, auto motive, gestione energetica, sistemi embedded audio e dell'automazione industriale.

Il Cortex-M4F è un processore con le stesse capacità del processore Cortex-M4, e include le funzionalità di aritmetica in virgola mobile (FPU). Gli ARM Cortex-M3 sono molto simili nelle caratteristiche agli M4, ma sono indirizzati a quelle applicazioni embedded che richiedono una risposta rapida agli interrupt, come sistemi di controllo auto motive e industriali.

L'ARM Cortex-M1 FPGA è stato progettato per applicazioni embedded che richiedono un piccolo processore integrato in un FPGA.

L'ARM Cortex-M0 e M0+ sono processori ad alta efficienza energetica destinati a microcontrollori e applicazioni embedded che richiedono un consumo di potenza e dimensioni del processore ottimizzate.

Il Cortex-M4 è un processore a 32 bit ad alto rendimento. Offre notevoli vantaggi per gli sviluppatori, tra cui:

- Eccezionali prestazioni di elaborazione combinate con una veloce gestione degli interrupt.

- Sistema di debug migliorato.
- Processor core e memoria ad elevate performance
- Bassissimo consumo di potenza nelle diverse modalità di standby e sleep mode.

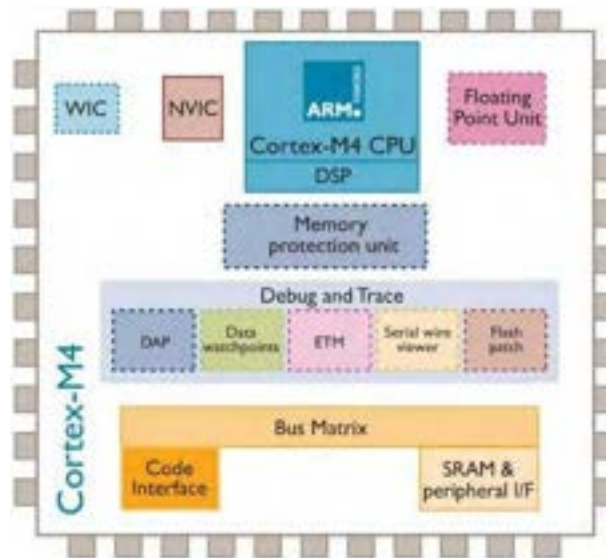


Figura 4 Cortex-M4 implementation

Il processore Cortex-M4 si basa su di un core ad alte prestazioni, con una architettura Harvard pipeline a tre fasi, che lo rende ideale per le applicazioni di tipo embedded. Il processore assicura un'eccezionale efficienza energetica attraverso un set di istruzioni efficiente ed ottimizzato per facilitare la progettazione di dispositivi a basso costo. Il processore Cortex-M4 implementa i componenti del sistema strettamente accoppiati in modo tale da ridurre significativamente le dimensioni del processore migliorando sensibilmente la gestione delle interrupt e le funzionalità di debug di sistema. Il processore Cortex-M4 implementa una versione del set di istruzioni Thumb[®] basato sulla tecnologia Thumb-2, garantendo un'elevata densità di codice. Il set di Istruzioni del Cortex-M4 fornisce le prestazioni eccezionali che ci si aspetta da una moderna architettura a 32 bit, con elevata densità di codice. Il processore Cortex-M4 si integra strettamente con l'unità standard Nested Vectored Interrupt Controller (NVIC), per offrire elevate prestazioni nella gestione delle interrupt. L'NVIC include un interrupt non mascherabile (NMI) che può fornire fino a 256 di priorità di interrupt. La stretta integrazione del core e dell'unità NVIC fornisce una rapida esecuzione delle routine di servizio di interrupt (ISR), riducendo drasticamente la latenza degli interrupt. Questo risultato è ottenuto attraverso la sovrapposizione dei registri hardware, e la possibilità di sospendere le operazioni di multiple di caricamento e memorizzazione.

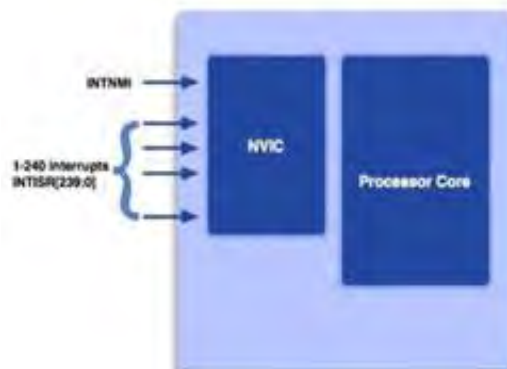


Figura 5 NVIC CORE Cortex M4

Figura 6

Il processore Cortex-M4 fornisce interfacce multiple che gestisce utilizzando AMBA (Advanced Microcontroller Bus Architecture) garantendo un'elevata velocità e bassa latenza negli accessi alla memoria. Supporta dati non allineati, accede a e implementa la manipolazione del singolo bit che consente di controllare più velocemente le periferiche.

3.2.2 Debug Integrato

Il processore Cortex-M4 implementa un debugger integrato, che fornisce visibilità del sistema del processore e della memoria sia attraverso una tradizionale porta JTAG o una porta SWD (Serial Wire Debug) che è l'ideale per i microcontrollori e altri dispositivi di piccole dimensioni.

3.2.3 Periferiche interne

Il Cortex M4 include molte periferiche come:

- Nested Vectored Interrupt Controller: è un controller di interrupt integrato che supporta l'elaborazione a bassa latenza degli interrupt.
- System Control Block (SCB): fornisce informazioni sul sistema di attuazione e controllo del sistema. Questo include la configurazione, il controllo e il reporting delle eccezioni di sistema.
- SysTick: è il timer di sistema, è un count down timer a 24-bit. Può essere utilizzato come RTOS (Real Time Operating System) o come un semplice contatore.
- Unità di protezione della memoria (MPU): migliora l'affidabilità del sistema, definendo le caratteristiche di memoria per le regioni di memoria differenti.
- Floating-point Unit: fornisce la possibilità di effettuare operazioni conformi allo standard IEEE 754 per operazioni a singola precisione, a 32 bit, e valori in virgola mobile.

3.2.4 Pipeline

La CPU del Cortex è in grado di eseguire più istruzioni in un unico ciclo. Come la CPU dell'ARM7 e dell'ARM9 questo è realizzato con una pipeline a tre stadi, mentre esegue un'istruzione, la prossima viene decodificata e la terza viene prelevata dalla memoria.

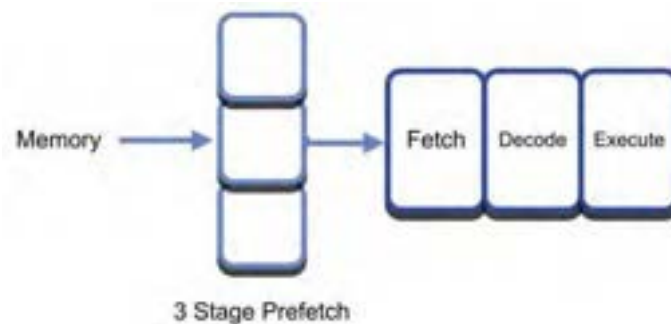


Figura 7 schema logico pipe line

Comunque il Cortex-M4 supporta anche la branch prediction per minimizzare il numero di flush della pipeline.

3.2.5 Eccezioni e interrupt

Il processore Cortex-M4 supporta gli interrupt ed eccezioni del sistema. Il processore e la NVIC gestiscono tutte le interruzioni del sistema. Un'interruzione cambia il normale flusso di controllo software. Il processore utilizza la modalità Handler per gestire tutte le interruzioni, tranne per il reset.

Interruzioni

Ogni interrupt può trovarsi in uno dei seguenti stati:

- Inattivo: l'eccezione non è né attiva né in attesa di essere servita.
- In Attesa: l'eccezione è in attesa di essere servita dal processore. Una richiesta di interrupt da una periferica o dal software può cambiare lo stato di attesa corrispondente all'interrupt.
- Attivo: l'eccezione è gestita da parte del processore, ma non è stata ancora completata.
Nota: un gestore di eccezione può interrompere l'esecuzione di un altro gestore di eccezioni. In questo caso entrambe le eccezioni sono nello stato attivo.
- Attivi e in sospeso: l'eccezione è gestita dal processore e vi è un'altra eccezione in sospeso.

Tipi di eccezioni

- Il microcontrollori basati su ARM CORTEX M4 possono gestire le seguenti tipologie di interruzione
- Reset: Il reset è un'interruzione particolare che ha il livello massimo di priorità, se un'interruzione di reset viene richiamata allora il processore si interrompe indipendentemente dall'operazione che sta svolgendo e il sistema viene riavviato.
- NMI (Non maskable interrupt): una interrupt non mascherabile (NMI) è un tipo particolare di interrupt che non può essere ignorato dalla CPU questa eccezione ha una priorità diversa dal reset. È sempre attiva e ha una priorità fissa di -2. NMI, può essere utilizzato da una periferica per segnalare un malfunzionamento.
- HardFault: è un'eccezione che si verifica a causa di un errore durante l'elaborazione delle eccezioni. Le eccezioni di tipo HardFaults hanno una priorità fissa di -1, nel senso che hanno una priorità più alta rispetto a qualsiasi eccezione con priorità configurabile.
- MemManage: è un'eccezione che si verifica a causa di un errore di protezione della memoria.
- BusFault: è un'eccezione che si verifica quando un programma tenta di accedere ad una posizione di memoria alla quale non gli è permesso accedere, oppure quando tenta di accedervi in una maniera che non gli è concessa. Questo potrebbe essere rilevato come un errore sul bus nel sistema di memoria.

- UsageFault: è un'eccezione che si verifica a causa di un errore che riguarda l'esecuzione delle istruzioni. Come ad esempio un'istruzione non definita, un accesso non allineato in memoria o una divisione per zero.
- SysTick: è un'eccezione generata da un Timer (quando quest'ultimo raggiunge lo zero).
- Interrupt (IRQ): un allarme, o IRQ, è un'eccezione segnalata da una periferica, o generata da una richiesta del software. Tutti gli allarmi sono asincroni rispetto all'esecuzione delle istruzioni. Questo tipo di interrupt è spesso utilizzato dalle periferiche per comunicare con il processore.

3.2.6 Gestori delle eccezioni

Il processore gestisce le eccezioni utilizzando:

ISR (Interrupt service routine), tutti gli interrupt di tipo IRQ sono eccezioni gestite dall'ISR. Fault handlers (Gestori d'errore) le interruzioni di tipo HardFault, MemManage fault, UsageFault, e BusFault sono eccezioni trattate dai gestori di errore.

System handlers (gestori del sistema) NMI, PendSV, SVC, SysTick, sono tutte eccezioni di sistema che vengono gestite da gestori del sistema.

3.2.7 Risparmio energetico

I processori ARM CORTEX sono i più utilizzati nel settore dello sviluppo di dispositivi mobili proprio per la loro capacità di fornire elevate prestazioni e consumi energetici ridotti. I processori cortex M4 implementano due modalità per ridurre il consumo energetico del processore:

- Sleep Mode (modalità di sospensione) si ferma il clock del processore
- Deep Sleep Mode (modalità di sonno profondo) si ferma l'orologio del sistema, si spegne il PLL e la memoria flash.

Possono essere implementate entrambe le modalità per fornire vari livelli di risparmio energetico.

Il core Cortex può essere messo in modalità 'sleep' dall'esecuzione dell'istruzione Wait For Interrupt (WFI) o dal Wait For Event (WFE). In caso dell'istruzione WFI, il core Cortex si riattiverà e servirà l'interrupt. Una volta servito l'interrupt ci sono due possibilità, la prima è che la CPU ritorni all'esecuzione del codice sorgente, oppure settando nel registro di sistema di controllo il bit SLEEPONEXIT, il core Cortex entrerà automaticamente nella modalità 'sleep' dopo che ha servito l'interrupt. Ciò ci permette di risparmiare potenza. L'istruzione WFE permette al Cortex di riprendere l'esecuzione dal punto in cui era stato messo in modalità sleep cioè all'avvento di un interrupt la CPU non salta alla service routine. Il 'risveglio' può essere causato da un interrupt di una periferica non abilitato nell'NVIC. Le istruzioni WFI e WFE non possono essere programmate in linguaggio C, ma il Thumb-2 instruction set presenta delle macro che possono essere inserite all'interno del codice C per richiamare tali istruzioni.

3.2.8 Floating-point unit (FPU)

L'FPU supporta pienamente tutte le operazioni in singola precisione. Fornisce inoltre le conversioni tra formati di dati a virgola fissa e virgola mobile. La FPU fornisce funzionalità di calcolo in floating point che sono compatibili con lo standard ANSI / IEEE Std 754-2008. Lo standard IEEE per il calcolo in virgola mobile (IEEE 754-2008) è lo standard più diffuso nel campo del calcolo automatico. Questo standard definisce il formato per la rappresentazione dei numeri in virgola mobile, ed un set di operazioni effettuabili su questi. Specifica inoltre quattro metodi di arrotondamento e ne descrive cinque eccezioni.

3.3 Microcontrollori STM32

La famiglia STM32 di microcontrollori flash a 32 bit basati su processore ARM Cortex-M, comprende una serie di prodotti a 32 bit che abbinano prestazioni elevate, funzionalità in tempo reale, elaborazione di

segnali digitali e funzionamento a bassa potenza e bassa tensione, pur mantenendo caratteristiche di totale integrabilità e facilità di sviluppo.

3.3.1 Serie STM32 F4

La serie STM32 F4 basata su ARM® Cortex™-M4 è un'estensione del portafoglio STM32, leader del settore, in grado di offrire prestazioni ancora più elevate. Questi MCU si avvalgono della tecnologia NVM a 90 nm di ST e di ART Accelerator di ST per raggiungere i massimi punteggi di benchmark del settore per microcontrollori basati su Cortex-M con 210 DMIPS e punteggio Coremark 469 eseguendo il benchmark dalla memoria flash a una frequenza di funzionamento di 168 MHz. Le istruzioni DSP e l'unità di calcolo a virgola mobile ampliano il numero di applicazioni nelle quali è possibile utilizzare questi MCU. La serie STM32 F4 è il risultato di una simbiosi perfetta delle capacità di controllo in tempo reale di un MCU e delle prestazioni di elaborazione dei segnali di un DSP e quindi arricchisce il portafoglio STM32 con una nuova classe di dispositivi: controllori per segnali digitali (DSC).



Figura 8 Schema a blocchi del microcontrollore ARM Cortex M4

Quando il microcontrollore viene avviato, un oscillatore interno (RC) a 16 MHz è selezionato di default come clock della CPU, questo oscillatore è costruito per offrire un'accuracy dell'1% nell'intero range di temperatura di funzionamento. Nelle nostre applicazioni possiamo utilizzare come sorgente del Clock sia l'oscillatore RC che un oscillatore esterno, che può avere una frequenza di funzionamento tra i 4 MHz ed i 26 MHz. Se si seleziona un clock esterno quest'ultimo può essere monitorato, in maniera tale che se dovesse essere rilevato un malfunzionamento il sistema passa automaticamente all'oscillatore interno (RC) generando un interrupt software quando quest'ultimo viene abilitato. La sorgente di clock va in ingresso ad un PLL che permette di portare la frequenza di funzionamento fino ad un massimo di 168 MHz. Diversi Prescaler consentono la configurazione di tre bus AHB (Advanced High Bus), dell 'high

speed APB2 (Advanced peripheral bus two) e del low speed APB1 (Advanced peripheral bus one). La massima frequenza dei tre bus AHB è 168 MHz, mentre la massima frequenza agli high speed APB è 84 MHz e ai low speed APB è 42 MHz. Tre differenti sorgenti di clock possono essere utilizzate per pilotare il clock di sistema (SYSCLK):

- HSI oscillator clock
- HSE oscillator clock
- Main PLL (PLL) clock

I dispositivi hanno le due seguenti fonti di clock secondarie:

- Low-speed internal RC (LSI RC) a 32 kHz che può guidare il watch dog, L'RTC può essere utilizzato come sveglia dalle modalità di stop e standby.
- Low-speed external crystal (LSE crystal) 32.768 kHz che può pilotare eventualmente l'orologio RTC (RTCCCLK)

Per ottimizzare i consumi di energia ogni sorgente di clock può essere attivata o disattivata in modo indipendente quando non viene utilizzata.

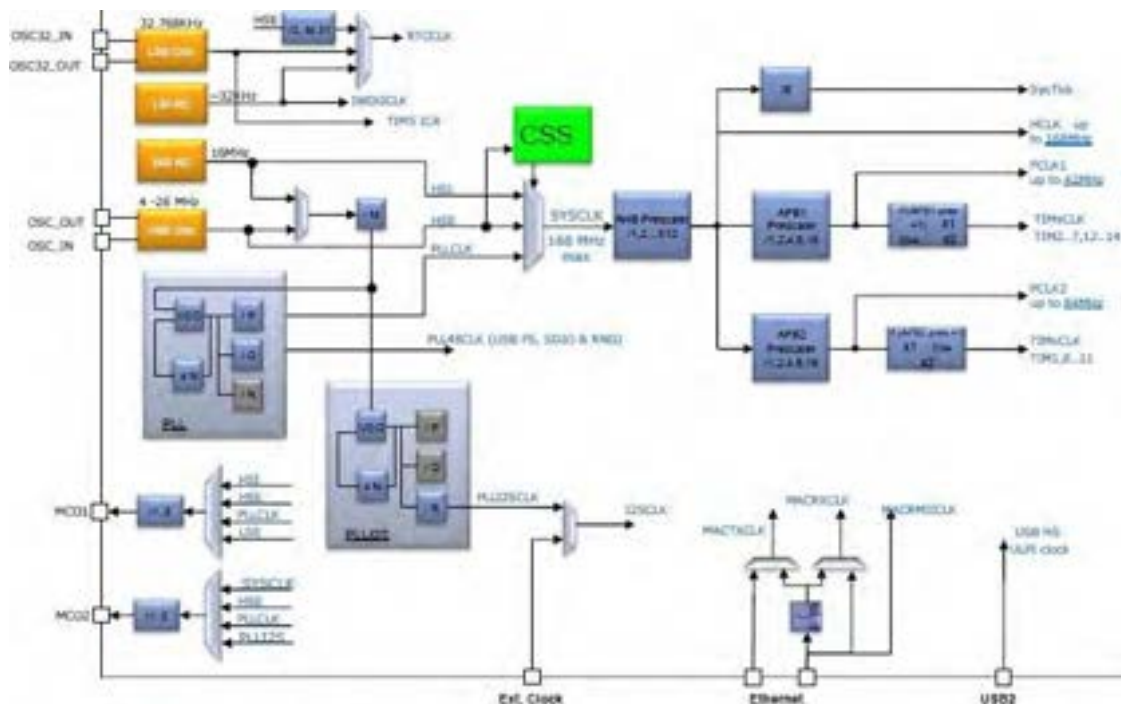


Figura 9 Schema del Clock del microcontrollore ARM Cortex M4

Il Clock controller fornisce un elevato grado di flessibilità nella scelta dell'oscillatore e della sua frequenza per avviare il core e le periferiche, allo stesso tempo garantisce la frequenza appropriata per periferiche che richiedono un clock specifico come l'Ethernet, USB, OTG, FS, I2S, SDIO ed altre.

3.4 Modem Powerline

In telecomunicazioni powerline (in inglese power line communication o PLC, in italiano onde convogliate) è una tecnologia per la trasmissione di voce o dati che utilizza la rete di alimentazione elettrica come mezzo trasmissivo.

Si realizza sovrapponendo al trasporto di corrente elettrica, continua o alternata a bassa frequenza (50Hz in Europa e gran parte dell'Asia e dell'Africa, 60 Hz in altre regioni del mondo), un segnale a

frequenza più elevata che è modulato dall'informazione da trasmettere. La separazione dei due tipi di correnti si effettua grazie al filtraggio e separazione degli intervalli di frequenze utilizzate.

La tecnica è utilizzata da decenni, prima dell'introduzione della telefonia mobile, per trasmissioni con treni in marcia (usando le linee di potenza, ad esempio dalla Telettra), per controllare apparati elettrici tramite la propria rete di alimentazione, per leggere contatori elettrici remotamente, per sistemi interfonici casalinghi ecc. Più recentemente è utilizzata per dare accesso dati (per esempio Internet) alle case tramite la rete elettrica senza necessità di accessi specifici per cavo coassiale o radio.

3.4.1 Utilizzi della tecnologia powerline

Questa tecnologia viene utilizzata per la domotica con l'utilizzo di vari standard. Quelli più datati come l'X10 (diffuso soprattutto negli Stati Uniti) consentono trasmissioni con banda limitata e sono dedicati a semplici automazioni domestiche come l'accensione di luci o la realizzazione di impianti antifurto. Richiedono l'installazione di appositi moduli di interfaccia all'interno delle prese e dei comandi. Il vantaggio rispetto ai sistemi tradizionali sta, oltre che nel risparmio di parte del cablaggio, nella possibilità di modificare con estrema semplicità il funzionamento dell'impianto e nella possibilità di realizzare funzionalità "intelligenti".

Una tecnologia powerline molto diffusa in tutto il mondo ed anche in Italia è quella basata sul protocollo LonWorks, ora anche standard ISO 14908-1-2-3-4, infatti su questo protocollo è basato il contatore che Enel installa da qualche anno che è in grado di fare la telelettura e le modifiche contrattuali da remoto [1].

Ultimamente è stato proposto in commercio un nuovo sistema domotico in tecnologia powerLine, denominato "PowerDom", [2] dalla D-Tech Electronic s.r.l., che sfrutta, per la comunicazione dati, un segnale in modulazione FSK con frequenza variabile tra 66 kHz e 132 kHz e velocità di trasmissione dati di 4800 bit/s. Inoltre il protocollo di comunicazione prevede un efficace algoritmo di controllo e correzione degli errori in grado di rendere affidabile la comunicazione per la realizzazione di tutte le funzioni di un impianto domotico di classe 1: sistema di allarme tecnico e anti-intrusione, controllo accessi, controllo remoto GSM, controllo luci e climatizzazione, gestione utenze e risparmio energetico, automazione di luci e oscuranti.

Nell'ambito delle reti locali per le abitazioni e/o piccoli uffici, è stato creato il consorzio HomePlug che consente la creazione dell'equivalente di una rete ethernet tramite l'utilizzo del normale impianto elettrico casalingo. Le specifiche HomePlug nella versione 1.0 prevedevano una velocità massima teorica di 14 Mbit/s, che risultavano in un effettivo di 5-6 Mbit/s. In seguito sono uscite delle estensioni non standard che prevedevano una velocità massima teorica di 85 Mbit/s ed un effettivo di 16-25 Mbit/s con prestazioni simili, quindi, a quanto fornito da soluzioni wireless 802.11g. Da ottobre 2006 è stato introdotto lo standard HomePlug AV, che supporta una velocità teorica di 200 Mbit/s con un effettivo variabile tra i 70 e 110 Mbit/s. Con questo sistema di trasmissione diviene possibile impiegare servizi quali flussi video in alta definizione prima non possibili. Come per i sistemi wireless, la velocità effettiva è dipendente da molti fattori quali la qualità dell'impianto, la struttura, la presenza di eventuali fonti di disturbo sull'impianto elettrico.

3.4.2 Trasmissioni esterne

Le compagnie di servizi pubblici utilizzano coppie di condensatori (con cui si realizzano filtri passa alto) per connettere trasmettitori radio a bassa frequenza a conduttori elettrici della rete elettrica. Le frequenze usate variano da 30 a 300 kHz con trasmettitori amplificati che elevano il segnale a centinaia di watt di potenza. Questi segnali possono essere diffusi sulle linee dell'alta tensione da uno a tre conduttori ed ogni linea dell'alta tensione (con uno, due o tre conduttori) può supportare molti canali di comunicazione powerline.

Alle sottostazioni vengono applicati dei filtri per evitare che le frequenze della portante attraversino gli apparati della centrale e per evitare che errori di trasmissione dovuti alla distanza non interessino neanche segmenti isolati di rete.

Questi circuiti sono usati per il controllo dei dispositivi di commutazione e per la protezione delle linee di trasmissione.

Ad esempio, un dispositivo di protezione può essere usato per liberare una linea se un errore è rilevato fra i suoi due nodi terminali, o per lasciarla operare se l'errore interessa tutte le linee di trasmissione (e non dipende dai singoli nodi, ma dalla rete). Da un lato gli operatori utilizzano basse tensioni e, in misura crescente, la fibra ottica; dall'altro powerline è una soluzione a basso costo nei casi in cui uno scavo in fibra ottica non sia economicamente conveniente.

3.4.3 Gli Standard

Per le installazioni domestiche, lo standard è deciso dall'Homeplug Powerline Alliance, consorzio di 70 imprese costituito nel 2000. I primi standard furono pubblicati nel 2001, l'Homeplug 1.0 e l'HomePlug 1.0 Turbo, per trasferire dati a una velocità teorica rispettivamente di 15 e 85 Mbit/secondo. Dopo il 2005, furono pubblicati gli standard per alte velocità della Universal Powerline Association e l'HomePlug Powerline AV con velocità teoriche di 200-500 Mbps, con supporti alla Web TV e HD TV, streaming audio e video, telefonia VoIP.

Dopo alcuni anni di competizione serrata sugli standard, i due standard al di sopra i 200 Mbs sono confluiti in un unico standard internazionale IEEE P1901 Powerline AV, cui rispondono ormai tutti i prodotti in commercio come HPAV Powerline HomePlug AV.

Prive di schermature, le linee elettriche attraversate dai segnali digitali Powerline irradiano nell'ambiente circostante in una banda fino a 50 Megahertz. La tecnologia Powerline trasmette una serie di sottoportanti nella banda che va da 2 a 30 Megahertz. Per minimizzare il rumore e le interferenze, le sottoportanti sono equamente spaziate e distribuite, i canali occupano bande relativamente piccole e i canali adiacenti sono ortogonali fra loro (Orthogonal Division Frequency Multiplexing).

I progressi nel tempo della tecnica di modulazione del segnale sono dovuti all'aumento delle sottoportanti e dei bit per carrier veicolati da ciascuna sottoportante.

La scelta delle sottoportanti da utilizzare e della tecnica di modulazione è dinamica, in funzione delle condizioni delle linee elettriche ritornate dai plug riceventi a quello emittente. In condizioni difficili, lo standard utilizza la modulazione RobO (Robust operation) di tipo DBPSK, meno efficiente ma molto resistente ai disturbi [5].

4 Tecnologie informatiche per lo sviluppo del Software

Nel seguente paragrafo vengono introdotte le tecnologie informatiche utilizzate per lo sviluppo del software relativo al concentratore di dati, in particolare verrà fatta una descrizione dettagliata dei linguaggi di programmazione come il Python, MySQL, HTML e PHP.

4.1 Python

Il Python è un linguaggio di programmazione ad alto livello, orientato agli oggetti, adatto per sviluppare applicazioni distribuite, di *scripting*, di computazione numerica e *system testing*. Python è spesso paragonato a Tcl, Perl, Java, JavaScript, Visual Basic o Scheme. Fu ideato da Guido van Rossum all'inizio degli anni novanta. Il nome fu scelto per via della passione di van Rossum per i Monty Python e per la loro serie televisiva Monty Python's Flying Circus.

4.1.1 Logica di funzionamento

Python è un linguaggio multi-paradigma, che fa della dinamicità, semplicità e flessibilità i suoi principali obiettivi. Supporta il paradigma *object oriented*, la programmazione strutturata e molte caratteristiche di programmazione funzionale e riflessione. Le caratteristiche più facilmente riconoscibili di Python sono le variabili non tipizzate e l'uso dell'indentazione per la definizione dei blocchi. Altre caratteristiche distintive sono l'*overloading* di operatori e funzioni tramite *delegation*, la presenza di un ricco assortimento di tipi e funzioni di base e librerie standard, sintassi avanzate quali *slicing* e *list comprehension*. Il controllo dei tipi è comunque forte (*strong typing*) e viene eseguito al *runtime* (*dynamic typing*). In altre parole una variabile è un contenitore al quale viene associata un'etichetta (il nome) che può essere associata a diversi contenitori anche di tipo diverso durante il suo tempo di vita. Usa un *garbage collector* per la liberazione automatica della memoria. Python ha qualche similarità con Perl, ma i suoi progettisti hanno scelto la via di una sintassi più essenziale e uniforme, con l'obiettivo di aumentare la leggibilità del codice. Come Perl, spesso è classificato come linguaggio di *scripting*, ma pur essendo utile per scrivere script di sistema (in alternativa ad esempio a *bash*), la grande quantità di librerie disponibili e la facilità con cui questo linguaggio permette di scrivere software modulare, favorisce anche lo sviluppo di applicazioni molto complesse.

4.1.2 Tipi di dati e strutture

Essendo Python a tipizzazione dinamica, tutte le variabili sono in realtà puntatori ad oggetto (*reference*) mentre gli oggetti sono invece dotati di tipo. Ad esempio, una variabile assegnata ad un intero, un istante dopo può essere assegnata una stringa o una lista. In Python c'è un moderato controllo dei tipi al momento del *runtime*. Si ha conversione implicita per i tipi numerici, che permette ad esempio di moltiplicare un numero complesso per un intero, ma non c'è, ad esempio, conversione implicita tra numeri e stringhe, per cui un numero è un argomento non valido per le operazioni su stringa (come avviene in PHP). Esso mette a disposizione un gran numero di tipi base, essenzialmente tipi numerici e contenitori. Caratteristica distintiva è il supporto nativo, oltre che ai classici tipi quali interi, *floating point*, stringhe, anche a tipi più evoluti quali interi a grandezza arbitraria, numeri complessi, liste, insiemi, dizionari, con delle comode sintassi per la costruzione degli stessi. Non è invece previsto un tipo specifico per i caratteri. Molti altri tipi sono importabili da librerie standard, e nuovi tipi possono essere creati attraverso il costrutto di classe.

4.1.3 Tipi numerici

I tipi interi (*int*) e *floating point* (*float*) hanno una precisione dipendente dalla macchina e dall'implementazione dell'interprete (in genere 32 e 64 bit). Sono supportati nativamente numeri interi arbitrariamente grandi (*long*, che diventano l'opzione di default per gli interi a partire da 3.0) e numeri complessi (*complex*). Sono definiti tutti i principali operatori logici e aritmetici fra numeri, compreso l'elevamento a potenza. Il tipo booleano (*bool*) appartiene anch'esso alla categoria dei numeri. Dalla versione 2.4 sono disponibili come libreria anche i numeri decimali (*decimal*), ossia numeri con la virgola a precisione illimitata come quelli disponibili in REXX o in Cobol, che non soffrono di problemi di arrotondamento e stabilità tipici dei numeri *floating point* classici.

4.1.4 Contenitori

Python considera in generale come contenitori gli oggetti che prevedono la possibilità di iterare su un insieme di elementi, e dunque utilizzabili all'interno di contesti di iterazione quali il ciclo *for* e funzioni quali *somma*, *ricerca* e *ordinamento*. I contenitori in genere permettono di contenere dati di tipo eterogeneo. Per quanto riguarda i contenitori standard propriamente detti, essi sono classificabili come sequenze, insiemi e dizionari. I contenitori seguono una filosofia comune e condividono gran parte dei metodi. Le sequenze sono contenitori ordinati, che condividono dei metodi basati sull'ordinamento,

l'indicizzazione intera e la creazione di sottosequenze tramite *slicing*. Le liste (list) sono sequenze estendibili, invece le *tuple* sono sequenze immutabili. Anche le stringhe (str e unicode) sono considerate sequenze. A partire da Python 3.0, i tipi str e unicode sono unificati e compare il tipo byte, equivalente grosso modo a una stringa binaria. Sono supportate tutte le operazioni classiche sulle stringhe, come concatenamento, formattazione, ricerca, sostituzione, ecc. Le stringhe in Python sono sequenze immutabili, cosicché qualsiasi operazione che in qualche modo potrebbe alterare una stringa (come ad esempio la sostituzione di un carattere) restituirà invece una nuova stringa (come avviene in Java). Altri contenitori di grande utilità sono i dizionari (*dict*), conosciuti in altri contesti con il nome di *hash table* oppure array associativi. Esiste una sintassi per la creazione di dizionari, i cui elementi sono specificati da una coppia di dati separati da due punti ":". Il primo elemento della coppia rappresenta l'indice, detto "chiave", mentre il secondo è il suo valore corrispondente. Ogni elemento è detto coppia chiave-valore.

4.1.5 Organizzazione ad oggetti

Il sistema dei tipi Python è ben integrato con il sistema delle classi. Anche se i tipi base non sono precisamente classi, una classe può ereditare da essi. In questo modo è possibile estendere stringhe, dizionari e perfino gli interi. È inoltre supportata l'ereditarietà multipla. Vengono supportate anche funzionalità estensive di introspezione sui tipi e sulle classi. I tipi e le classi sono a loro volta oggetti che possono essere esplorati e confrontati. Gli attributi sono gestiti in un dizionario.

4.1.6 Decoratori

Un decoratore è un qualsiasi oggetto di Python chiamabile, usato per modificare una funzione, un metodo o una definizione di classe. Un decoratore è passato all'oggetto e ritorna l'oggetto modificato. Sono ispirati in parte dalla notazione Java, e hanno una sintassi simile.

4.1.7 Indentazione

Un aspetto inusuale del Python è il metodo che usa per delimitare i blocchi di programma, che lo rende unico fra i linguaggi più diffusi. Nei linguaggi derivati dall'ALGOL come Pascal, C e Perl, i blocchi di codice sono indicati con le parentesi oppure con parole chiave (il C ed il Perl usano { }, il Pascal usa *begin* ed *end*). In questi linguaggi è solo una convenzione degli sviluppatori il fatto di indentare il codice interno ad un blocco, per metterlo in evidenza rispetto al codice circostante. Python, invece, deriva il suo sistema di indentazione dal meno noto linguaggio di programmazione Occam, invece di usare parentesi o parole chiave, usa l'indentazione stessa per indicare i blocchi nidificati (si può usare sia una tabulazione o un numero arbitrario di spazi bianchi, ma lo standard Python è di 4 spazi bianchi), figura 10.

```
class Persona(object):
    # Inizializzatore della classe
    def __init__(self, nome, cognome):
        self.name = nome
        self.surname = cognome

    def fullname(self):
        full = "Sig. " + self.surname + " " + self.name
        return full

persona = Persona('Mario', 'Rossi')
```

```
print(persona.fullname())
```

Figura 10

Codice sorgente di un programma scritto in Python

4.1.8 Gestione delle interruzioni

Python supporta e usa estensivamente la gestione delle eccezioni come mezzo per segnalare e controllare eventuali condizioni di errore, incluse le eccezioni generate dagli errori di sintassi. Le eccezioni permettono un controllo degli errori più conciso ed affidabile rispetto a molti altri modi possibili che sono usati in genere per segnalare errori o situazioni anomale. Le eccezioni sono *thread-safe*, non sovraccaricano il codice sorgente come fanno invece i controlli sui valori di errore restituiti e, inoltre, possono facilmente propagarsi verso l'alto nello *stack* delle chiamate a funzione quando un errore deve venire segnalato ad un livello più alto del programma. Con la gestione delle eccezioni i controlli preventivi sono sostituiti da un più agevole meccanismo che permette di eseguire direttamente l'azione desiderata e catturare separatamente le eventuali eccezioni che si possono verificare. Oltre che per la gestione degli errori, in alcune occasioni le eccezioni sono usate in Python anche per il controllo di flusso. L'operazione di iterazione (e di conseguenza il ciclo `for`) è ad esempio basata su una segnalazione tramite eccezione.

4.1.9 Libreria Standard

Python ha una vasta libreria standard, il che lo rende adatto a molti impieghi. Oltre ai moduli della libreria standard se ne possono aggiungere altri scritti in C oppure Python per soddisfare le proprie esigenze particolari. Tra i moduli già disponibili ve ne sono alcuni per scrivere applicazioni web (sono supportati MIME, HTTP e tutti gli altri standard Internet). Sono disponibili anche moduli per creare applicazioni con interfaccia grafica, per connettersi a database relazionali, per usare le espressioni regolari. La libreria standard è uno dei punti forti di Python. Essa infatti è compatibile con tutte le piattaforme, ad eccezione di poche funzioni, segnalate chiaramente nella documentazione come specifiche di una piattaforma particolare.

4.1.10 Altre caratteristiche

Sebbene Python venga in genere considerato un linguaggio interpretato, in realtà il codice sorgente non viene convertito direttamente in linguaggio macchina poiché, invece, passa prima da una fase di pre-compilazione in *bytecode*. Lo stesso *bytecode* viene quasi sempre riutilizzato dopo la prima esecuzione del programma, evitando così di dover ogni volta interpretare il sorgente ed incrementando di conseguenza le prestazioni. Inoltre è possibile distribuire programmi Python direttamente in *bytecode*, saltando a piè pari la fase di interpretazione da parte dell'utilizzatore finale, e permettendo anche di avere programmi Python a sorgente chiuso. Come il Lisp e a differenza del Perl, l'interprete Python supporta anche un modo d'uso interattivo, attraverso il quale è possibile inserire codice direttamente da un terminale, visualizzando immediatamente il risultato. L'interprete è inoltre contenuto nella libreria standard e come in molti altri linguaggi interpretati è possibile far valutare stringhe arbitrarie nel contesto corrente. È però possibile passare all'interprete anche un contesto completamente diverso, sotto forma di liste che contengono l'elenco dei simboli definiti. Python dispone anche di un *framework* per lo *unit testing* che supporta lo sviluppo di test unitari automatici.

4.1.11 Prestazioni

Se paragonato ai linguaggi compilati *statically typed*, come ad esempio il C, la velocità di esecuzione non è uno dei punti di forza di Python, specie nel calcolo matematico. Esiste un'estensione, Psyco, che è una

sorta di compilatore JIT e permette di velocizzare in modo notevole alcuni tipi di codice, specialmente l'implementazione di algoritmi, pur pagando un prezzo in termini di memoria utilizzata. Comunque, anche in questo modo un qualunque compito che prevede numerosi calcoli puri non è adatto ad un programma Python. Le performance di Python sono invece allineate o addirittura superiori ad altri linguaggi interpretati, quali PHP e Ruby, e in certe condizioni può rivaleggiare anche con Java. Non va inoltre dimenticato che Python permette di aggirare in modo facile l'ostacolo delle performance pure, è infatti relativamente semplice scrivere un'estensione in C o C++ e poi utilizzarla all'interno di Python, sfruttando così l'elevata velocità di un linguaggio compilato solo nelle parti in cui effettivamente serve e sfruttando invece la potenza e versatilità di Python per tutto il resto del software. Con opportune accortezze e utilizzando solo moduli standard, in alcuni casi può raggiungere una velocità di esecuzione pari ad un codice equivalente in C, grazie ad una serie di ottimizzazioni interne della PVM.

4.1.12 Altre implementazioni

Sono attualmente disponibili le seguenti implementazioni di Python:

- Jython: in linguaggio Java.
- IronPython: per la piattaforma Microsoft.NET
- Python for S60: per dispositivi con sistema operativo Symbian.
- PyPy: scritta in Python stesso, ha tra i principali obiettivi la semplificazione dello sviluppo del linguaggio e la sua ottimizzazione in termini prestazionali.
- Scripting Layer for Android (SL4A): per il sistema operativo Android che include l'interprete Python 2.6.2.

4.2 MySQL

MySQL, definito Oracle MySQL, è un *Relational Database Management System* (RDBMS), composto da un client con interfaccia a riga di comando e un server, entrambi disponibili sia per sistemi Unix o Unix-like come GNU/Linux che per Windows, anche se prevale un suo utilizzo in ambito Unix. Possiede delle interfacce per diversi linguaggi, compreso un driver ODBC, due driver Java, un driver per Mono e .NET ed una libreria per Python.

Il codice sorgente di MySQL era inizialmente di proprietà della società MySQL AB, veniva però distribuito con la licenza GNU GPL oltre che con una licenza commerciale. Fino alla versione 4.0, una buona parte del codice del client era licenziato con la GNU LGPL e poteva dunque essere utilizzato anche per applicazioni proprietarie. Dalla versione 4.1 in poi, anche il codice del client è distribuito sotto GNU GPL. Esiste peraltro una clausola estensiva che consente l'utilizzo di MySQL con una vasta gamma di licenze libere. Nel luglio 2007 la società svedese MySQL AB aveva 385 dipendenti in numerosi paesi. I suoi principali introiti provenivano dal supporto agli utilizzatori di MySQL tramite il pacchetto Enterprise, dalla vendita delle licenze commerciali e dall'utilizzo da parte di terzi del marchio MySQL. Il 16 gennaio 2008 Sun Microsystems ha acquistato la società per un miliardo di dollari, stimando il mercato del database in 15 miliardi di dollari. Il 20 aprile 2009 alla stessa Sun Microsystems è stata proposta l'acquisizione da parte di Oracle per 7,4 miliardi di dollari. L'accordo, approvato dall'antitrust USA, è poi passato al vaglio degli organi corrispondenti dell'Unione Europea, preoccupati dal conflitto di interessi costituito dai database commerciali Oracle rispetto a MySQL. Il padre di MySQL, Michael Widenius, ha lanciato una petizione online per opporsi alla fusione. Nonostante ciò l'Unione Europea ha dato parere favorevole, e l'acquisizione è stata completata il 27 gennaio 2010. Il ramo 5.5 è il primo a includere estensioni non Open Source, disponibili solo nella versione Enterprise.

4.2.1 Amministrazione

Esistono diversi tipi di MySQL Manager, ovvero di strumenti per l'amministrazione di MySQL. Uno dei programmi più popolari per amministrare i database MySQL è phpMyAdmin (richiede un server web

come Apache HTTP Server ed il supporto del linguaggio PHP). Si può utilizzare facilmente tramite un qualsiasi browser. Alcune offerte di terze parti sono HeidiSQL, SQLYog o Toad for MySQL. In alternativa la stessa MySQL AB offre programmi quali MySQL Administrator (amministrazione del database, degli utenti, operazioni pianificate, carico del server,...) e MySQL Query Browser (esecuzione di svariati tipi di query), MySQL Migration Toolkit per importare da altri DBMS. Per il disegno e la modellazione di database MySQL esiste MySQLWorkbench: integra il disegno, la modellazione, la creazione e l'aggiornamento di database in un unico ambiente di lavoro. In passato veniva sviluppato anche MySQLcc (MySQL control center), sostituito da MySQL Query Browser. In alternativa a gestori grafici è possibile utilizzare direttamente la riga di comando, preferita da utenti/amministratori esperti.

4.2.2 Sviluppo

Fino a qualche anno fa lo sviluppo del programma era opera soprattutto dei suoi sviluppatori iniziali: David Axmark, Allan Larsson e Michael Widenius. Quest'ultimo era il principale autore del codice, oltre che principale socio della società, e tuttora coordina il progetto, tra l'altro vagliando i contributi che pervengono dai volontari. I contributi vengono accettati a condizione che il loro autore condivida i diritti d'autore con la società. Da luglio 2007 la società impiega un centinaio di sviluppatori a tempo pieno.

4.2.3 Tipi tabelle MySQL (Storage Engine)

In MySQL una tabella può essere di diversi tipi (o Storage Engine). Ogni tipo di tabella presenta proprietà e caratteristiche differenti (transazionale o meno, migliori prestazioni, diverse strategie di *locking*, funzioni particolari, ecc.). Esiste poi un'API che si può utilizzare per creare in modo relativamente facile un nuovo tipo di tabella, che poi si può installare senza dover ricompilare o riavviare il server.

Storage Engine ufficiali

I tipi di tabella predefiniti sono:

- MyISAM
- InnoDB
- Memory
- Merge
- NDB o ClusterDB
- CSV
- Federated
- Archive
- Blackhole
- Falcon
- Aria

Esistono anche *storage engine* prodotti da terze parti. Eccone alcuni:

- XtraDB
- OQGRAPH
- solidDB
- RitmarkFS
- Distributed Data Engine
- SphinxSE
- CassandraSE
- mdbtools
- BrightHouse

4.2.4 Compatibilità

MySQL, essendo scritto in linguaggio C e C++, è disponibile su molti differenti sistemi operativi e le piattaforme di riferimento sono Linux e Solaris. La documentazione di MySQL offre comunque un aiuto per chi avesse bisogno di provare a compilare il software su qualsiasi sistema operativo discretamente diffuso. MySQL utilizza anche i *tool automake*, *autoconf* e *libtools* per aumentare la compatibilità. Infine, il linguaggio SQL di MySQL comprende numerose estensioni che sono tipiche di altri DBMS, quali PostgreSQL, Oracle e Sybase. In questo modo le *query* non standard scritte per altri DBMS in alcuni casi funzioneranno senza problemi.

4.3 PHP

PHP ("*Hypertext Preprocessor*", preprocessore di ipertesti, originariamente acronimo di "Personal Home Page") è un linguaggio di programmazione interpretato, originariamente concepito per la programmazione web ovvero per la realizzazione di pagine web dinamiche. L'interprete ha una licenza open source e libera (ma incompatibile con la GPL). Attualmente è utilizzato principalmente per sviluppare applicazioni web lato server, ma può essere usato anche per scrivere script a riga di comando o applicazioni stand-alone con interfaccia grafica.

Nato nel 1994 ad opera del danese Rasmus Lerdorf, PHP era in origine una raccolta di script CGI che permettevano una facile gestione delle pagine personali. Il significato originario dell'acronimo era Personal Home Page. Il pacchetto originario venne in seguito esteso e riscritto dallo stesso Lerdorf in C, aggiungendo funzionalità quali il supporto al database mSQL e prese a chiamarsi PHP/FI, dove FI sta per Form Interpreter (interprete di *form*), prevedendo la possibilità di integrare il codice PHP nel codice HTML in modo da semplificare la realizzazione di pagine dinamiche. In quel periodo, 50.000 domini Internet annunciavano di aver installato PHP. A questo punto il linguaggio cominciò a godere di una certa popolarità tra i progetti open source del web, e venne così notato da due giovani programmatori: Zeev Suraski e Andi Gutmans. I due collaborarono nel 1998 con Lerdorf allo sviluppo della terza versione di PHP (il cui acronimo assunse il significato attuale) riscrivendone il motore che fu battezzato Zend da una contrazione dei loro nomi. Le caratteristiche chiave della versione PHP 3.0, frutto del loro lavoro, erano la straordinaria estensibilità, la connettività ai database e il supporto iniziale per il paradigma a oggetti. Verso la fine del 1998 PHP 3.0 era installato su circa il 10% dei web server presenti su Internet. PHP diventò a questo punto talmente maturo da competere con ASP, linguaggio lato server analogo a PHP sviluppato da Microsoft, e cominciò ad essere usato su larga scala. La versione 4 di PHP venne rilasciata nel 2000 e prevedeva notevoli migliorie. Attualmente siamo alla quinta versione, sviluppata da un team di programmatori, che comprende ancora Lerdorf, oltre a Suraski e Gutmans. La popolarità del linguaggio PHP è in costante crescita grazie alla sua flessibilità: nel giugno 2001, ha superato il milione di siti che lo utilizzano. Nell'ottobre 2002, più del 45% dei server Apache usavano PHP. Nel gennaio 2005 è stato insignito del titolo di "Programming Language of 2004" dal TIOBE Programming Community Index, classifica che valuta la popolarità dei linguaggi di programmazione sulla base di informazioni raccolte dai motori di ricerca. Nel 2005 la configurazione LAMP (Linux, Apache, MySQL, PHP) supera il 50% del totale dei server sulla rete mondiale. Nel 2008 PHP 5 è diventata l'unica versione stabile in fase di sviluppo. A partire da PHP 5.3.0, PHP implementa una funzione chiamata "*late static binding*" che può essere utilizzata per fare riferimento alla classe chiamata in un contesto di eredità statica. A partire dal 5 febbraio 2008, a causa dell'iniziativa GoPHP5, sostenuta da una serie di sviluppatori PHP, molti dei progetti open source di alto profilo cessano di supportare PHP 4 nel nuovo codice e promuovono il passaggio da PHP 4 a PHP 5.

4.3.1 Caratteristiche

PHP riprende per molti versi la sintassi del C, come peraltro fanno molti linguaggi moderni, e del Perl. È un linguaggio a tipizzazione debole e dalla versione 5 migliora il supporto al paradigma di

programmazione ad oggetti. Certi costrutti derivati dal C, come gli operatori fra bit e la gestione di stringhe come array, permettono in alcuni casi di agire a basso livello, tuttavia è fondamentalmente un linguaggio di alto livello, caratteristica questa rafforzata dall'esistenza delle sue moltissime API, oltre 3.000 funzioni del nucleo base. PHP è in grado di interfacciarsi a innumerevoli database tra cui MySQL, PostgreSQL, MariaDB, Oracle, Firebird, IBM DB2, Microsoft SQL Server, solo per citarne alcuni, e supporta numerose tecnologie, come XML, SOAP, IMAP, FTP, CORBA. Si integra anche con altri linguaggi/piattaforme quali Java e .NET e si può dire che esista un *wrapper* per ogni libreria esistente, come CURL, GD, Gettext, GMP, Ming, OpenSSL ed altro. Fornisce un'API specifica per interagire con Apache, nonostante funzioni naturalmente con numerosi altri web server. È anche ottimamente integrato con il database MySQL, per il quale possiede più di una API. Per questo motivo esiste un'enorme quantità di script e librerie in PHP, disponibili liberamente su Internet. La versione 5, comunque, integra al suo interno un piccolo database *embedded*, SQLite. Dispone di un archivio chiamato PEAR che mette a disposizione un *framework* di librerie riutilizzabili per lo sviluppo di applicazioni PHP e di PECL che raccoglie tutte le estensioni conosciute scritte in C. PHP non ha ancora un supporto nativo per le stringhe Unicode o *multibyte*, il supporto Unicode è in fase di sviluppo per una futura versione di PHP, e consentirà di usare caratteri non ASCII in stringhe e nomi di funzioni, classi e metodi.

4.3.2 Sicurezza

La percentuale di software non sicuro scritto in PHP, sul totale di tutte le falle nei software elencate dal *Common Vulnerabilities and Exposures*, ammontava al: 12% nel 2003, 20% nel 2004, 28% nel 2005, 43% nel 2006, 36% nel 2007, 34.8% nel 2008, 29.9% nel 2009 e 27.2% nel 2010. La maggior parte di questi punti vulnerabili possono essere sfruttati tramite remoto, ovvero senza accedere al computer che ospita l'applicazione vulnerabile. Le falle più comuni sono dovute al mancato adempimento delle *best practice* nella programmazione e da vulnerabilità presenti in codice scritto in versioni vecchie di PHP.

4.3.3 Gestione dei parametri

Il PHP permette il passaggio di parametri da una pagina all'altra attraverso tre array di variabili: \$_GET, \$_POST e \$_SESSION. Il primo tipo di parametro viene passato tramite la stringa che compare nella barra dell'indirizzo del browser, il secondo viene passato in background, mentre il terzo rimane persistente durante la sessione.

4.3.4 php.ini

Il file di configurazione di PHP, chiamato php3.ini in PHP 3, e poi semplicemente php.ini nelle versioni successive, è letto all'avvio dell'interprete del linguaggio e fornisce le impostazioni dei vari moduli con cui l'interprete è stato compilato. Nella versione server modulare di PHP, questo avviene solo una volta, all'avvio del web server. Per le versioni CGI e CLI, invece, è invocato ad ogni richiesta. Per visualizzare tutte le opzioni di configurazione è possibile utilizzare la funzione phpinfo().

4.3.5 Connessione con database MySQL usando l'estensione MySQLi

Il PHP permette l'elaborazione di dati da database MySQL, i quali verranno richiesti, inseriti e modificati attraverso opportune *"query"*. Tutto ciò è possibile grazie ad alcune estensioni, tra cui spicca per importanza MySQLi. L'estensione MySQLi (dove la "i" sta per *"improved"*) è stata messa a disposizione di PHP per sfruttare alcune nuove funzionalità messe a disposizione dalle versioni di MySQL 4.1.3 e successive ed è disponibile per PHP 5 e release superiori. Risulta essere al momento l'estensione più performante e dinamica.

4.4 HTML

L'Hyper Text Markup Language (HTML) (traduzione letterale: linguaggio a marcatori per ipertesti), in informatica è il linguaggio di *markup* solitamente usato per la formattazione di documenti ipertestuali disponibili nel World Wide Web sotto forma di pagine web.

È un linguaggio di pubblico dominio, la cui sintassi è stabilita dal World Wide Web Consortium (W3C), e che è derivato da un altro linguaggio avente scopi più generici, l'SGML.

L'HTML è stato sviluppato verso la fine degli anni '80 del XX secolo da Tim Berners-Lee al CERN di Ginevra assieme al noto protocollo HTTP che supporta invece il trasferimento di documenti in tale formato. Verso il 1994 ha avuto una forte diffusione in seguito ai primi utilizzi commerciali del web.

Nel corso degli anni, seguendo lo sviluppo di Internet, l'HTML ha subito molte revisioni, ampliamenti e miglioramenti, che sono stati indicati secondo la classica numerazione usata per descrivere le versioni dei software. Attualmente l'ultima versione disponibile è la versione 4.01, resa pubblica il 24 dicembre 1999. Dopo un periodo di sospensione, in cui il W3C si è focalizzato soprattutto sulle definizioni di XHTML (applicazione a HTML di regole e sintassi in stile XML) e dei fogli di stile (CSS), nel 2007 è ricominciata l'attività di specifica con la definizione, ancora in corso, di HTML5. Il W3C ha annunciato che la prima versione di tale standard sarà pronta per fine 2014 e l'html 5.1 per il 2016; la prima Candidate Recommendation venne pubblicata dal W3C il 17 dicembre 2012.

Un'ulteriore ed importante caratteristica di HTML è che esso è stato concepito per definire il contenuto logico e non l'aspetto finale del documento. I dispositivi che possono accedere ad un documento HTML sono molteplici e non sempre dotati di potenti capacità grafiche. Proprio per questo gli sviluppatori di HTML hanno optato per un linguaggio che descrivesse dal punto di vista logico, piuttosto che grafico, il contenuto dei documenti. Questo significa che non esiste alcuna garanzia che uno stesso documento venga visualizzato in egual modo su due dispositivi. Se da una parte questo ha imposto in passato dei forti limiti agli sviluppatori di pagine Web, ha dall'altro garantito la massima diffusione di Internet ed evitato che essa diventasse un medium di élite.

Attualmente i documenti HTML sono in grado di incorporare molte tecnologie, che offrono la possibilità di aggiungere al documento ipertestuale controlli più sofisticati sulla resa grafica, interazioni dinamiche con l'utente, animazioni interattive e contenuti multimediali.

Si tratta di linguaggi come CSS, JavaScript e jQuery, XML, JSON, o di altre applicazioni multimediali di animazione vettoriale o di streaming audio o video.

Al giorno d'oggi molti web designer delegano la scrittura del codice HTML ad applicazioni specifiche, come per esempio i cosiddetti editor WYSIWYG che permettono al designer di occuparsi dell'aspetto grafico finale della pagina mentre il codice vero e proprio viene generato automaticamente. Gli sviluppatori puri preferiscono invece utilizzare direttamente il codice HTML, in modo da avere un maggior controllo sul risultato finale e sulla pulizia del codice scritto, cosa che gli editor WYSIWYG odierni nonostante siano sempre più avanzati non possono sempre garantire, vuoi anche per la diversa resa dei browser sulle parti di codice HTML.

4.4.1 Caratteristiche generali

L'HTML non è un linguaggio di programmazione (in quanto non prevede alcuna definizione di variabili, strutture dati, funzioni, strutture di controllo), ma solamente un linguaggio di formattazione che descrive cioè le modalità di impaginazione o visualizzazione grafica (layout) del contenuto, testuale e non, di una pagina web attraverso *tag* di formattazione. Tuttavia, l'HTML supporta l'inserimento di script e oggetti esterni quali immagini o filmati.

Ogni documento ipertestuale scritto in HTML deve essere contenuto in un file, la cui estensione è tipicamente .htm o .html (l'estensione .htm dipende dal fatto che all'epoca il sistema operativo MS-DOS non permetteva di utilizzare estensioni superiori a 3 caratteri).

La formattazione consiste nell'inserimento nel testo di marcatori o etichette, detti *tag*, che descrivono caratteristiche come la funzione, il colore, le dimensioni, la posizione relativa all'interno della pagina.

4.4.2 Archiviazione e manipolazione

I documenti HTML vengono immagazzinati sui dischi rigidi di macchine elaboratrici (computer-server) costantemente collegate e connesse alla rete Internet. Su queste macchine è installato un software specifico (web server) che si occupa di produrre e inviare i documenti ai browser degli utenti che ne fanno richiesta usando il protocollo HTTP per il trasferimento dati.

Spesso il documento HTML viene generato del tutto o parzialmente tramite un codice eseguibile residente sul server Internet (elaborazione lato server) in grado di interagire con altre applicazioni presenti sul server stesso, come per esempio un database, e inviare poi al browser il risultato finale, realizzando le cosiddette pagine web dinamiche con cui un utente può compiere operazioni interattive avanzate (ad es. filtrare gli articoli all'interno di un catalogo on-line, inviare e registrare dati, fare login ecc..). È il caso di documenti scritti in linguaggi come ASP, PHP, Perl o Java. In altri casi invece alcuni tipi di elaborazione sono operati lato client con linguaggi come Javascript.

4.4.3 Sintassi

Il componente principale della sintassi di questo linguaggio è l'elemento, inteso come struttura di base a cui è delegata la funzione di formattare i dati o indicare al browser delle informazioni.

Ogni elemento è racchiuso all'interno di marcature dette *tag*, costituite da una sequenza di caratteri racchiusa tra due parentesi angolari o uncinate (<>), cioè i segni minore e maggiore (Es.:; il *tag* di questo esempio serve per indicare un ritorno a capo).

Quando il *tag* deve essere applicato a una sezione di testo o di codice, l'ambito di applicazione deve essere delimitato fra un *tag* di apertura ed uno di chiusura (chiusura esplicita), che coincide col *tag* di apertura preceduto da una barra (/) dopo la parentesi angolare aperta (Es.: testo testo testo, in questo caso, il testo compreso tra questi due *tag* verrà visualizzato in grassetto dal browser).

Alcuni *tag* presentano un'applicazione puntuale, come per esempio il *tag* che serve per inserire un'immagine in un determinato punto della pagina, e in quanto tali non richiedono il *tag* di chiusura; in questo caso si parla di *tag* a chiusura implicita. In XHTML, invece, la chiusura implicita è proibita e tutti i *tag* devono essere sempre chiusi esplicitamente tramite un *tag* di chiusura o, nel caso dei *tag* puntuali, usando il carattere '/' alla fine del *tag* stesso (per esempio
). Per questi *tag*, i browser sono in grado di accettare entrambe le modalità, per motivi di compatibilità.

4.4.4 Struttura di un documento HTML

Un documento HTML comincia con l'indicazione della definizione del tipo di documento (*Document Type Definition* o DTD), la quale segnala al browser l'URL delle specifiche HTML utilizzate per il documento, indicando quindi, implicitamente, quali elementi, attributi ed entità si possono utilizzare e a quale versione di HTML si fa riferimento. Di fatto, questa informazione serve al browser per identificare le regole di interpretazione e visualizzazione appropriate per lo specifico documento. Questa definizione deve pertanto precedere tutti i *tag* relativi al documento stesso.

Dopo il DTD, il documento HTML presenta una struttura ad albero annidato, composta da sezioni delimitate da *tag* opportuni che al loro interno contengono a loro volta sottosezioni più piccole, sempre delimitate da *tag*.

La struttura più esterna è quella che delimita l'intero documento, eccetto la DTD, ed è compresa tra i *tag* <html>e</html>.

All'interno dei *tag* <html>lo standard prevede sempre la definizione di due sezioni ben distinte e disposte in sequenza ordinata:

la sezione di intestazione o *header*, delimitata tra i *tag* <head> e </head>, che contiene informazioni di controllo normalmente non visualizzate dal browser, con l'eccezione di alcuni elementi

la sezione del corpo o *body*, delimitata tra i *tag* `<body>` e `</body>`, che contiene la parte informativa vera e propria, ossia il testo, le immagini e i collegamenti che costituiscono la parte visualizzata dal browser.

Al di sotto di questa suddivisione generale, lo standard non prevede particolari obblighi per quanto riguarda l'ordine e il posizionamento delle ulteriori sottosezioni all'interno dell'*header* o del *body*, a parte l'indicazione del rispetto dei corretti annidamenti (le sottosezioni non si devono sovrapporre, ossia ogni sottosezione deve essere chiusa prima di iniziare la sottosezione successiva), lasciando così completa libertà allo sviluppatore o al progettista per quanto riguarda la strutturazione e l'organizzazione successive.

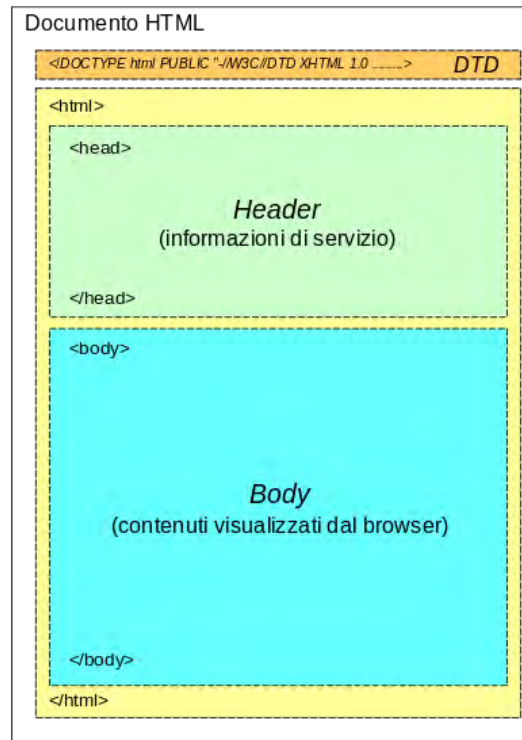


Figura 11 Struttura di un generale documento HTML

4.4.5 Tag della head

I *tag* utilizzati nella sezione *head* sono tipicamente di tipo diverso da quelli utilizzati nella sezione *body*, essendo destinati a scopi differenti. I *tag* utilizzati nella sezione *head* normalmente non vengono visualizzati dal browser ma servono come informazioni di controllo e di servizio quali:

- metadata per convogliare informazioni utili ad applicazioni esterne (es. motori di ricerca) o al browser (es. codifica dei caratteri, utile per la visualizzazione di alfabeti non latini)
- metadata di tipo *http-equiv* per controllare informazioni aggiuntive nel protocollo HTTP
- collegamenti verso file di servizio esterni (CSS, script, icone visualizzabili nella barra degli indirizzi del browser)
- inserimento di script (codice eseguibile) utilizzati dal documento
- informazioni di stile (CSS locali)
- il titolo associato alla pagina e visualizzato nella finestra principale del browser

4.4.6 Tag del body

All'interno della sezione di body, che racchiude la parte visualizzabile del documento, si utilizzano i tag specifici previsti per la formattazione dei contenuti accessibili all'utente finale, ossia per il controllo di:

- intestazioni (titoli di capitoli, di paragrafi eccetera)
- strutture di testo (testo indentato, paragrafi, eccetera)
- aspetto del testo (grassetto, corsivo, eccetera)
- elenchi e liste (numerate, generiche, di definizione)
- tabelle
- moduli elettronici (campi compilabili dall'utente, campi selezionabili, menu a tendina, pulsanti eccetera)
- collegamenti ipertestuali e ancore
- layout generico del documento
- inserimento di immagini
- inserimento di contenuti multimediali (audio, video, animazioni eccetera)
- inserimento di contenuti interattivi (script, applicazioni esterne)

Tra quelli elencati sopra, il tag <a> è quello che gioca un ruolo chiave in quanto descrive un collegamento (o link) a un altro documento ipertestuale consentendo, con un click del mouse o con un'operazione da tastiera, di abbandonare la pagina o il sito che si sta visualizzando e caricare quella indicata dal link, realizzando così la funzione di navigazione tipica dell'uso di Internet [5].

5 Realizzazione del concentratore dati embedded

Uno dei principali scopi della presente attività è stato quello di realizzare un concentratore di dati *embedded* per reti di *Smart Meter*, utilizzando un Raspberry Pi, ovvero un pc di basso costo, interfacciato ad una rete di *Smart Meter* realizzati con microcontrollore STM32F407 che hanno il compito di inviare i dati (identificativo SM, tensione, corrente, frequenza, energia attiva, energia reattiva, potenza attiva, potenza reattiva, power factor, THD di tensione, THD di corrente). I dati sono gestiti dal concentratore dati tramite un'applicazione creata in linguaggio di programmazione Python che, utilizzando un database MySQL, li archivia. La gestione del concentratore dati, avviene attraverso un'apposita interfaccia *web* realizzata in HTML, integrata con *script* in linguaggio PHP, mentre la comunicazione tra i dispositivi è realizzata tramite *Power Line Communication*.

5.1 Fase preliminare di progettazione e scelta dei parametri di comunicazione

Nella fase preliminare alla progettazione, è stato eseguito uno studio approfondito dei componenti scelti per la realizzazione del concentratore dati e dagli *Smart Meter*, ovvero il Raspberry Pi e i microcontrollori STM32F407.

5.1.1 Comunicazione

I dati da prelevare dagli *Smart Meter* possono essere trasmessi dal misuratore al concentratore di dati attraverso le linee elettriche, cioè con modem *Power Line Communication*, la motivazione di tale scelta risiede nel fatto che questa tecnologia è più sicura contro manipolazioni esterne. Infatti, poiché il PLC è un "tecnologia *wired*", per effettuare un attacco esterno un utente deve essere in grado di collegarsi fisicamente alla rete. Inoltre la rete proposta necessita di un basso consumo di energia e di un bit rate basso, essendo bassa anche la quantità di dati trasmessi in rete. Queste considerazioni ci hanno portato a scegliere la tecnologia PLC. Per interfacciare i componenti della rete con i modem PLC si è scelto il canale di comunicazione bidirezionale RS-232 per la trasmissione dei dati, ovvero il canale di comunicazione seriale che è presente in entrambi i componenti.

L'RS-232 è uno standard che definisce un'interfaccia di livello fisico del modello ISO/OSI per la realizzazione di una trasmissione seriale, sia in modalità sincrona che asincrona. L'interfaccia RS-232 ridotta (porta seriale a 9 pin) utilizza solo protocolli di trasmissione di tipo asincrono. Per trasmissione "seriale" si intende che i bit, che costituiscono l'informazione, sono trasmessi uno alla volta su di un solo "filo".

5.1.2 Sicurezza

La sensibilità del tipo di dati e l'affidabilità del controllo all'interno di una rete di Smart Meter richiede come requisito fondamentale la sicurezza informatica della rete di comunicazione. Per questo motivo si è scelto come algoritmo di crittografia l'Advanced Encryption Standard (AES-128 bit).

L'AES è una specifica per la crittografia dei dati elettronici stabiliti dal US National Institute of Standards and Technology (NIST) nel 2001.

AES è basato su un principio di progettazione conosciuto come una rete a sostituzione e permutazione ed è veloce sia software che hardware.

5.1.3 Formato dati

Il passo successivo è stato quello di analizzare la modalità di trasmissione dei dati scelta, cioè lo standard RS-232. Attraverso lo studio del microcontrollore STM32F407 e il suo linguaggio di programmazione, si è appreso che questa può essere effettuata utilizzando varie tipologie di dati, ovvero di tipo float, intero o con le stringhe.

Tra i vari tipi di trasmissione dati si è scelto di utilizzare quella con l'invio di float e interi, per garantire una buona gestione e manipolazione tramite il Python. Il formato scelto dei dati da trasmettere è rappresentato in tabella 2.

Parameter	Type	Unit	Description
ID	int (4 Byte)	-	Node identifier with 4 field
RMSv	float (4 Byte)	V	Voltage Root Mean Square
RMSi	float (4 Byte)	A	Current Root Mean Square
fs	float (4 Byte)	Hz	Voltage Frequency
eA	float (4 Byte)	Wh	Active Energy
eR	float (4 Byte)	VARh	Reactive Energy
P	float (4 Byte)	W	Active Power
Q	float (4 Byte)	VAR	Reactive Power
PF	float (4 Byte)	-	Power Factor
THDv	float (4 Byte)	%	Voltage Total Harmonic Distortion
THDi	float (4 Byte)	%	Current Total Harmonic Distortion

Tabella 2 Data format

Il formato di dati è composto da un campo "ID" che è l'identificatore del nodo nella rete locale e dai 10 parametri descritti nei prossimi paragrafi. Come illustrato nella tabella 2, l'identificatore è composto da 4 byte di tipo intero e gli altri parametri sono composti da 4 byte di tipo Float. La dimensione totale del pacchetto dati è di 44 byte. Poiché i dati sono cifrati con AES - 128, la dimensione finale del pacchetto è 64 byte

La scelta di utilizzare il Python, come linguaggio di programmazione per l'implementazione dell'applicazione del concentratore dati (Raspberry Pi), è stata presa per due motivi:

- Il primo è dovuto al fatto che esso è presente nel Raspberry Pi come linguaggio di sviluppo del pc all'interno del sistema operativo Raspbian ovvero una versione port del Debian.

- Il secondo, di maggiore importanza, è perché esso permette una connessione diretta al database MySQL, dovuta alla sua integrabilità con quest'ultimo.

La connessione a questo linguaggio di programmazione è risultata fondamentale anche per un altro aspetto, perché l'idea del progetto oltre che alla realizzazione di un'applicazione di monitoraggio, prevede anche il salvataggio dei dati e la possibilità di poterli consultare da remoto, tramite l'utilizzo di un comune browser.

5.2 Scelta dell'architettura della rete di Smart Meter

Ultimate queste scelte di progettazione si è passati alla configurazione dei vari componenti utilizzati, quindi l'installazione del sistema operativo all'interno del Raspberry Pi e la programmazione del microcontrollore STM32F407 per l'invio dei dati. In figura 12 è mostrato un esempio di architettura di rete di riferimento.

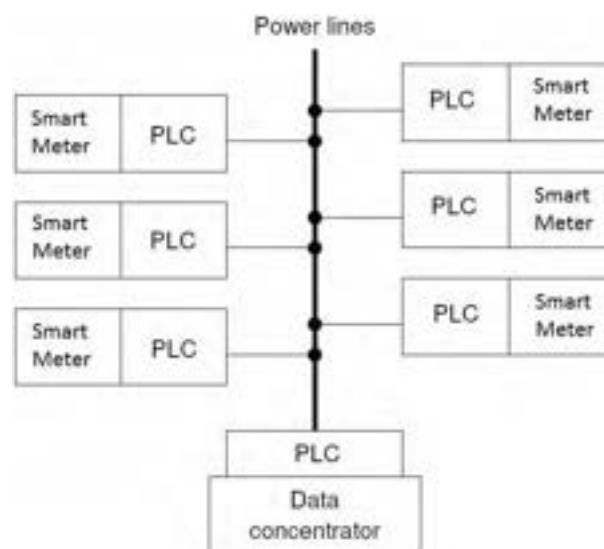


Figura 12 Architettura della rete

5.2.1 Installazione sistema operativo sul Raspberry Pi (concentratore dati)

Come primo passo, si è scelto di scaricare dal sito ufficiale la versione più aggiornata del sistema operativo Raspbian e tramite una memoria SD da 8GB, da inserire nell'apposita slot del pc, è stata installata sul Raspberry Pi. Successivamente si è passato all'installazione dei linguaggi di programmazione, come detto già precedentemente il Python in questa versione è già disponibile, mentre invece il MySQL e il PHP sono stati installati tramite gli opportuni comandi da terminale. Per poter poi usufruire della connessione diretta tra i linguaggi di programmazione Python e MySQL, si è proceduto all'installazione della libreria *pyhton-mysqldb* e analogamente, per poter sfruttare la connessione tra PHP e MySQL, è stato aggiunto il pacchetto *php5-mysql*. Per quanto riguarda la rappresentazione dei dati raccolti in forma grafica, è stata installata la libreria *pChart* a supporto del linguaggio PHP.

5.2.2 Programmazione del microcontrollore STM32F407 (Smart Meter)

All'interno del microcontrollore è stato implementato un Wattmetro utilizzando il linguaggio di programmazione C. Il wattmetro è composto da due stadi:

- stadio di trasduzione
- stadio di elaborazione

Lo stadio di trasduzione è composto a sua volta da:

- un trasduttore di tensione
- un trasduttore di corrente

Lo stadio di elaborazione è composto da:

- un microcontrollore ARM Cortex (STM32F407)
- un display per la visualizzazione dei risultati

Il microcontrollore ha il compito di acquisire i segnali di tensione e corrente, opportunamente condizionati dallo stadio di trasduzione, al fine di calcolare:

- valore efficace di tensione (RMSV)
- valore efficace di corrente (RMSI)
- frequenza della tensione (fs)
- energia attiva (eA)
- energia reattiva (eR)
- potenza attiva (P)
- potenza reattiva (Q)
- fattore di potenza (Pf)
- distorsione armonica totale di tensione (THDV)
- distorsione armonica totale di corrente (THDI)

Gli algoritmi implementati vengono elencati di seguito.

RMS della tensione (RMSV)

$$RMS_V = \sqrt{\frac{1}{N} \sum_{k=1}^N v_k^2}$$

RMS della corrente (RMSI)

$$RMS_I = \sqrt{\frac{1}{N} \sum_{k=1}^N i_k^2}$$

Potenza Attiva (P)

$$P = \frac{1}{N} \sum_{k=1}^N v_k \cdot i_k$$

Potenza reattiva QENEL

$$Q_{ENEL} = \frac{1}{N} \cdot \sum_{k=1}^N v_k \cdot i_{k+\frac{N}{4}}$$

Potenza reattiva QIEEE

$$Q = Q_{ieee} = \frac{1}{N} \left(\sum_{k=1}^N v_k \cdot \sum i_k \right)$$

Potenza reattiva Qfryze

$$S_r = RMS_V \cdot RMS_I$$

$$Q_{fryze} = \sqrt{S_r^2 - P_A^2}$$

Fattore di potenza P

$$PF = \frac{P}{A}$$

Potenza reattiva $Q_{budeanu}$

$$Q_{budeanu} = \frac{1}{2} \cdot \sum_{h=1}^M V_h \cdot I_h \cdot \sin \varphi_h$$

Distorsione armonica totale della tensione THD_v

$$THD_v = \sqrt{\frac{\sum_{h=2}^M V_h^2}{V_1^2}}$$

Distorsione armonica totale della corrente THD_i

$$THD_i = \sqrt{\frac{\sum_{h=2}^M I_h^2}{I_1^2}}$$

Individuazione delle armoniche di tensione e corrente attraverso il calcolo della FFT

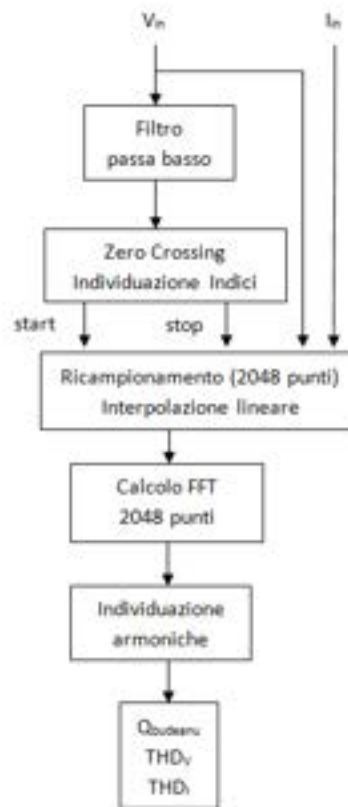


Figura 13 Schema per ricavare le armoniche di tensione e corrente

5.3 Codice dell'applicazione concentratore di dati

L'applicazione realizzata per consentire la visualizzazione dei dati inviati da un microcontrollore (*Smart Meter*) e allo stesso tempo effettuare la memorizzazione all'interno di un *database* MySQL dei dati (identificativo SM, tensione, corrente, frequenza, energia attiva, energia reattiva, potenza attiva, potenza reattiva, power factor, thdv, thdi), è stata scritta nel linguaggio di programmazione Python.

Di seguito verrà descritto il codice realizzato e le funzioni che svolgono alcune parti di esso. Inizialmente sono state importate tutte le librerie opportune come vediamo dallo stralcio di codice che segue:

```

import sys
import string
import struct
import serial
import MySQLdb as mdb
import time
from Crypto.Cipher import AES
  
```

Listing 5.1: Stralcio di codice per importare le librerie

Il motivo dell'importazione di queste particolari librerie è dovuto a:

- sys: per sfruttare alcuni parametri e funzioni specifiche per il sistema
- string: per sfruttare alcune funzioni sulle stringhe.

- struct: conversioni tra i valori Python e le strutture C
- serial: per permettere l'utilizzo della seriale sul Raspberry Pi.
- MySQLdb as mdb: per permettere la connessione tra Python e MySQL.
- time: per poter effettuare delle sleep.
- AES: per utilizzare algoritmi e protocolli crittografici AES-128

```
con = mdb.connect('localhost', 'root', 'alex', 'tesi')
floatArray=float()
ID1= int
ID2= int
ID3= int
ID4= int

key = "smartmetering"
decryptor = AES.new(key)

ser = serial.Serial("/dev/ttyAMA0",timeout=1)
```

Listing 5.2: Stralcio di codice per la connessione seriale

In questa parte di codice si è effettuata la connessione ad un *database* MySQL creato appositamente per immagazzinare i dati, l'accesso viene effettuato come prima istruzione in modo da permettere all'applicazione di interagire con il *database* in diverse fasi del programma. Inizialmente all'interno del codice sono state dichiarate le variabili da utilizzare e in particolare si è creata la variabile *ser* per effettuare la lettura dal canale seriale del Raspberry Pi. La variabile *key*, invece è la chiave utilizzata per la crittografia del tipo AES-128.

```
lista=["Smart Meter non letti"]

if id3==0:
    with con:
        cur = con.cursor()
        stringa = "select id3,id4 from smartmeter where status=true;"
        cur.execute(stringa)
        rows=cur.fetchall()
elif id4==0:
    with con:
        cur = con.cursor()
        stringa = "select id3,id4 from smartmeter where status=true and id3=%d;"%(id3)
        cur.execute(stringa)
        rows=cur.fetchall()
else:
    with con:
        cur = con.cursor()
        stringa = "select id3, id4 from smartmeter where status=true and id3=%d and id4=%d;"%(id3, id4)
        cur.execute(stringa)
        rows=cur.fetchall()
```

Listing 5.3: Stralcio di codice per il controllo SM non attivi

In questa parte di codice viene fatto un controllo all'interno del *database* per ricavare gli identificativi degli *Smart Meter* che risultano attivi nella rete, questa informazione viene passata alla variabile *row* e utilizzata per effettuare la lettura dei dati solo degli *Smart Meter* attivi.

```
indice=0;
for row in rows:
    start = time.time()# inizio calcolo del tempo di esecuzione

    ser.flushInput()
    write = ser.write(chr(row[0])) # messaggio di inizio lettura
    write = ser.write(chr(row[1]))

    buf=""
    bufenc=""
    i=0
    while i<64:
        read=""
        read = ser.read(1)
        if len(read)==0:
            i=80
            lista.append("0.0.%d.%d"%(row[0],row[1]))
            indice=indice+1
            bufenc = bufenc+read
            i = i+1

    ser.flushInput()
    if i==64:
        buf = decryptor.decrypt(bufenc)
```

Listing 5.4: Stralcio di codice per la lettura dei dati

A questo punto del codice i valori dallo *Smart Meter* sono stati letti, ma sono criptati per cui viene applicato il metodo *decrypt* ai dati e inseriti nella variabile *buf*, in attesa di essere passati ognuno nelle rispettive variabili, come mostrato nel codice sottostante.

```
i=0
j=0
while i<56:
    val=struct.unpack('f', buf[i:i+4])

    if j == 0:
        val=struct.unpack('i', buf[i:i+4])
        ID = val[0]
    if j == 1:
        RMSv = val[0]
    if j == 2:
        RMSi = val[0]
    if j == 3:
        fs = val[0]
    if j == 4:
        eA = val[0]
```

```

if j == 5:
    val=struct.unpack('i', buf[i:i+4])
    OWA = val[0]
if j == 6:
    eR = val[0]
if j == 7:
    val=struct.unpack('i', buf[i:i+4])
    OWR = val[0]
if j == 8:
    P = val[0]
if j == 9:
    Q = val[0]
if j == 10:
    PF = val[0]
if j == 11:
    THDv = val[0]
if j == 12:
    THDi = val[0]
if j == 13:
    val=struct.unpack('i', buf[i:i+4])
    code = val[0]
i=i+4
j=j+1

```

Listing 5.5: Stralcio di codice per il passaggio dei dati

Il programma si conclude con l'aggiornamento del *database* MySQL dei valori appena letti, tramite il metodo *execute* a cui è stata passata come parametro una stringa con all'interno il comando *insert* in linguaggio SQL.

```

with con:
    cur = con.cursor()
    stringa = "insert into data (id1, id2, id3, id4, RMSv, RMSi, fs, eA, OWA, eR, OWR, P, Q, PF, THDv, THDi, code) value (\'%d\', '%d\', '%d\', '%d\', '%f\', '%f\', '%f\', '%f\', '%d\', '%f\', '%d\', '%f\', '%f\', '%f\', '%f\', '%d\')" %(ID1, ID2, ID3, ID4, RMSv, RMSi, fs, eA, OWA, eR, OWR, P, Q, PF, THDv, THDi, code)
    cur.execute(stringa)

```

Listing 5.6: Stralcio di codice per l'inserimento dei dati nel database.

Dopo aver completato la scrittura del metodo *readSM* è stato creato il *main* come segue.

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

```

import cgi
import cgitb
cgitb.enable()
import readSM
import MySQLdb as mdb

print "Content-type: text/html\n\n"
con = mdb.connect('localhost', 'root', 'alex', 'tesi')

```

```

cur = con.cursor()
stringa = "select ID3, ID4 from request where CHIAVE=0;"
cur.execute(stringa)
row=cur.fetchone()
print("<h2>Data Concentrator</h2>")
print("<h3>Results:</h3>")

sm=readSM.readSM(row[0],row[1])
if len(sm) > 1:
    for s in sm:
        print(s+"<BR>")

```

Listing 5.7: Stralcio di codice del main.

All'interno del codice il programma effettua una connessione al *database* per interrogare la tabella *request*, nella quale è presente l'identificativo dello *Smart Meter* scelto dall'utente per la lettura dei dati, viene poi utilizzato il metodo *readSM*, per l'avvio della ricezione dei dati da parte dello *Smart Meter*.

5.4 Script per la visualizzazione in locale o in remoto

Un altro obiettivo di questo lavoro è stato quello di rendere accessibile il concentratore di dati da remoto. Per fare ciò è stata realizzata una interfaccia web in linguaggio HTML integrata con la scrittura di due script in PHP.

L'interfaccia web permette di interrogare il concentratore di dati in tre modalità:

- **Broadcast:** interroga tutti gli *Smart Meter* della rete
- **Single:** interroga uno o più *Smart Meter* selezionati
- **Storico:** fornisce i dati storici relativi ad un parametro scelto

Gli script in PHP sfruttano la connessione al *database* MySQL che è stato riempito con l'applicazione descritta precedentemente.

Le prime due modalità, Broadcast e Single fanno riferimento al medesimo script, il cui codice è quello che segue.

```
<?php
```

```

$host = "localhost";
$utente = "root";
$password = "alex";

```

```

mysql_connect("$host","$utente","$password") or die (mysql_error());
mysql_select_db ("tesi") or die (mysql_error());

```

```

foreach($_POST as $chiave=>$val){
echo ($chiave."=".$val);
}

```

```

$query = "UPDATE request set ID1='{$_POST['Itm_8_00_1']}', ID2='{$_POST['Itm_8_00_2']}',
ID3='{$_POST['Itm_8_00_3']}', ID4='{$_POST['Itm_8_00_4']}' WHERE CHIAVE=0";

```

```

mysql_query($query);
mysql_close();

```

```
header("location: /cgi-bin/main.py");
exit;
```

```
?>
```

Listing 5.8: codice dello script per la lettura in modalità Broadcast e Single

Lo script si occupa di connettersi al *database* ed aggiornare, attraverso una *query*, la tabella *request*, nella quale viene inserito l'identificativo dello *Smart Meter* da interrogare. Lo script interagisce con il file *main.py* il quale si occupa di richiamare il metodo *readSM* per leggere e salvare nel *database* i dati dagli *Smart Meter*

Il codice dello script per la modalità **storico**, invece prevede l'utilizzo della libreria *pChart* per la stampa dei dati in forma grafica e viene di seguito riportato.

```
<?php
// Standard inclusions
include("pChart.1.27d/pChart/pData.class");
include("pChart.1.27d/pChart/pChart.class");

$host = "localhost";
$utente = "root";
$password = "alex";

mysql_connect("$host","$utente","$password") or die (mysql_error());
mysql_select_db ("tesi") or die (mysql_error());

if ($_POST['tempo']=="ultimo giorno"){ $t=1;}
if ($_POST['tempo']=="ultima settimana"){ $t=7;}
if ($_POST['tempo']=="ultime 2 settimane"){ $t=14;}
if ($_POST['tempo']=="ultimo mese"){ $t=30;}
if ($_POST['tempo']=="ultimi 2 mesi"){ $t=60;}
if ($_POST['tempo']=="ultimi 6 mesi"){ $t=180;}
if ($_POST['tempo']=="ultimo anno"){ $t=365;}
$ID1=$_POST['itm_8_00_1'];
$ID2=$_POST['itm_8_00_2'];
$ID3=$_POST['itm_8_00_3'];
$ID4=$_POST['itm_8_00_4'];
echo ("ID1 = {$ID1}<BR>");
echo ("ID2 = {$ID2}<BR>");
echo ("ID3 = {$ID3}<BR>");
echo ("ID4 = {$ID4}<BR>");
echo ("Periodo = {$t} giorni<BR>");

$query = "select {$_POST['valore']} from data where id1={$ID1} and id2={$ID2} and id3={$ID3} and
id4={$ID4} and date_sub(curdate(), interval {$t} day)<=time";

$results=mysql_query($query);

$data=array();
```

```

$i=0;
while($row=mysql_fetch_array($results))
{
    $data[$i]=$row["{$_POST['valore']}"];
    //echo($data[$i]."<BR>");
    $i=$i+1;
}

// Dataset definition
$DataSet = new pData;
$DataSet->AddPoint($data,"Serie1");
$DataSet->AddAllSeries();
$DataSet->SetAbsciseLabelSerie();
$DataSet->SetSerieName("{"$_POST['valore']}","Serie1");

// Initialise the graph
$Test = new pChart(700,230);
$Test->setFixedScale(0,max($data)*1.5);
$Test->setFontProperties("pChart.1.27d/Fonts/tahoma.ttf",8);
$Test->setGraphArea(50,30,585,200);
$Test->drawFilledRoundedRectangle(7,7,693,223,5,240,240,240);
$Test->drawRoundedRectangle(5,5,695,225,5,230,230,230);
$Test->drawGraphArea(255,255,255,TRUE);
$Test->drawScale($DataSet->GetData(),$DataSet-
>GetDataDescription(),SCALE_NORMAL,150,150,150,TRUE,0,2);
$Test->drawGrid(4,TRUE,230,230,230,50);

// Draw the 0 line
$Test->setFontProperties("pChart.1.27d/Fonts/tahoma.ttf",6);
$Test->drawTreshold(0,143,55,72,TRUE,TRUE);

// Draw the cubic curve graph
$Test->drawCubicCurve($DataSet->GetData(),$DataSet->GetDataDescription());

// Finish the graph
$Test->setFontProperties("pChart.1.27d/Fonts/tahoma.ttf",8);
$Test->drawLegend(600,30,$DataSet->GetDataDescription(),255,255,255);
$Test->setFontProperties("pChart.1.27d/Fonts/tahoma.ttf",10);
$Test->drawTitle(50,22,"{"$_POST['valore']}","50,50,50,585);
$Test->Render("graph.png");

mysql_close();

echo("<img src='graph.png'/>");

?>

```

Listing 5.9: codice dello script per la modalità storico

Lo script si occupa di connettersi ed interrogare la tabella *data* del *database*, nella quale sono presenti tutti valori letti dalla rete di *Smart Meter*. L'interrogazione avviene tramite una *query*, nella quale è inserito l'identificativo dello *Smart Meter*, l'intervallo di tempo e la grandezza misurata da visualizzare. I dati ricevuti, grazie alla libreria *pChart*, vengono utilizzati per generare un grafico dell'andamento della grandezza misurata.

5.4.1 Interfaccia web per la visualizzazione in remoto

Il risultato finale per la visualizzazione in remoto del concentratore di dati è costituito da un interfaccia *web* realizzata in HTML mostrata in figura 14.



Figura 14 Home page dell'interfaccia web

Dalla pagina iniziale dell'interfaccia web (figura 13) è possibile interrogare la rete in tre diverse modalità:

1 – Broadcast: in questa sezione dell'interfaccia grafica è possibile avviare l'interrogazione di tutti gli *Smart Meter* della rete, l'utente semplicemente cliccando sul tasto "invia" avvierà gli applicativi descritti in precedenza che si occuperanno di effettuare tutte le procedure per poter visualizzare i risultati dell'interrogazione e nel contempo archivarli nel *database*.



Figura 15 Interfaccia web per l'interrogazione Broadcast

2 – Single: in questa sezione dell'interfaccia grafica è possibile avviare la lettura di uno o più *Smart Meter* della rete, la procedura è analoga alla precedente modalità *Broadcast* con la differenza che in questo caso l'utente potrà selezionare l'identificativo dello *Smart Meter* da interrogare.

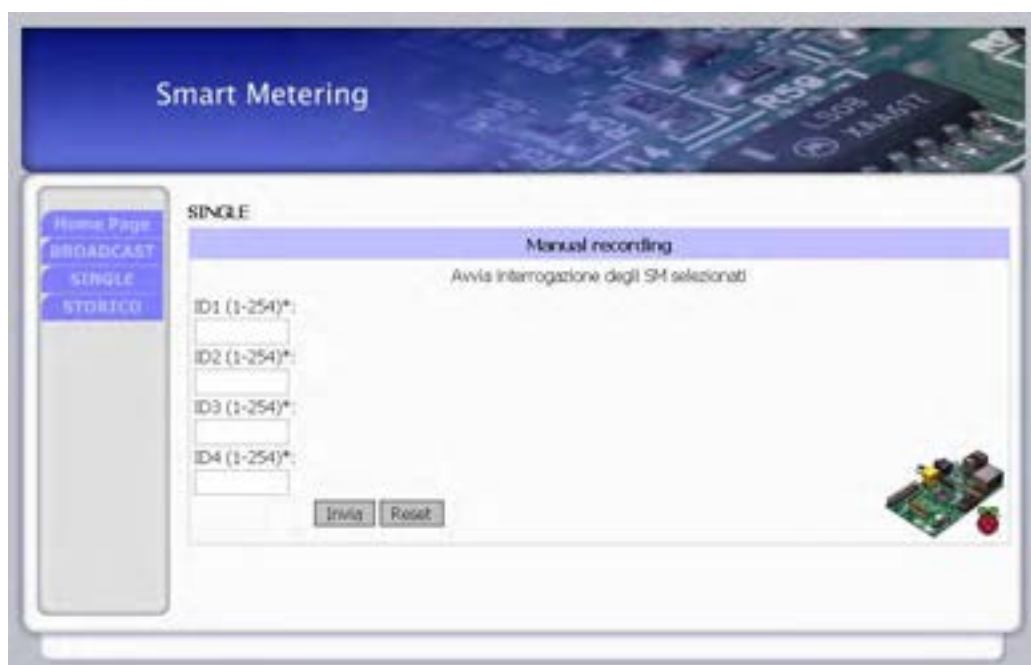


Figura 16 Interfaccia web per l'interrogazione Single

3 – Storico: a differenza delle precedenti modalità in questa sezione dell'interfaccia grafica è possibile interrogare il *database* per ricavare i valori storici di uno *Smart Meter*, l'utente selezionando alcuni parametri come: identificativo dello *Smart Meter*, intervallo di tempo e parametro da visualizzare, avvierà una *query* all'interno del *database*, tramite lo script descritto in precedenza, in modo da visualizzare solo i dati appartenenti allo *Smart Meter* selezionato nell'intervallo di tempo scelto.

Figura 17 Interfaccia web per l'interrogazione Storico

5.5 Creazione de database MySql

Il *database* per l'archiviazione dei dati in MySQL è stato creato prima di avviare l'applicazione in Python. Dalla figura 18 si vede che è stato creato un database con il nome "tesi" e all'interno di esso sono state inserite tre tabelle con il nome "data", "request" e "smartmeter". La tabella "data" è composta da tutti i campi utili per l'archiviazione delle grandezza fisiche lette (identificativo SM, tensione, corrente, frequenza, energia attiva, energia reattiva, potenza attiva, potenza reattiva, power factor, thdv, thdi). La tabella "smartmeter", contiene i campi riguardanti le informazioni relativi agli *Smart Meter* della rete (identificativo SM, ubicazione, stato). La tabella "request" viene utilizzata dallo script in PHP per il salvataggio delle informazioni sullo *Smart Meter* letto.

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 53
Server version: 5.5.35-0ubuntu0.12.04.2 (Ubuntu)

Copyright (c) 2000, 2013, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use test
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
mysql> show tables;
+-----+
| Tables_in_test |
+-----+
| data            |
| request         |
| smartmeter      |
+-----+
3 rows in set (0.00 sec)

mysql> 
```

Figura 18 Creazione del database

6 Implementazione e test sperimentali

In questo paragrafo verrà mostrata l'implementazione fisica del concentratore dati e i test effettuati sull'applicazione e lo *script* per la visualizzazione in remoto dei dati ricevuti dai *Smart Meter*.

6.1 Implementazione e configurazione della rete di Smart Meter

Allo scopo di effettuare test sperimentali sull'applicazione e lo *script* del concentratore di dati è stato realizzato lo schema a blocchi rappresentato in figura 19.

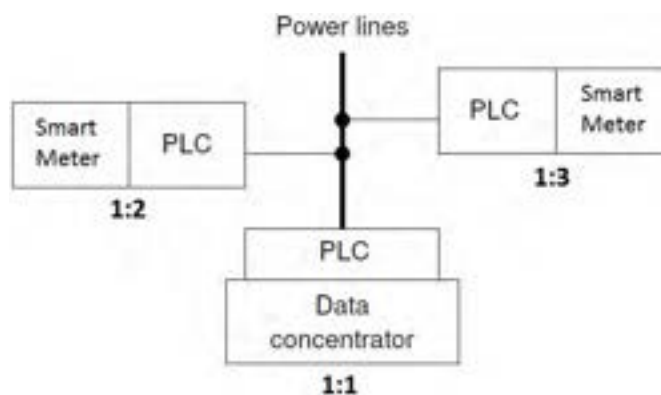


Figura 19 Architettura della rete

La rete di *Smart Meter* è composta da due *Smart Meter* per il monitoraggio di due carichi, realizzati con microcontrollore STM32F407, e da un concentratore di dati per la gestione e l'archiviazione de dati, realizzato con Raspberry Pi. Ogni componente è collegato alla rete elettrica tramite modem PLC per lo scambio di dati.

I modem PLC degli *Smart Meter* sono programmati con gli indirizzi 1:2 e 1:3, mentre il Modem PLC del concentratore Dati è programmato con indirizzo 1:1.

L'architettura di rete realizzata è mostrata in figura 20 a destra è presente la scheda Raspberry Pi, mentre a sinistra i due *Smart Meter*.



Figura 20 Architettura della rete di Smart Meter realizzata

6.2 Test della parte software

Una volta finita la configurazione della rete, si è passati alla fase di test, quindi dal terminale del concentratore (*Raspberry Pi*) è stata avviata l'interfaccia *web* attraverso il *browser* di navigazione. Si è poi scelto di effettuare un test per ogni modalità di interrogazione possibile (*Broadcast*, *Single* e *Storico*) di seguito descritta:

- **Broadcast:** in questo test è stata avviata la lettura di tutti gli *Smart Meter* della rete, in figura 21 è mostrato il risultato del test, dove sono visibili i dati relativi ai due *Smart Meter* collegati alla rete, inoltre viene visualizzato la data della lettura effettuata.

Smart Metering

result broadcast

RISULTATI

ID1	ID2	ID3	ID4	RMTD-2	RMTD-3	Rn-2	Rn-3	Rn-4	Rn-5	P-2	P-3	P-4	P-5	Time
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:43:47
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:47:54
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:48:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:49:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:50:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:51:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:52:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:53:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:54:58
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:55:58

Figura 21 Test Broadcast - i dati ricevuti.

– **Single:** questo test differisce dal precedente per la possibilità di poter scegliere uno o un gruppo di *Smart Meter* della rete da interrogare, inserendo il valore dell'identificativo. In figura 22 è mostrato il risultato del test.

Smart Metering

result single

RISULTATI

ID1	ID2	ID3	ID4	RMTD-2	RMTD-3	Rn-2	Rn-3	Rn-4	Rn-5	P-2	P-3	P-4	P-5	Time
01	02	03	04	05	06	07	08	09	10	11	12	13	14	2004-08-13 10:47:47

Figura 22 Test Single - i dati ricevuti.

– **Storico:** questo test permette di interrogare il *database* per ricavare i valori storici di uno *Smart Meter* selezionando alcuni parametri quali: identificativo dello Smart Meter, intervallo di tempo, parametro da visualizzare. In figura 23 è mostrato il risultato del test dove sono visibili i dati relativi allo *Smart Meter* interrogato, inoltre viene visualizzato un grafico dell'andamento della grandezza selezionata.



Figura 23 Test Storico - i dati ricevuti. Risultati sperimentali sul modem PLC

La sezione più importante e interessante dell'intero lavoro è sicuramente quella che ora andremo ad affrontare. Essa sarà rivolta allo studio sperimentale del modem powerline ST7537 della SGS-THOMSON MICROELECTRONICS (le cui caratteristiche saranno descritte nel prossimo paragrafo).

In particolare verranno affrontati i problemi relativi a:

- calcolo del bit error rate in funzione di vari parametri, quali velocità di trasmissione e lunghezza dei frame trasmessi; a tale scopo analizzeremo a fondo gli aspetti della programmazione in LabVIEW che hanno permesso di ottenere tali risultati.
- caratterizzazione del canale in termini di risposta in frequenza e di livelli di attenuazione esistenti; a riguardo daremo anche risalto alla strumentazione adoperata.

6.3 Test della tecnologia di comunicazione

Il modem powerline ST7537 della SGS-THOMSON MICROELECTRONICS è fabbricato con tecnologia CMOS ed è in grado di trasmettere e ricevere dati a 2400 bps in modalità half duplex usando una frequenza portante di 132,45 kHz (in accordo con lo standard CENELEC EN50065, il quale specifica l'uso di frequenze portanti per home automation nel range compreso tra i 125 kHz e i 140 kHz, e la regolamentazione US FCC, la quale specifica l'uso di frequenze portanti ad un valore inferiore ai 450 kHz).

La modulazione adoperata è quella FSK (Frequency Shift Keying), di cui abbiamo brevemente discusso in precedenza. La scelta della FSK è dettata dal fatto che risulta essere uno dei più convenienti tipi di modulazione per trasmissioni su cammini altamente rumorosi e ad alta densità di "traffico", in accordo con la necessità di avere circuiti poco costosi e complessi. Cerchiamo ora di capire come funziona il dispositivo.

All'interno del modulatore FSK, il segnale da trasmettere viene individuato da due sole frequenze: al bit 0 viene associata una frequenza pari a 133,05 kHz mentre al bit 1 viene associata una frequenza pari a 131,85 kHz, ottenute tramite un oscillatore a 11,0592 MHz.

Il segnale modulato viene poi filtrato da un filtro passa banda per limitare lo spettro di uscita e ridurre il livello delle componenti armoniche. Lo stadio finale del blocco di trasmissione è costituito da un amplificatore operazionale il quale necessita di un segnale feedback dall'amplificatore di potenza.

Nella sezione di ricezione il segnale è applicato al RAI Input (con una sensibilità di 1 mV_{RMS}) dove viene filtrato da un filtro passa banda caratterizzato da una banda passante di 12 kHz, centrata alla frequenza della portante. L'uscita di questo filtro è amplificata di 20 dB, dopodiché il segnale viene convertito da un mixer connesso ad un oscillatore locale sintetizzato dal blocco modulatore FSK.

Infine, un filtro passa banda a frequenza intermedia, la cui frequenza centrale è 5,4 kHz, ha il compito di migliorare il rapporto segnale rumore all'ingresso del demodulatore FSK. L'accoppiamento tra l'uscita del filtro e l'ingresso del demodulatore è realizzato tramite un condensatore esterno che elimina problemi di offset.

Nell 'ST7537 ci sono 2 importanti funzioni aggiuntive: il watchdog (WD) ed il carrier detect (CD), di cui ora ne chiariremo la funzionalità.

La funzione WD serve ad assicurare che il microcontrollore del modem stia funzionando correttamente. Il software del microcontrollore deve includere istruzioni in grado di inviare un impulso al WD input almeno una volta ogni 1,5 s. Se non avviene nessuna transizione a questo ingresso per 1,5 s viene generato un segnale di reset per reinizializzare il microcontrollore. Quindi la funzione watchdog assicura che malfunzionamenti causati da fenomeni glitched siano corretti velocemente.

Il CD, invece, viene utilizzato in quanto in pratica, in quasi tutte le applicazioni, più di due dispositivi saranno connessi alla powerline. Prima di trasmettere, un dispositivo deve perciò verificare a priori che non ci sia già una portante presente sulla linea perché altrimenti, se così fosse, dovrebbe aspettare e riprovare a trasmettere in un istante successivo.

L'uscita del carrier detect negato ($\overline{CD}$) è posta bassa quando l'ampiezza del segnale in ingresso al RAI (Receive Analog Input) è maggiore di V_{CD} (che è tipicamente pari a 5 mV_{RMS}) per un tempo pari almeno a T_{CD} (che è tipicamente pari a 4 ms). Quando il segnale d'ingresso scompare o diventa più piccolo di V_{CD} , $\overline{CD}$ è mantenuto basso per un tempo pari almeno a T_{CD} , prima di ritornare al livello alto. V_{CD} input è la tensione di soglia del carrier detect, la quale è settata internamente. Il grafico di figura 24 rappresenta la minima ampiezza del segnale ricevuto (V_{in}) il quale può essere rivelato (per $\overline{CD} = 0$) in funzione della frequenza.

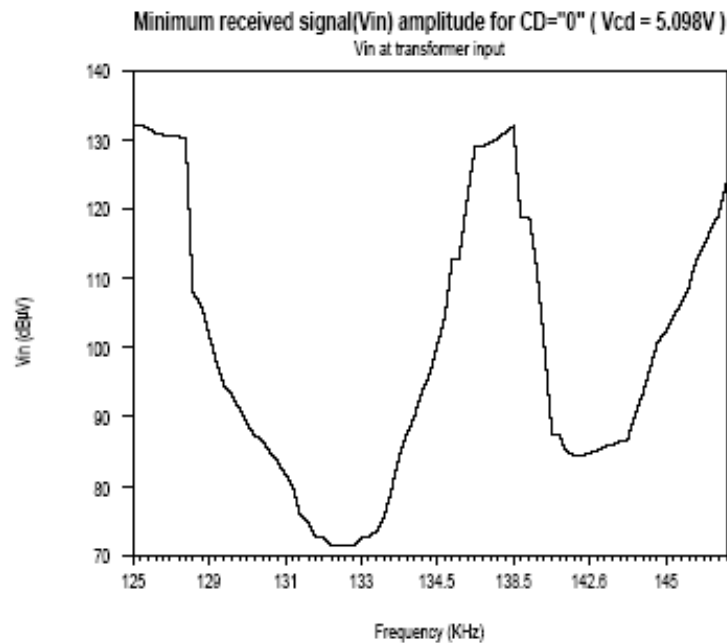


Figura 24 RAI Input minimum detection level

In questo modo il segnale d'ingresso (V_{in}), alla frequenza di 133,05 kHz (livello logico alto) e di 131,85 kHz (livello logico basso) può essere rivelato ad un livello molto basso. Per frequenze più piccole di 129 kHz o più grandi di 150 kHz la rivelazione è fatta a un livello molto alto del segnale d'ingresso V_{in} . Perciò solo frequenze significative del segnale d'ingresso possono essere rivelate.

Andiamo ora ad analizzare gli aspetti relativi al miglioramento di sensibilità del dispositivo.

In tutti i modem, il rivelatore di portante stoppa (cioè non fa passare) i dati digitali uscenti (presenti sul pin RXD, Receive Data Output) quando il segnale analogico ricevuto (RAI) è al di sotto di un livello di segnale definito (il livello di carrier detect, V_{CD} , che è tipicamente pari a $5 mV_{RMS}$). Questo significa che stiamo trasmettendo dati quando il segnale è minore del livello di carrier detect).



Figura 25 Andamento di RXD in funzione di RAI

Nell 'ST7537 il clamping di $\overline{CD}$ nei confronti di RXD è programmabile grazie al pin TXD.

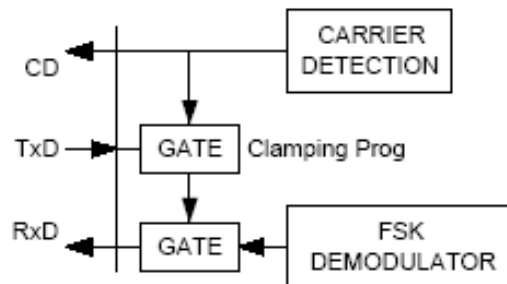


Figura 26 Modalità di comunicazione

In questo modo noi siamo in grado di ricevere dati anche se il segnale analogico ricevuto (RAI) è minore di 5 mV . Quando rimuoviamo il clamping su RxD da parte di $\overline{CD}$, siamo in grado di acquisire dati senza errore con un livello di ricezione di $400\text{ }\mu\text{V}$.

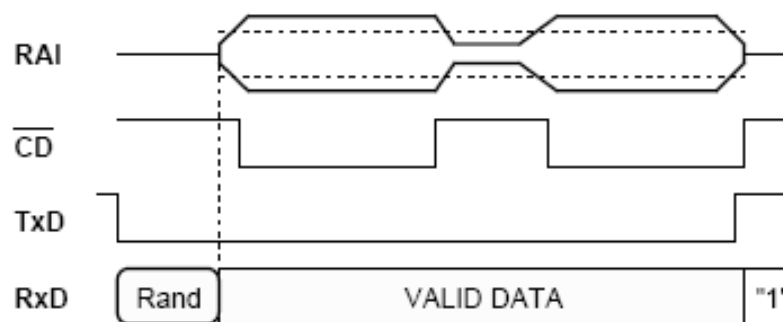


Figura 27 Andamento di RxD in funzione di RAI e TxD

Come si nota dalla figura 27, ogni qualvolta RAI è minore del livello di carrier detect (V_{CD}) avviene l'acquisizione dei dati (su RxD) se $\text{TxD} = 0$. Quando $\text{TxD} = 0$ e il segnale ricevuto (RAI) non è uno di quelli previsti per il modem ST7537 (ad esempio a causa del rumore) l'RxD è random (in molte configurazioni è a 0).

Abbiamo visto che programmando il $\text{TxD} = 0$ nella modalità ricezione, abbiamo un aumento della sensibilità del modem perché non ci sono più clamping da parte di $\overline{CD}$. Il dispositivo è in grado di ottenere una buona comunicazione con un segnale ricevuto pari a circa $50\text{ dB}\mu\text{V}$, il che significa una dinamica di circa 70 dB . Siccome vogliamo beneficiare di una buona sensibilità per il nostro modem, poniamo TxD a 0 in modalità ricezione e creiamo con semplicità un frame rivelatore. Useremo il segnale $\overline{CD}$ come menzionato dal CENELEC solo quando vogliamo trasmettere il frame.

Differenti software di rivelazione dei frame possono essere implementati in funzione del particolare tipo di microcontrollore. Si può programmare il microcontrollore ad andare ad ispezionare nel frame ricevuto se esso riceve il byte previsto.

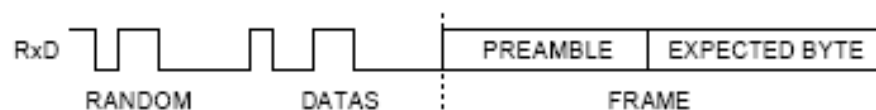


Figura 28 Frame rivelatore

In questo modo il preambolo serve per “addestrare” il demodulatore (quando inizia una comunicazione i primi 3 byte sono riservati al ricevitore) e quando si avrà il byte previsto il microcontrollore andrà nella routine del frame ricevuto.

Le prossime figure rappresentano lo schema elettrico del modem powerline ST7537 e la descrizione dettagliata della piedinatura del chip ST7537CFN. All'interno del report è comunque presente il data sheet completo del modem, dove alcuni aspetti vengono esaminati con maggiore approfondimento.

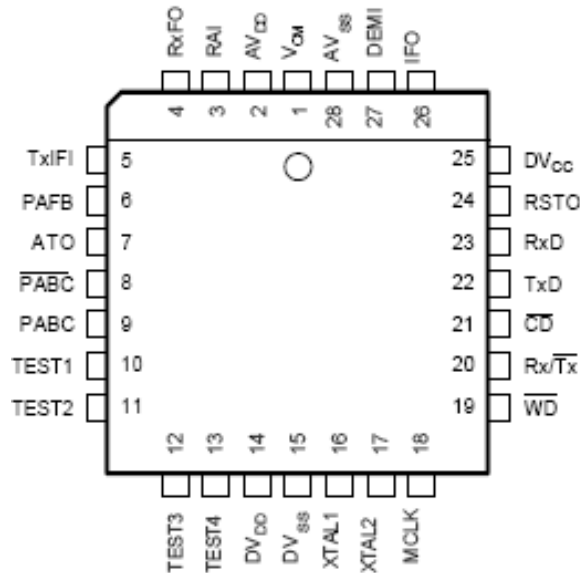


Figura 29 Il chip ST7537CFN

PIN DESCRIPTION

Pin Name	Pin Number	Pin Type	Description
V _{CM}	1	Analog	Common Mode Voltage
AV _{DD}	2	Supply	Analog Power Supply : 10V ±5 %
RAI	3	Analog	Receive Analog Input
RxF _O	4	Analog	Receive Filter Output
TxIF _I	5	Analog	Transmit and Intermediate Frequency Filters Test Input (mode TEST3)
PAFB	6	Analog	Power Amplifier Feed-back Input
ATO	7	Analog	Analog Transmit Output
PABC	8	Digital (10V)	Power Amplifier Bias Current Control Complementary Output
PABC	9	Digital (10V)	Power Amplifier Bias Current Control Output
TEST1	10	Digital	Tx to Rx Automatic Mode Switching Control Input
TEST2	11	Digital	Automatic Mode Switching Time and Watch-dog Time Reduction Control Input
TEST3	12	Digital	TxIF _I Selection Input
TEST4	13	Digital	Undelayed Reset Input
DV _{DD}	14	Supply	Digital Power Supply : 10V ±5%
DV _{SS}	15	Supply	Digital Ground : 0V
XTAL1	16	Digital (10V)	Crystal Oscillator Input
XTAL2	17	Digital (10V)	Crystal Oscillator Output
MCLK	18	Digital	Master Clock Output
WD	19	Digital	Watch-dog Input
Rx/Tx	20	Digital	Rx or Tx Mode Selection Input
CD	21	Digital	Carrier Detect Output
TxD	22	Digital	Transmit Data Input
RxD	23	Digital	Receive Data Output
RSTO	24	Digital	Reset Output
DV _{CC}	25	Supply	Digital Buffers Supply Voltage : 5V ±5 %
IFO	26	Analog	Intermediate Frequency Filter Output
DEMI	27	Analog	Demodulator Input
AV _{SS}	28	Supply	Analog Ground : 0V

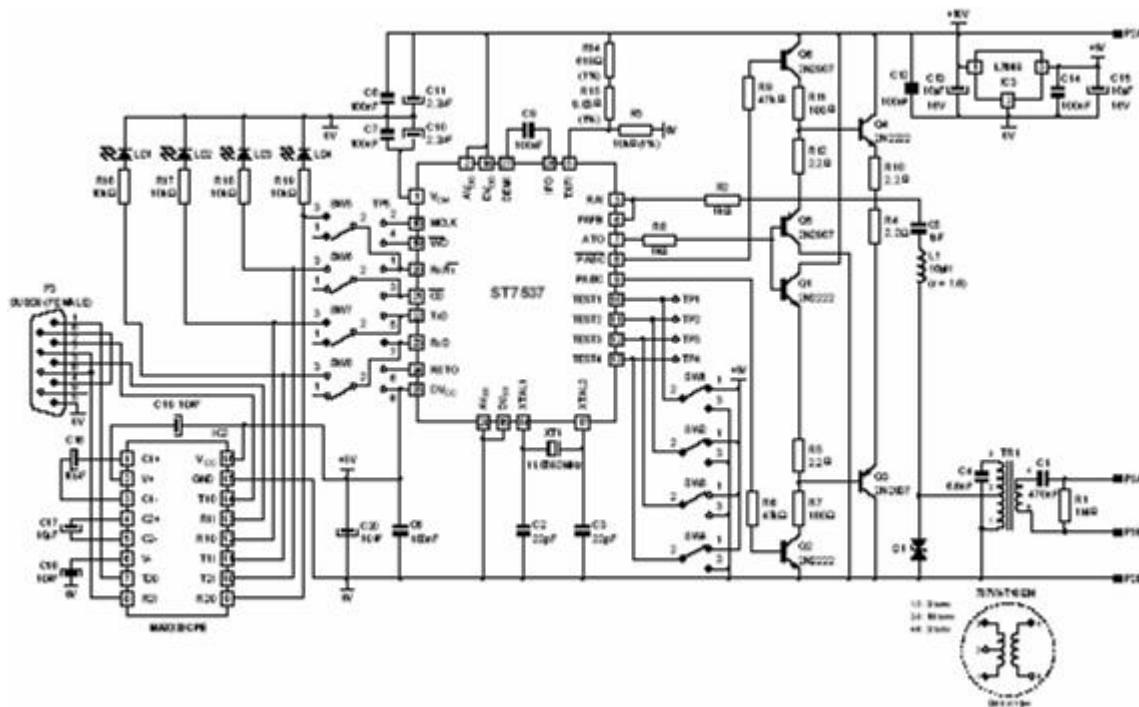


Figura 30 Lo schema elettrico del modem powerline ST7537



Figura 31 Modem powerline ST7537

6.4 L'ambiente Labview

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) è un ambiente di sviluppo di programmi applicativi particolarmente indirizzato alla creazione e gestione di strumentazione elettronica. Elaborato dalla National Instrument, si differenzia da altri ambienti di sviluppo come Basic, C o LabWindows/CVI per un aspetto importante: mentre gli altri sistemi di programmazione utilizzano un linguaggio testuale e si usano creando linee di codice, LabView utilizza un linguaggio di programmazione

grafica, il G, e si usa creando programmi sottoforma di simboli grafici. Le differenti funzioni del linguaggio sono rappresentate con delle icone e la programmazione si svolge cablando tali icone con opportuni collegamenti. LabVIEW può essere utilizzato anche senza una preparazione specifica informatica: l'utilizzare simboli grafici piuttosto che linguaggio testuale nel descrivere le operazioni di programmazione (ovvero il fatto, in altre parole, che il programmatore non veda sullo schermo una sequenza di righe ma un circuito cablato con simboli e icone a lui familiari) consente un'immediata percezione del tipo di funzionalità che l'applicativo offre.

LabVIEW è un ambiente di programmazione potente che permette di sviluppare operazioni complesse come acquisizioni di dati, gestione della memoria e dei tempi, gestione di strumenti esterni, elaborazioni complesse di dati, ecc.. Anche se è un ambiente di programmazione ad uso generale e possiede delle librerie di funzioni estese, adatte a tutti i campi di programmazione, LabVIEW eccelle nelle librerie di funzioni sviluppate specificatamente per l'acquisizione di dati, il controllo di strumenti di misura, l'analisi, la presentazione e la memorizzazione di dati.

Può essere utilizzato per gestire le porte RS-232, RS-485, GPIB (IEEE 488), VXI e PXI, oltre che per gestire specifiche schede di acquisizione dati per PC. Inoltre integra delle librerie studiate per adoperare logiche di gestione dati standard, come TCP/IP ed ActiveX.

Un'importante e comoda caratteristica di tale linguaggio è costituita dal fatto che esso si occupa di compilare l'applicativo in tempo reale durante l'editing, cioè durante la costruzione del diagramma. In questo modo, ad ogni modifica del diagramma, può avvertire il programmatore di eventuali errori che sta facendo. Ovviamente gli errori che il compilatore segnala sono solo quelli funzionali (per esempio quello in C potrebbe essere un'operazione per la quale non è stata definita la variabile oppure un ciclo while senza la condizione).

Un'altra importante caratteristica della programmazione è la possibilità di creare l'interfaccia con l'utente, chiamata "Front Panel", in maniera veloce e separatamente dalla programmazione della funzionalità dell'applicativo, che si svolge in un'altra schermata chiamata "Block Diagram". Non per nulla le applicazioni sviluppate in questo ambiente sono chiamate VI (Virtual Instrument), in ragione del loro aspetto durante il funzionamento che può imitare quello di alcuni strumenti reali.

Inoltre, strumenti come la definizione nel programma di punti d'arresto temporaneo oppure di esecuzione passo passo aiutano e velocizzano lo sviluppo e la realizzazione del programma stesso.

Un'altra particolarità, nell'ambito della programmazione, è rappresentata dalla semplicità con la quale si possono creare diagrammi nei quali diverse funzioni siano svolte simultaneamente, motivo per cui si può definire il LabVIEW come un ambiente multitasking. Questa caratteristica è una diretta conseguenza della programmazione mediante connessione di blocchi funzionali: ovvero definizione di "nodi", collegati tra loro da "fili" che definiscono il percorso del flusso dei dati. Si deve comunque tenere presente che l'esecuzione di un determinato nodo ha luogo solamente quando tutti i relativi ingressi sono disponibili, in altre parole quando tutti i nodi precedenti su quel cammino del flusso dei dati hanno reso disponibili le loro uscite.

LabVIEW, è distribuito in diverse versioni per poter essere utilizzato su piattaforme Windows, Machintosh, Sun, Unix, ecc.. In conclusione, il LabVIEW è stato studiato per offrire in campo scientifico un linguaggio di programmazione potente ma rivolto particolarmente a persone con competenze specifiche nel campo dell'elettronica e della strumentistica. A questo punto ci dedicheremo ad una breve analisi del programma in LabVIEW che è stato adoperato per il calcolo del bit error rate (BER) del modem powerline ST7537. Infatti, il problema iniziale che si è dovuto affrontare era rappresentato dalla necessità di far colloquiare i due dispositivi. Il punto cardine del lavoro è stato quello di far ricevere correttamente al modem in ricezione il preambolo di cui esso necessitava per potersi "svegliare" e interpretare i dati che gli arrivavano in maniera corretta. Tale preambolo era costituito da 8 byte rappresentati, nell'ordine, da quattro caratteri y, due caratteri a, dalla dimensione (size) del messaggio e dalla modalità di trasmissione (nel nostro caso abbiamo utilizzato la modalità "testo"). Un altro ostacolo, molto ostico da superare, è stato quello di non avere problemi di sincronizzazione tra trasmettitore e ricevitore. E 'stato, infatti, necessario introdurre opportuni ritardi (sia nel blocco di trasmissione che in

quello di ricezione) per evitare che il modem ricevente potesse inavvertitamente perdere la sincronizzazione (naturalmente quando ciò accadeva il BER raggiungeva valori del tutto inaccettabili). Per realizzare il programma è stato necessario creare dei driver (ben visibili nella figura precedentemente riportata) che svolgessero delle determinate funzioni all'interno dello stesso.

Nel dettaglio, ci siamo serviti di: un sottoprogramma che configurasse il tipo di trasmissione (in particolare, era importante stabilire quale doveva essere il baud rate e su quali porte seriali del PC doveva avvenire la comunicazione); di due driver che si occupassero, rispettivamente, della trasmissione e della ricezione di un generico messaggio lungo la powerline (questi due blocchi erano sicuramente i più importanti tra quelli adoperati perché erano proprio quelli predisposti a risolvere i problemi di sincronizzazione e di invio del preambolo appena descritti); ed infine, di un driver in grado di chiudere la comunicazione.

Risolto questo problema, è stato poi semplice aggiungere al programma quelle funzionalità che gli permettessero di calcolare il BER in funzione di parametri quali baud rate (fino a 2400 bps), lunghezza dei frame e distanza di collegamento tra i due modem.

6.5 Calcolo Del Bit Error Rate

Per prima cosa, analizziamo il BER calcolato con misure effettuate proprio sul modem di nostro interesse. A riguardo è utile osservare, nella figura seguente, la disposizione dei dispositivi adoperati per questo test.

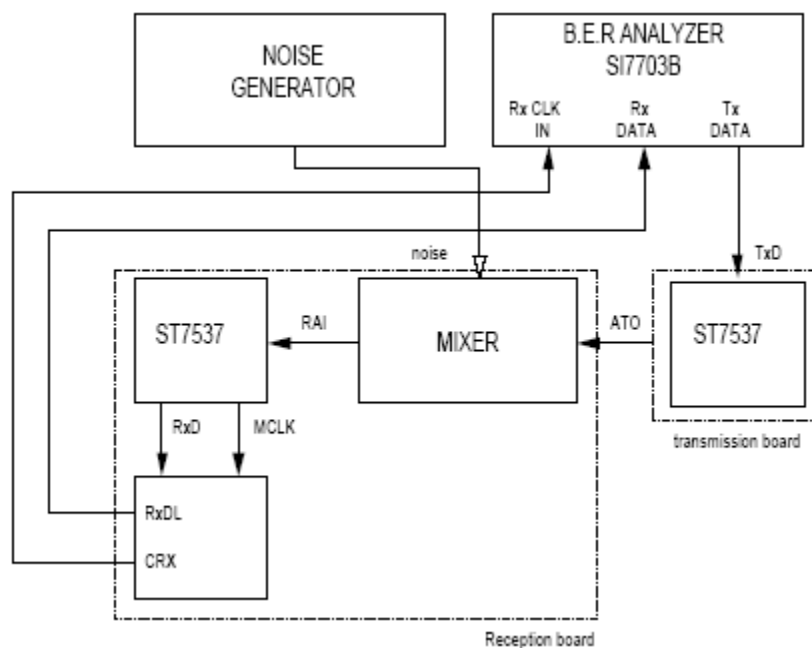


Figura 32 BER test

Sono necessari due modem: uno per la trasmissione, l'altro per la ricezione (va specificato che si è scelto di utilizzare un baud rate pari a 1200). Tramite un mixer, viene aggiunto rumore bianco all'uscita (ATO) dell'ST7537 trasmettente. Lo scopo è quello di evidenziare l'andamento del BER al variare del rapporto segnale/rumore. L'uscita del mixer è posta all'ingresso (RAI) del modem ricevente. Tramite un "BER Analyzer" (con clock asincrono interno pari a 1,2 kHz) viene generato un segnale digitale pseudo-random che viene posto all'ingresso (TXD) dell'ST7537 trasmettente. In uscita (CRX) al blocco di ricezione viene generato un segnale di clock a 1,2 kHz tramite il segnale MCLK proveniente dall'ST7537 ricevente. Il segnale digitale (RXD) all'uscita di quest'ultimo viene quindi amplificato (RXDL) e sincronizzato con il

segnale di clock CRX. Questi due segnali vengono poi inviati al BER Analyzer. Le misure sono state ottenute adoperando due diversi valori del RAI (ovvero 10,023 e 1,14 mV_{RMS} : vedi figure 33 e 34).

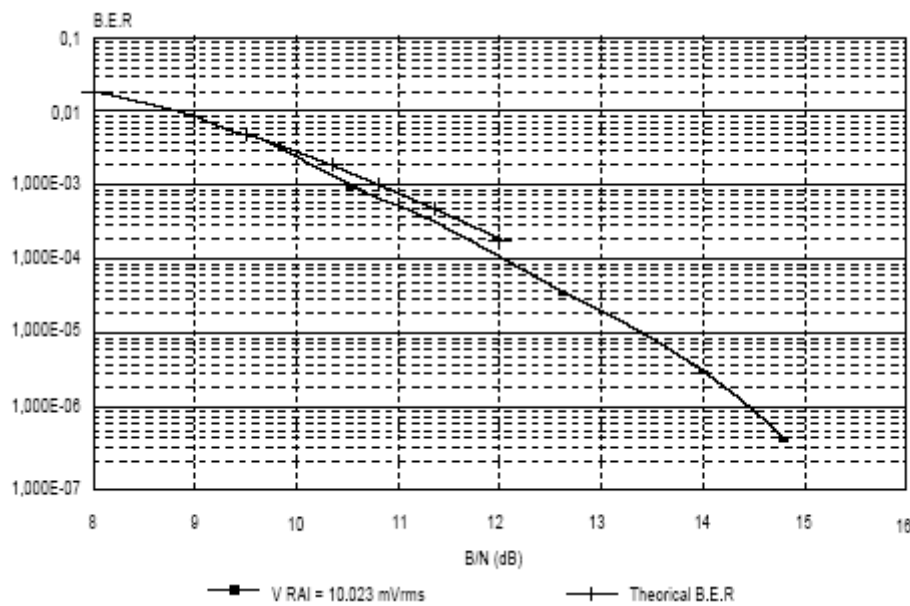


Figura 33 BER per RAI pari a 10,023 mV_{RMS}

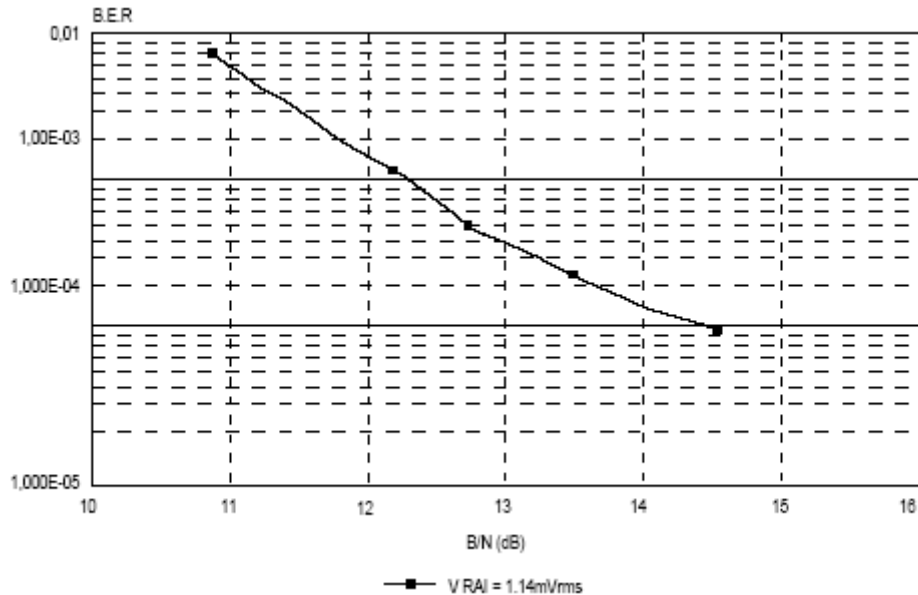


Figura 34 BER per RAI pari a 1,14 mV_{RMS}

Dalle valori riportati, si evince che con un RAI di 10 mV_{RMS} ed un S/N pari a 15 dB il BER vale 10^{-7} ; mentre con lo stesso S/N ed un RAI di 1,14 mV_{RMS} il BER si aggira intorno a 10^{-4} . Ad ogni modo, l'ST7537 è in grado di comunicare con bassi livelli del segnale d'ingresso (RAI), ovvero nell'intorno 1

mV_{RMS} . Nella figura 33, il BER misurato viene confrontato con il BER teorico di un tipico modem a modulazione BFSK.

Andiamo ora ad analizzare in dettaglio le misure effettuate.

Innanzitutto, i bit rate utilizzati sono stati i seguenti: 600, 1200 e 2400 bps. La lunghezza dei frame trasmessi è stata anch'essa variabile: in pratica venivano inviati, per ogni misura, 10^7 bit, sottoforma di pacchetti di 22, 52, 112 e 125 byte ciascuno. Questi valori sono stati scelti a piacere, ma comunque la scelta è stata vincolata dal fatto che sulla porta seriale (utilizzata per connettere il modem al calcolatore) non potevano viaggiare più di 127 byte.

Possiamo riassumere in una tabella i valori di BER ottenuti:

byte/bps	2400	1200	600
22	2,49E-03	1,00E-05	1,00E-07
52	6,14E-03	4,16E-05	4,20E-06
112	1,29E-02	1,91E-05	
125	1,37E-02		

Da cui si ricavano i seguenti grafici:

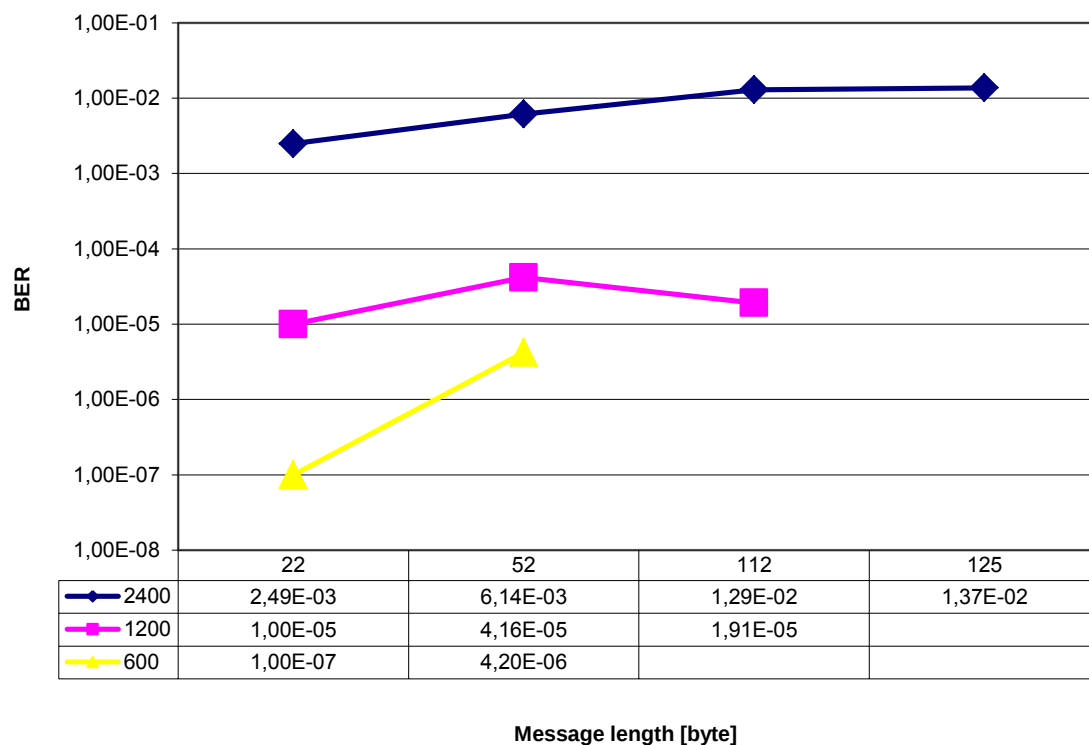


Figura 35 BER vs. message lenght [byte]

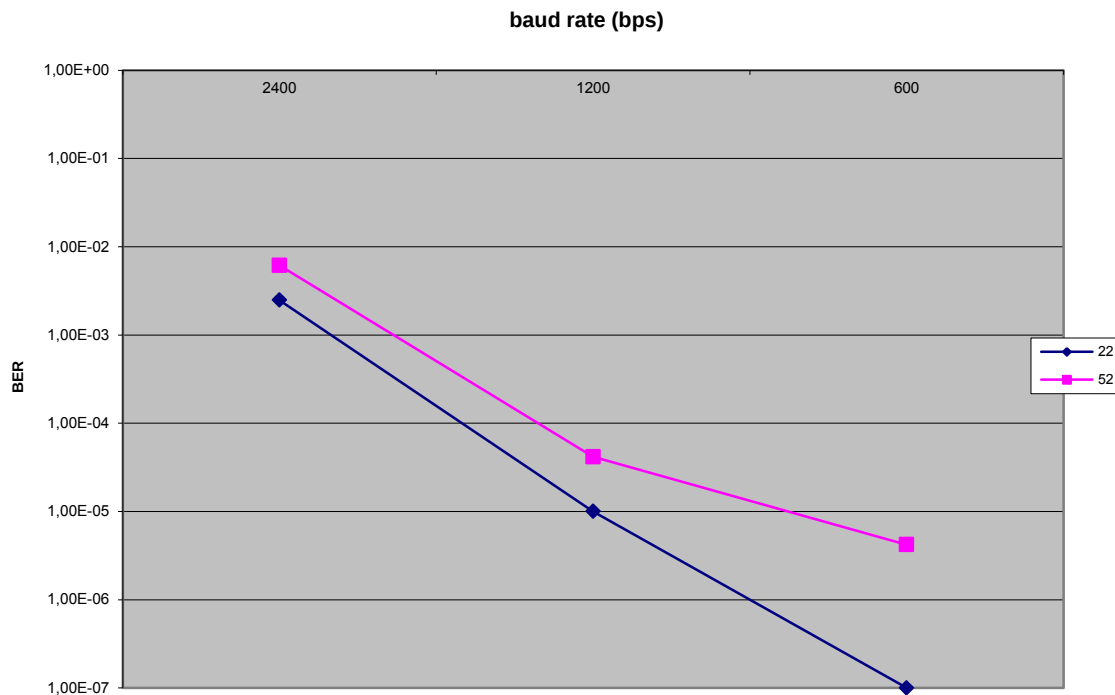


Figura 36 BER vs. baud rate [bps]

La figura 35 mostra l'andamento del BER in funzione della lunghezza dei frame per diversi valori del bit rate, mentre nel grafico di figura 36 il BER varia al variare della velocità di trasmissione per differenti lunghezze dei pacchetti.

Gli andamenti ottenuti sperimentalmente si avvicinano molto a quelli previsti, visto che il BER aumenta all'aumentare del baud rate e all'aumentare delle dimensioni del frame trasmesso. L'unico valore "anomalo" registrato riguarda il calcolo del BER per un bit rate pari a 1200 bps: infatti, in questo caso osserviamo che ad un certo punto i bit non esattamente riconosciuti in ricezione diminuiscono invece di aumentare.

Nel complesso, comunque, il bit error rate ricavato risulta soddisfare i requisiti di una buona comunicazione in quanto si attesta su valori abbastanza bassi; addirittura, a 600 bps, il BER vale soltanto $4,2 \cdot 10^{-6}$: ciò implica una bassa percentuale di bit non esattamente interpretati da parte dal modem. Osserviamo adesso i risultati del calcolo di BER ottenuti con un sistema di modulazione OFDM opportunamente simulato. Sono stati utilizzati N sottocanali ognuno con larghezza di banda $1/T$ e separati tra di loro di $1/T$, dove T è l'intervallo di simbolo di ogni singolo sottocanale. E' stata assunta assenza di ISI senza un dato sottocanale, mentre i sottocanali sono stati scelti con una banda stretta abbastanza per potere ritenere come unico effetto del multipath il "flat fading" in ogni sottocanale. In generale, anche se il canale multipath ha caratteristiche di tipo flat lungo un dato sottocanale, all'interno della banda del segnale multiportante il canale è selettivo in frequenza. Nelle figure 37-39 vengono mostrati gli andamenti dello spettro di potenza, del segnale con rumore e della costellazione 64-QAM della simulazione di trasmissione OFDM.

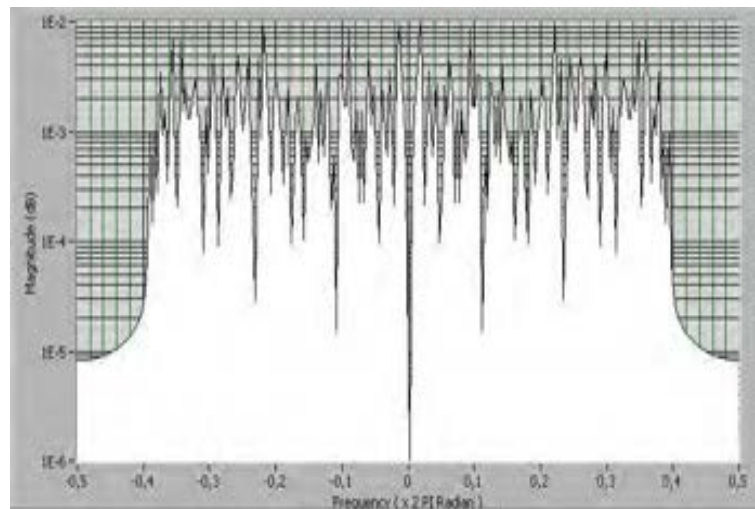


Figura 37 Spettro di potenza del segnale OFDM/64-QAM

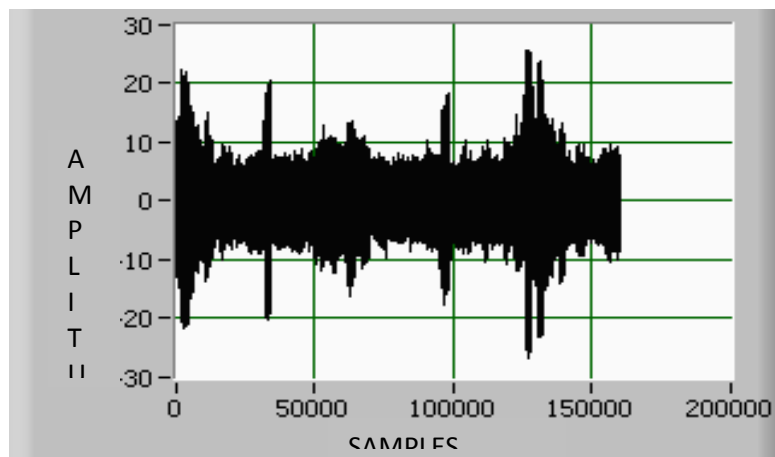


Figura 38 Segnale OFDM/64-QAM ricevuto

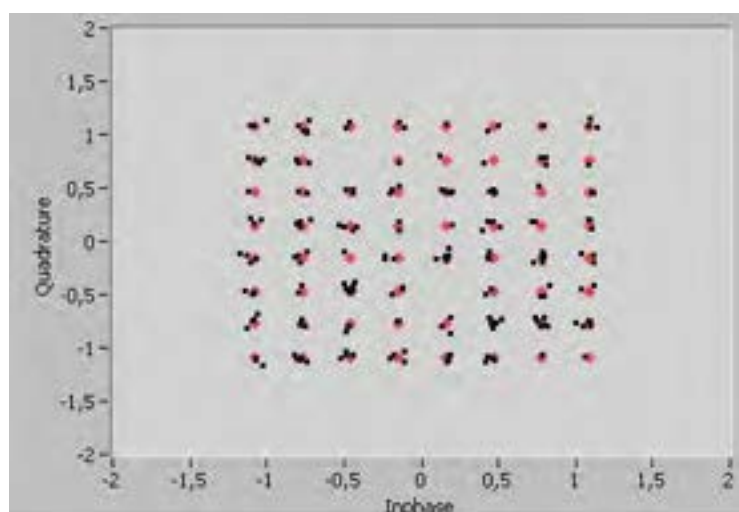


Figura 39 Mapping della costellazione 64-QAM

Nel caso specifico è stata simulata una modulazione QAM/OFDM su canale gaussiano, con quest'ultimo caratterizzato da media nulla e varianza dipendente dalla costellazione QAM ($M=16$ e $M=64$). In particolare, il legame tra la varianza e i parametri del sistema QAM/OFDM risulta essere la seguente:

$$\sigma = \sqrt{\frac{(M-1)N}{3S}}$$

Dove N/S è il rapporto rumore segnale, il quale è stato modificato nei vari passi di simulazione. Il grafico successivo mostra il valore del BER in funzione della variazione di N/S .

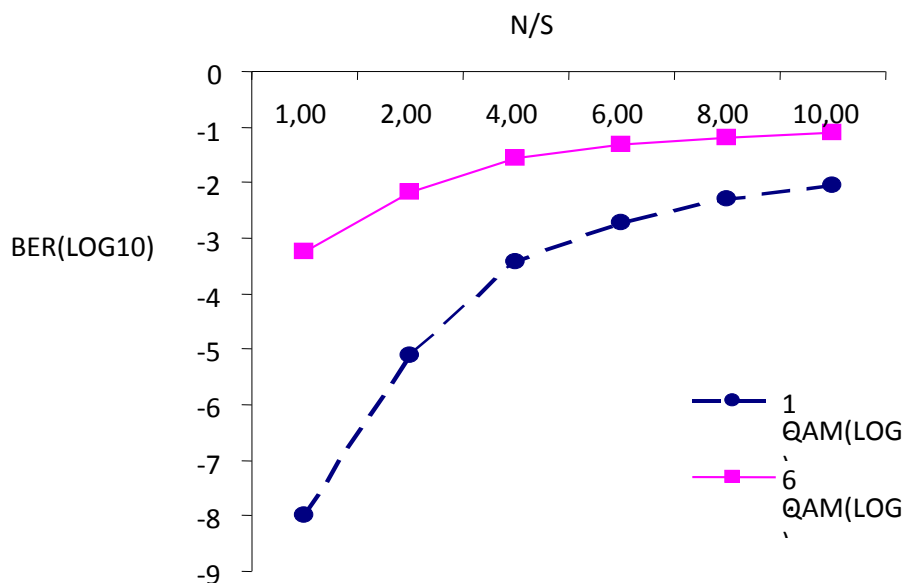


Figura 40 Andamento del BER in funzione del rapporto N/S per una trasmissione OFDM

Ogni valore del BER è calcolato per dieci milioni di bit relativamente ai due sistemi di modulazione 16-QAM e 64-QAM. Le considerazioni da fare, andando ad effettuare un paragone tra gli andamenti del BER ottenuti per i due tipi di modulazione studiati (FSK e OFDM), sono legate alle migliori prestazioni che è possibile ottenere, a parità di rapporto segnale rumore, con un sistema di trasmissione OFDM rispetto ad uno di tipo FSK. D'altronde, questo risultato non è altro che una conferma della bontà della tecnica OFDM, che presenta i ben noti vantaggi di una ridotta interferenza intersimbolica (ISI) e di una migliore immunità ai disturbi (avendo la possibilità di trasmettere l'informazione su più portanti). La tecnica FSK può comunque essere proficuamente utilizzata in applicazioni che richiedono una circuiteria semplice ma affidabile per la sua robustezza. Nelle PLC però, i limiti sono conosciuti e viene impiegata per trasmissioni a basso baud rate.

6.6 L'analizzatore Di Spettro Agilent E74002a

Prima di andare ad analizzare le caratteristiche tecniche dell'analizzatore di spettro che abbiamo adoperato per le nostre rivelazioni sperimentali (ovvero l'Agilent E74002A), cercheremo di capire come funziona e a cosa serve un strumento di questo tipo.

L'analizzatore di spettro è uno strumento in grado di misurare il contenuto armonico e la banda occupata di un determinato segnale elettrico. Il principio di funzionamento del dispositivo è il primo aspetto che andremo ad analizzare.

Partiamo da lontano, ricordando che un segnale può essere analizzato secondo due "punti di vista", cioè considerando l'evoluzione nel tempo di una grandezza elettrica oppure considerando il suo spettro nella banda delle frequenze. In gergo matematico si direbbe "dominio del tempo" e "dominio della frequenza", in pratica, potremmo dire che si tratta di due modi apparentemente diversi per rappresentare la stessa cosa. La differenza consiste nel tipo di informazioni che vengono offerte all'osservatore, che potrebbe avere interesse a preferire l'una o l'altra a seconda del tipo di applicazione. Nel caso di analisi nel dominio del tempo lo strumento che si usa è l'oscilloscopio, in grado di rappresentare l'evoluzione nel tempo di una tensione applicata al suo ingresso; nel caso di analisi nel dominio della frequenza si utilizza l'analizzatore di spettro, meno adatto dell'oscilloscopio a rappresentare segnali variabili nel tempo, ma in grado di fornire indicazioni sulla distribuzione dell'energia del segnale in una certa banda di frequenze. La teoria matematica di Fourier dimostra che un segnale periodico (ad esempio onda quadra, triangolare, ecc.) è composto da un insieme di sinusoidi, aventi ciascuna una frequenza di oscillazione pari a un multiplo della "fondamentale". Tali sinusoidi altro non sono che le cosiddette "armoniche", presenti sempre nello spettro di un segnale periodico qualsiasi. Con un'unica eccezione: il caso in cui il segnale fondamentale in questione non sia già una sinusoide ideale. Solo e soltanto in questo caso (è matematicamente dimostrabile) non troveremmo alcuna armonica. Non appena da sinusoide ideale il segnale si trasforma in qualcosa di diverso, allora cominciano a comparire le armoniche, che in un certo senso rappresentano quindi tutto ciò che differisce da una "sinusoide ideale". L'analizzatore consiste, sostanzialmente, in un ricevitore in grado di sintonizzarsi in modo ripetitivo su tutte le frequenze di una determinata banda impostata. I segnali presenti nella suddetta banda danno origine, nel momento in cui vengono sintonizzati, ad altrettante "righe" verticali nella traccia sullo schermo, collocate in un certo punto dell'asse orizzontale (asse delle frequenze), e con altezza proporzionale alla rispettiva potenza (spettro a righe di un segnale). Per poter operare in tal senso, l'architettura dello strumento prevede i seguenti blocchi funzionali:

- Generatore di rampa;
- Mixer di ingresso;
- Filtro passa basso di ingresso;
- Attenuatore d'ingresso;
- Filtri di frequenza intermedia (per ciascuna conversione);
- Amplificatori di frequenza intermedia;
- Filtri di risoluzione;
- Rivelatore di inviluppo;
- Blocco di rappresentazione sullo schermo.

Andiamo, ora, a descrivere il funzionamento dello strumento.

Il segnale di ingresso, dopo aver attraversato il filtro passa basso che limita la massima frequenza della banda di misura (al fine di evitare risposte indesiderate dovute alla banda immagine), raggiunge l'attenuatore, controllabile dall'operatore, al quale è demandato il compito di adeguare il livello di potenza del segnale a quello massimo accettabile dal primo mixer. Alle porte di ingresso di quest'ultimo vengono applicati rispettivamente il segnale di misura e quello proveniente dal primo oscillatore locale. Grazie al fatto che il primo oscillatore locale ha la proprietà di variare la sua frequenza di oscillazione in funzione del valore della tensione applicata alla porta di controllo (una rampa periodica), la frequenza generata subirà una variazione tra un minimo e un massimo valore (impostati dall'operatore, "SPAN"). Supponiamo, ad esempio, di avere un'escursione compresa tra 1500 e 1600 MHz, e che la frequenza centrale del primo filtro di IF sia 1000 MHz. In questo caso il mixer opererà una conversione tale che si otterrà un'escursione di frequenza nella sintonizzazione della banda di misura compresa tra 500 e 600 MHz, che è quella che risulterà rappresentata nello schermo. Ogni volta che, durante questa esplorazione di banda, in corrispondenza di un particolare valore di frequenza viene sintonizzato un

segnale, si osserva un "guizzo" verso l'alto della traccia sullo schermo per tutto il tempo in cui tale segnale rimane entro la banda del filtro di IF (infatti la frequenza di sintonia si sposta in continuazione, proprio come se l'operatore girasse la "manopola della sintonia" in modo continuo da un estremo all'altro della banda). La rappresentazione della riga spettrale, in realtà, ricalcherà quella della funzione di trasferimento del filtro di IF, proprio a causa di questo incrociarsi nel tempo tra la risposta in frequenza del filtro (una sorta di "campana") e la riga ideale, di spessore infinitesimo, della componente spettrale del segnale in esame.

Quanto più stretta sarà la banda del filtro, tanto migliore sarà la risoluzione, ovvero la capacità di discriminare due segnali posti a frequenza molto vicina. In realtà, poiché non è possibile realizzare un filtro molto selettivo a una frequenza così alta, per migliorare la risoluzione dello strumento si operano diverse conversioni in discesa e altrettanti filtraggi sempre più selettivi, fino al vero e proprio filtro di risoluzione (RBW, Resolution Band Width). La larghezza di banda di quest'ultimo è impostabile dall'operatore, e determina la risoluzione finale. Il "motore" di tutte queste operazioni è il generatore di rampa, che pilota, contemporaneamente, l'oscillatore locale (determinandone l'escursione di frequenza) e la deflessione orizzontale nello schermo (sviluppando in una traccia sull'asse orizzontale la rappresentazione della banda di frequenze esplorata ad ogni singola rampa). Non è possibile impostare in modo totalmente arbitrario i valori di SPAN (ampiezza della banda esplorata), RBW (risoluzione) e SWEEP TIME (velocità di esplorazione della banda, che dipende dalla frequenza della rampa): infatti, all'aumentare della risoluzione deve necessariamente essere ridotta la velocità di esplorazione, oppure l'ampiezza della banda (o tutti e due), pena l'introduzione di un notevole errore nella misura. Tale errore è determinato dal fatto che più la banda di un filtro è stretta, più lungo è il tempo di risposta dello stesso.

L'esplorazione dovrà essere dunque eseguita più lentamente al fine di consentire la corretta risposta al filtro di risoluzione che, altrimenti, "non farà in tempo" a far salire la traccia fino a rappresentare il corretto valore di ampiezza. In generale, per evitare errori di impostazione conviene utilizzare la modalità di accoppiamento automatico tra SPAN, SWEEP TIME e RBW, che, impostato uno dei tre parametri, ricava automaticamente il migliore valore per i rimanenti due. Un altro circuito di cui vale al pena parlare è il rivelatore di inviluppo, costituito da un rivelatore di picco (nei modelli più costosi è presente anche il "quasi picco", per le misure EMC) che ha il compito di filtrare la componente di alta frequenza in uscita all'ultimo filtro di IF (quello di risoluzione, centrato, in genere, intorno ai 3 MHz) ma di lasciar passare le variazioni dell'inviluppo (più lente), quelle che consentono il formarsi sullo schermo delle righe spettrali.

Possiamo riassumere brevemente le principali cause di errore nelle misure di ampiezza che caratterizzano l'analizzatore di spettro:

- Incertezza sull'attenuatore d'ingresso.
- Incertezza sul guadagno dell'amplificatore IF.
- Incertezza sulla risposta di ampiezza dei filtri di risoluzione.
- Possibilità di compressione e di risposte spurie da parte del mixer d'ingresso.

Oltre a queste, altre cause di errore sono certamente presenti:

- Non linearità dell'amplificatore logaritmico che fornisce la rappresentazione in scala lineare in dB.
- Imprecisione del segnale di calibrazione interno (rispetto al quale viene allineato lo strumento).
- Disadattamento dell'impedenza d'ingresso (la quale può produrre una riflessione del segnale di misura con conseguente riduzione della sua ampiezza), specialmente agli estremi della banda di misura.
- Risposta in frequenza non piatta.

Alla fine non dobbiamo meravigliarci se scopriamo che, sommando tutte le cause di possibili incertezze, i migliori strumenti in assoluto non riescono a garantire un'accuratezza migliore di ± 1 dB, che passa a ± 2 dB o peggio per modelli più economici. Si ricorda che 1 dB rappresenta circa il 25% della potenza. Una tale precisione viene in genere garantita, oltretutto, solo appena dopo l'effettuazione della

calibrazione e prima che l'operatore prema qualsiasi tasto: ogni cambio di funzione potrebbe infatti introdurre un piccolo errore casuale. Questa consapevolezza deve essere sempre presente quando si effettuano misure "di precisione", che dovrebbero sempre essere accompagnate da una stima dell'errore. Passiamo ora a descrivere alcune "voci" che caratterizzano le prestazioni dell'analizzatore di spettro e

che sono comunemente riportate nei cataloghi di presentazione dello strumento. La loro corretta interpretazione consentirà un più agevole confronto tra strumenti diversi.

6.6.1 Rumore di fase

Si tratta del rumore, con componenti sia di ampiezza che di fase, visibile accanto alle righe spettrali a livelli di ampiezza molto bassi. Tale rumore, in parte compensato internamente dai circuiti ad aggancio di fase degli oscillatori, e tanto più basso quanto migliore è la qualità (ed il prezzo) dello strumento, può rendere difficoltosa la misura di segnali di debole intensità molto vicini ad un segnale molto più forte. Viene specificato in termini di dBc/Hz a 10 o 100 kHz di distanza dalla portante.

6.6.2 FM residua

In questo caso la componente di rumore è una modulazione di frequenza spuria visibile "in cima" alla riga spettrale. Appare come una classica modulazione di frequenza, compensata pure in questo caso, entro certi limiti, dai circuiti interni a PLL. La FM residua non consente di utilizzare dei filtri di risoluzione più "stretti" di un certo valore in quanto la migliore risoluzione ottenibile risulterebbe vanificata. Se, ad esempio, un certo strumento fosse affetto ad una FM residua di 1 kHz picco a picco sarebbe inutile dotarlo di filtri di risoluzione aventi una larghezza di banda inferiore al kHz in quanto, utilizzandoli, non si otterrebbe una migliore risoluzione tra due righe spettrali vicine ma soltanto una loro sovrapposizione dovuta alla modulazione FM.

6.6.3 Range dinamico

E' la massima differenza di ampiezza tra due segnali che riescono ed essere correttamente misurati. Il range dinamico risulta limitato da vari fattori: in alto, dalla massima ampiezza di segnale accettabile in ingresso dallo strumento prima della sua rottura o di una eccessiva compressione del segnale di misura; in basso, dalla distorsione del mixer (quando si intendono effettuare misure di armoniche o di intermodulazione) e dal livello di rumore intrinseco nello strumento, ovvero dalla sua sensibilità. La limitatezza del range dinamico è la causa che impedisce una misura di precisione del livello delle armoniche di un segnale senza ricorrere all'ausilio di un filtro elimina banda sintonizzato sulla fondamentale.

6.6.4 Sensibilità

Rappresenta il valore di ampiezza del segnale più debole che è possibile misurare. La migliore sensibilità la si ottiene ponendo a 0 dB il valore di attenuazione in ingresso (per migliorare la cifra di rumore dello strumento) ed utilizzando la minima larghezza di banda del filtro di risoluzione (per ridurre la potenza di rumore che passa attraverso la banda del filtro). Quindi, aumentando progressivamente il guadagno dell'amplificatore di IF, ovvero diminuendo il valore in dBm del Reference Level, è possibile far salire verso l'alto la traccia del rumore fino a poterla correttamente misurare. Un buon analizzatore dovrebbe presentare un valore di sensibilità non inferiore, mediamente, ai -110 dBm. In generale, per misurare segnali di debole intensità bisogna selezionare il filtro di risoluzione più spinto e ridurre a 0 il valore di attenuazione dell'attenuatore in ingresso.

6.6.5 Fattore di forma dei filtri di risoluzione:

Abbiamo già accennato al fatto che se dal punto di vista matematico un segnale sinusoidale è rappresentabile con una riga (quella che l'analizzatore di spettro vuole rappresentare), in realtà tale riga viene ricavata mediante un filtro passa banda che mette a disposizione la sua funzione di trasferimento per rappresentarla nel modo migliore possibile. Più pendente e stretto sarà il filtro, migliore sarà la somiglianza con una riga ideale e, soprattutto, maggiore sarà la probabilità di individuare e misurare altre righe spettrali di debole intensità molto prossime ad altre di maggiore ampiezza. Il fattore di forma è specificato fornendo due valori di larghezza di banda: a -3 dB ed a -60 dB; il rapporto tra questi due valori è per l'appunto il fattore di forma. Supponiamo che il filtro in questione abbia una larghezza di banda di 10 kHz a -3 dB rispetto alla sua frequenza centrale, e di 150 kHz a -60 dB. Potremmo dire che il suo fattore di forma vale 15, sicuramente peggiore di quello di un filtro che garantisce, ad esempio, 5. In questo secondo caso la banda a -60 dB sarebbe di soli 50 kHz contro i 150, e si avrebbe una migliore risoluzione nei confronti di segnali di debole intensità molto prossimi ad altri più forti.

6.6.6 Risposte spurie:

Pure in assenza di segnale in ingresso è possibile che vengano rappresentate delle righe spettrali di ampiezza molto modesta, sia vicino che lontano rispetto alla portante (il segnale di "zero"). Queste sono dovute ad interferenze all'interno dello strumento (alimentatore switching) o a filtraggi insufficienti di frequenze di conversione. In generale, il livello di tali spurie si aggira sull'ordine dei -100 dBm.

Caratteristiche dell'Agilent E74002A:

- Resistenza agli urti ed alle vibrazioni: conforme a standard MIL.
- Tempo di warm-up: <15 minuti.
- Gamma di frequenza: da 30 Hz a 3 GHz.
- Span in frequenza: 0 Hz (zero span), da 100 Hz a 3 GHz.
- Incertezza dello span in frequenza: $\pm 0.5\%$ dello span.
- Risoluzione in frequenza del contatore: da 1 Hz a 10 kHz.
- Base dei tempi: aging $< \pm 10^{-7}$ /anno; stabilità con la temperatura $< \pm 10^{-8}$.
- Filtri RBW: da 10 Hz a 5 MHz (-3 dB) in passi 1-3.
- Bande EMI: 200 Hz, 9 kHz, 120 kHz (-6 dB).
- Filtri video: da 1 Hz a 3 MHz in passi 1-3.
- Rumore di fase SSB: < -118 dBc/Hz a 3 GHz.
- Tempo di sweep: 5 ms - 2000 s; 20 ms - 2000 s (zero span);
- Gated sweep per misure su segnali TDMA: range 1 ms - 100 s.
- Accuracy tempo di sweep: $\pm 1\%$.
- Range d'ampiezza: max livello DC 50 V; average continuous power +30 dBm (1W); peak pulse power +30 dBm (1W).
- Massima incertezza nella misura di livello: $< \pm 0.6$ dB.
- Livello di rumore medio visualizzato: 1 MHz - 10 MHz < -120 dBm; 10 MHz - 1 GHz: < -135 dBm; 1 GHz - 3 GHz: < -130 dBm; con preamplificatore: -130 dBm.
- Livello scala logaritmica: da 1 dB/div a 20 dB/div.
- Rivelatori: max peak, min peak, auto peak, sample, quasi peak, average, rms.
- Risposta in frequenza da 9 kHz a 3 GHz: < 0.5 dB.
- Trigger: free run, video, ext, IF lev; funzione gated sweep.
- Display a colori.
- Interfacce: GP.IB (IEEE-488); printer (compatibile Centronics); connettore mouse; connettore per monitor ext. VGA.
- Capacità di archiviare i risultati su supporto magnetico da 3.5", con possibilità di memorizzare almeno 400 tracce ed il setup dello strumento.

- Formato dei dati archiviati importabile da Excel.
- Possibilità di introdurre i fattori di correzione per antenna.
- Gestione completa di tutte le funzioni tramite PC esterno.
- Drivers software per LabView e LabWindows.
- Certificato di calibrazione.
- Circuito per il controllo di un generatore di tracking esterno.
- Interfaccia LAN.
- Alimentazione a batteria.



Figura 41 L'analizzatore di spettro AGILENT E74002A

6.7 Caratterizzazione della Linea Elettrica

Il secondo obiettivo (dopo quello del calcolo del BER) che si è posto, dal punto di vista sperimentale, è stato quello della caratterizzazione della rete elettrica, sempre tramite l'utilizzo di un modem powerline ST7537.

Per raggiungere tale scopo abbiamo utilizzato un cavo bifilare flessibile della ICEL (omologato IMQ n° A2447 e prodotto secondo le norme CEI 20-22 vigenti) dello spessore di $1,5 \text{ mm}^2$ e della lunghezza di 100 m . Tale cavo è stato, tra una misura e la successiva, decurtato di 5 m , in modo da avere la possibilità di effettuare 20 misure complessive, su distanze via via decrescenti.

Lo strumento di misura utilizzato è stato l'analizzatore di spettro Agilent E74002A (di cui ne abbiamo precedentemente spiegato le caratteristiche). In pratica, lo scopo dell'analizzatore era quello di andare a misurare, in funzione della distanza, il cosiddetto "output transmit level", misurato in $\text{dB}\mu\text{V}$, ovvero il segnale trasmesso dal modem powerline ST7537. Le figure seguenti ci danno un'idea del tipo di misura che è stata effettuata.

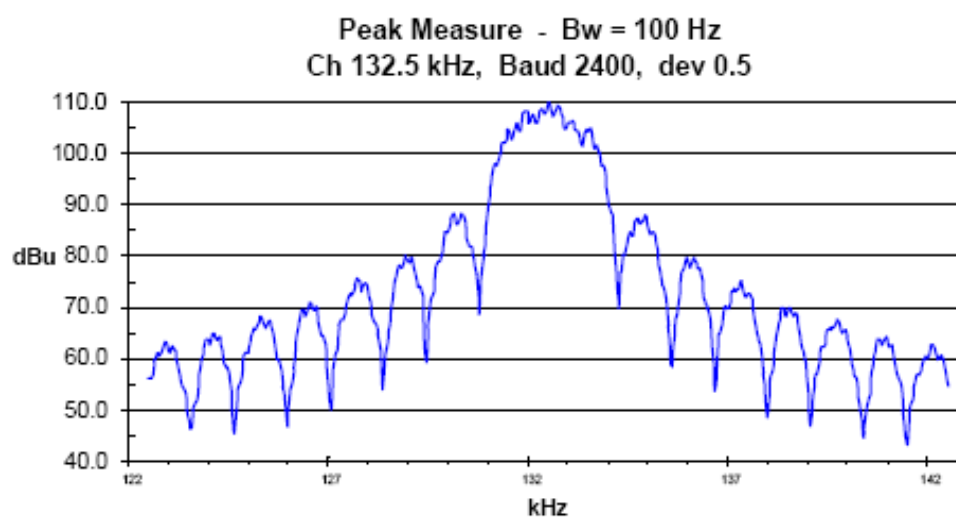
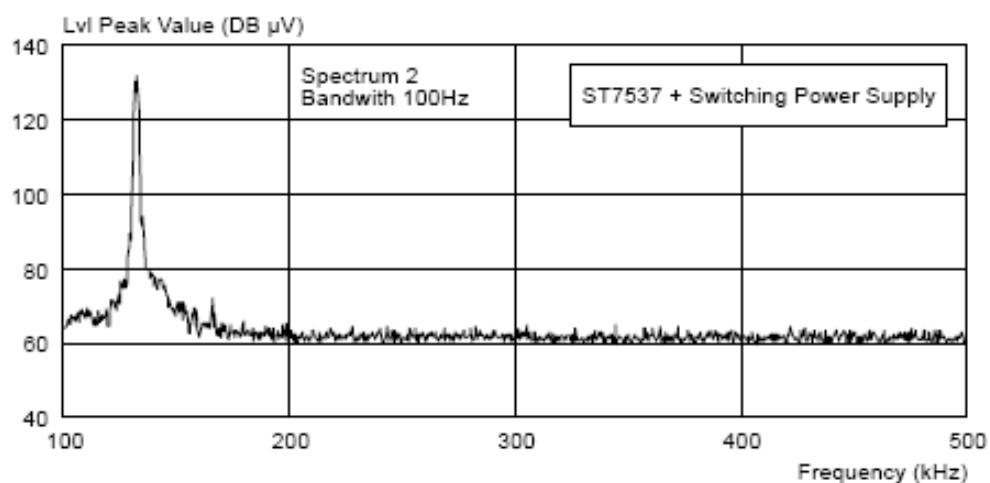


Figura 42

Portante presente intorno ai 132,45 kHz

Appare chiara la presenza della portante attorno alla frequenza di 132,45 kHz, come avevamo già precedentemente preannunciato. Andando a misurare il valore di picco (in $dB\mu V$) di questo segnale, al termine di tutte le misure è stato possibile ricavare l'attenuazione della linea tramite il grafico seguente:

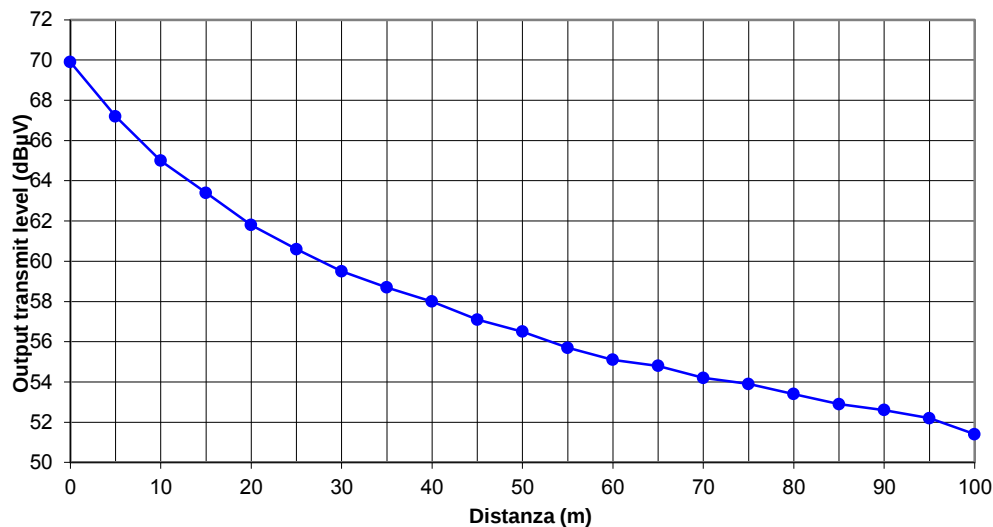


Figura 43 Caratterizzazione del canale

Si può notare come il segnale si attenua di circa 20 $dB\mu V$ nell'arco di 100 m. L'attenuazione risulta essere maggiore nelle prime misure (cioè per distanze più piccole): una causa di questo comportamento è data dal fatto che l'effetto di entità parassite, che incide sulle piccole distanze, risulta essere sempre più trascurabile all'aumentare della distanza di collegamento. Se a brevi distanze risulta incidere sulla misura il solo effetto resistivo, all'aumentare della distanza l'effetto reattivo inizia ad essere significativo e ciò si traduce in un apprezzabile aumento dell'attenuazione del canale. Riportiamo nel seguito anche l'andamento dell'attenuazione nel caso in cui il modem non sia connesso alla powerline; è un aspetto che non ha particolare interesse pratico ma che comunque può aiutare a capire meglio il problema.

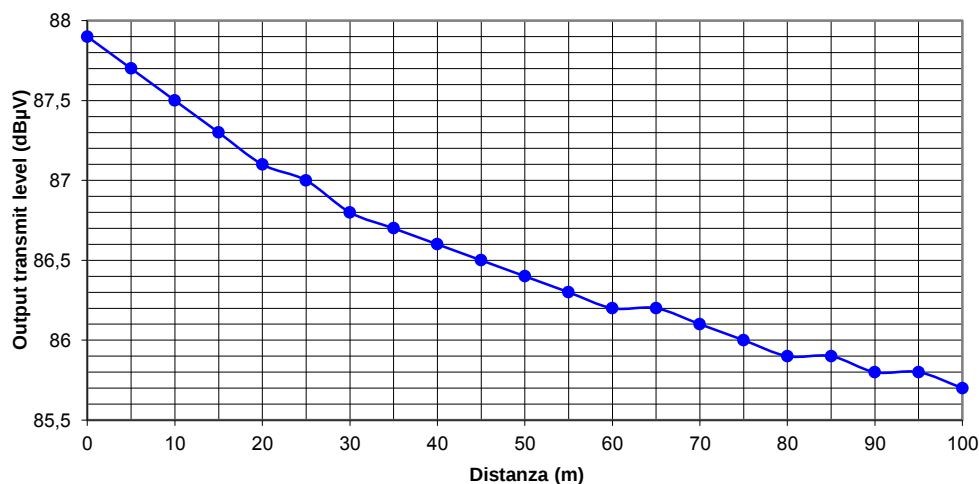


Figura 44 Andamento dell'attenuazione nel caso di non connessione alla powerline

A questo punto, è utile capire quali sono gli effetti dell'attenuazione del segnale e della variazione di impedenza di una linea di potenza. Sono due aspetti che tratteremo singolarmente ma questo non vuol dire che siano completamente indipendenti fra di loro.

Le cause che determinano l'attenuazione del segnale nelle linee di potenza sono diverse. E 'stato osservato che, tipicamente, il segnale inviato in un cavo elettrico non utilizzato per la trasmissione dell'energia elettrica richiede un livello di potenza sufficiente per la comunicazione tra i 3 e i 5 dB inferiore rispetto a quello delle powerline. La prima causa di questo fenomeno è da attribuire al segnale di accoppiamento tra le tre fasi della linea di potenza. La seconda è invece dovuta alle modalità di collegamento dei carichi alla linea elettrica. In figura 43, ricavata per distanze di circa 30 m, è indicato il diverso comportamento della linea elettrica al variare della frequenza.

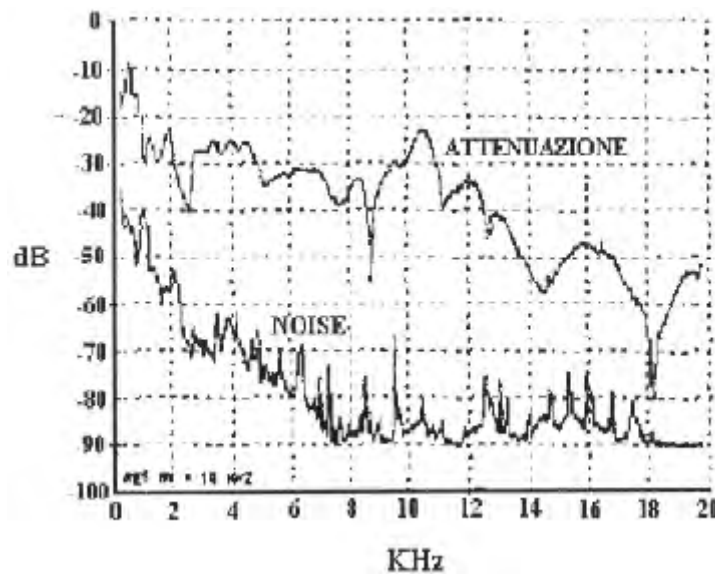


Figura 45 Andamento dell'attenuazione al variare della frequenza

Nella figura 44 viene, invece, rappresentato il diverso comportamento della linea elettrica al variare del carico.

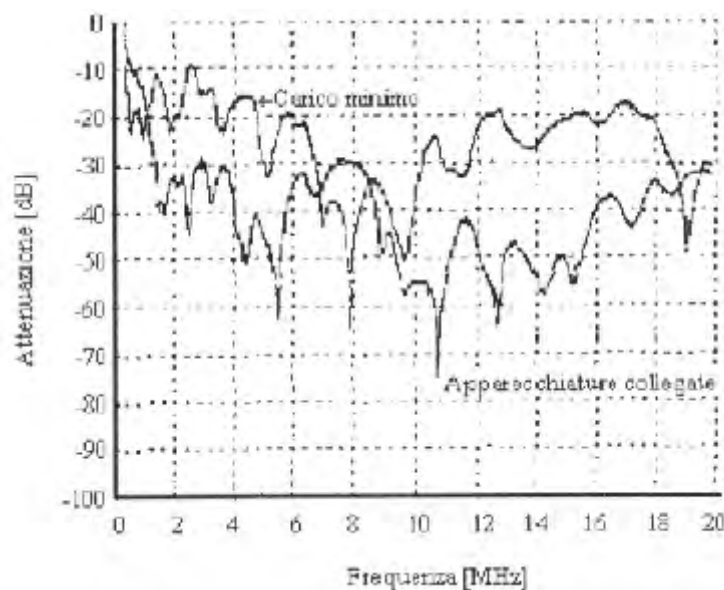


Figura 46 Andamento dell'attenuazione al variare del carico

Quando le apparecchiature vengono collegate alla linea, c'è un aumento dell'attenuazione da 20 a 30 dB rispetto all'attenuazione media che aumenta all'aumentare dei carichi collegati alla linea.

Questi peggioramenti sono dovuti alle risonanze causate dagli effetti reattivi associati alle apparecchiature alimentate, presenti in linea, che generano veri e propri filtri risonanti. Sono stati trovati, invece, risultati interessanti facendo il paragone fra una linea con apparecchi collegati all'alimentazione tutti in un punto della linea (cioè tutti alla stessa presa) ed apparecchi collegati, come in genere accade, in punti diversi della linea (cioè ognuno alla sua presa). Si è visto che nel caso di apparecchiature collegate tutte alla stessa presa di corrente elettrica si ha un'attenuazione rispetto al caso di apparecchiature collegate a prese elettriche diverse. Possiamo quindi affermare che l'attenuazione varia sia con l'ubicazione dei carichi che con la variazione di carico nel tempo sulla linea di potenza.

Non sempre, infatti, un carico è collegato alla stessa presa, ma il più delle volte non ha una sistemazione fissa, tutto ciò crea una variazione acuta dell'attenuazione con picchi imprevedibili nel tempo.

Per quanto riguarda la variazione dell'impedenza, da prove effettuate su spezzoni di cavo si è visto che l'impedenza varia significativamente con la frequenza in un range che va da qualche Ω a qualche k Ω . Esistono valori di frequenza in corrispondenza dei quali l'impedenza raggiunge dei picchi molto alti e la rete si comporta come un circuito risonante parallelo.

In questo range di frequenza la rete mostra, anche, un comportamento induttivo o capacitivo. L'impedenza caratteristica di un cavo powerline è di circa 90 Ω . L'impedenza di rete è influenzata oltre che dall'impedenza caratteristica della linea, anche dalla topologia della rete e dai carichi collegati che possono far variare in maniera decisiva le impedenze e di conseguenza l'impedenza della linea. A frequenze particolari, la linea introduce effetti non lineari che causano disadattamento di impedenza con conseguenti riflessioni di segnali ed echi.

Le riflessioni e la discontinuità di impedenza, causano echi del segnale trasmesso che generano variazione di impedenza e quindi attenuazione.

7 Riferimenti bibliografici

1. G. Bucci, E. Fiorucci, F. Ciacetta, M. Luiso, "Measuring System for Micro Electric Power", Instrumentation and Measurement, IEEE Transactions on, ISSN: 00189456, Volume: 63, Digital Object Identifier: 10.1109/TIM.2013.2280475, Publication Year: 2014 , Page(s): 410 – 421.
2. D. Gallo, C. Landi, M. Luiso, E. Fiorucci, "Survey on Voltage Dip Measurement in Standard Framework", Instrumentation and Measurement, IEEE Transactions on, ISSN: 00189456, Volume: 63 , Issue: 2, Digital Object Identifier: 10.1109/TIM.2013.2278996, Publication Year: 2014 , Page(s): 374 – 387.
3. G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, "FPGA-based real time compensation method for medium voltage transducers", ACTA IMEKO, ISSN: 2221-870X, in press.
4. G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, "Medium Voltage Divider Coupled With an Analog Optical Transmission System", Instrumentation and Measurement, IEEE Transactions on, ISSN: 00189456, Volume 63, Issue 9, Digital Object Identifier: 10.1109/TIM.2014.2317294.
5. G. Aurilio, M. Balato, G. Graditi, C. Landi, M. Luiso, and M. Vitelli, "Fast Hybrid MPPT Technique for Photovoltaic Applications: Numerical and Experimental Validation," Advances in Power Electronics, vol. 2014, Article ID 125918, 15 pages, 2014. doi:10.1155/2014/125918.
6. D. Gallo, C. Landi, M. Luiso, A. Rosano, M. Landi, V. Paciello, "Testing Protocols for Battery Characterization", Proceedings of IEEE International Instrumentation and Measurement Technology Conference I2MTC 2014, Montevideo, Uruguay, 12-15 May 2014.
7. G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, C. Cherbaucich, P. Mazza, "Low Cost Measurement Equipment for the Accurate Calibration of Voltage and Current Transducers", Proceedings of IEEE International Instrumentation and Measurement Technology Conference I2MTC 2014, Montevideo, Uruguay, 12-15 May 2014.
8. G. Aurilio, D. Gallo, C. Landi, M. Luiso, G. Graditi, "A Low Cost Smart Meter Network for a Smart Utility", Proceedings of IEEE International Instrumentation and Measurement Technology Conference I2MTC 2014, Montevideo, Uruguay, 12-15 May 2014.
9. G. Aurilio, D. Gallo, C. Landi, M. Luiso, V. Cigolotti, G. Graditi, "Low cost combined voltage and current transducer for Smart Meters", Proceedings of IEEE International Instrumentation and Measurement Technology Conference I2MTC 2014, Montevideo, Uruguay, 12-15 May 2014.
10. E. Fiorucci, G. Bucci, F. D'Innocenzo, M. Luiso, "A Measurement Technique for Quality Control of Windings for Cast Resin Power Transformers", Proceedings of IEEE International Instrumentation and Measurement Technology Conference I2MTC 2014, Montevideo, Uruguay, 12-15 May 2014.
11. G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, "A characterized method for the real-time compensation of power system measurement transducers", Proceedings of IEEE Conference on Precision Electromagnetic Measurements CPEM 2014, Rio de Janeiro, Brazil, 24-29 August 2014.
12. F. Barra, L. Costanzo, G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, "DIVISORE IN MEDIA TENSIONE CON USCITA ANALOGICA ED ISOLAMENTO IN FIBRA OTTICA", Atti del XXXI Congresso Nazionale del Gruppo di Misure Elettriche ed Elettroniche GMEE 2014, 11-13 Settembre 2014, Ancona.
13. G. Aurilio, D. Gallo, G. Graditi, C. Landi, M. Luiso, "RETE DI SMART METER A BASSO COSTO PER UNA SMART UTILITY", Atti del XXXI Congresso Nazionale del Gruppo di Misure Elettriche ed Elettroniche GMEE 2014, 11-13 Settembre 2014, Ancona.
14. G. Crotti, D. Gallo, D. Giordano, C. Landi, M. Luiso, C. Cherbaucich, P. Mazza, "UN SISTEMA DI

MISURA A BASSO COSTO PER LA TARATURA DI TRASDUTTORI DI TENSIONE E CORRENTE CONVENZIONALI E NON CONVENZIONALI”, Atti del XXXI Congresso Nazionale del Gruppo di Misure Elettriche ed Elettroniche GMEE 2014, 11-13 Settembre 2014, Ancona.

8 Curriculum scientifico del gruppo di lavoro impegnato nell'attività

Carmine Landi è nato a Salerno nel 1955. Ha conseguito la laurea in ingegneria elettrica nel 1981 presso l'Università di Napoli (Italia). E' stato ricercatore di Misure Elettriche dal 1983 al 1992 presso l'Università di Napoli, Professore Associato di Misure Elettriche ed Eletttroniche presso l'Università di L'Aquila, dal 1992 al 1999 e professore ordinario 1999-2002 presso la stessa università. Dal 2002 è Professore Ordinario presso la Seconda Università degli studi di Napoli. Dal 1998 è membro del comitato scientifico della collana editoriale Automazione delle Misure e Controllo edita dalla Franco Angeli. Dal 2001 è revisore di progetti di ricerca di rilevante interesse nazionale. Nel 2002 è stato inserito nell'albo degli esperti del MIUR per la valutazione di progetti di ricerca e sviluppo precompetitivo. Da ottobre 2007 è Presidente del CT 85/66-Strumentazione di misura, di controllo e da laboratorio del CEI (Comitato Elettrotecnico Italiano). Da novembre 2009 è Presidente del Consorzio Interuniversitario MeSE di ricerca Me.S.E. – Metriche e tecnologie di misura sui Sistemi Elettrici. E' responsabilescentifico di importanti progetti di ricerca e sviluppo industriale finanziati da MiSE e MIUR. E' revisore delle più importanti riviste internazionali del settore Misure (IEEE Trans. on Instrumentation and Measurements, Measurement, Sensors and Interface, etc.). E' autore di più di trecento lavori scientifici pubblicati su riviste internazionali e nazionali ed in atti di congressi internazionali e nazionali.

Daniele Gallo è nato a S. Maria Capua Vetere (CE) il 04/08/1974. Si laurea in Ingegneria Elettronica il 30 Marzo 1999 presso la Seconda Università degli Studi di Napoli, con voti 110/110 e Lode. Consegue il titolo di Dottore di Ricerca in "Conversione dell'Energia Elettrica", presso la Seconda Università degli Studi di Napoli il 20 novembre 2003 con eccellente giudizio della commissione esaminatrice. Dal 2008 è ricercatore Universitario e professore aggregato del settore scientifico disciplinare ING-INF/07. Vincitore del primo premio nell'ambito della "IEEE Power Engineering Society 2001 Winter Meeting Student Poster Competition". Vincitore del primo premio per la migliore tesi di dottorato nell'ambito della selezione per "Best Thesis Awards in Power Engineering". Dal 2002/03 è stato docente di vari insegnamenti quali "Sistemi Automatici di Misura Laurea", "Misure per la Diagnostica di Componenti e Sistemi", "Misure per la Produzione e l'Automazione Industriale" per dei corsi di Laurea di Ingegneria Elettrica ed Ingegneria Elettronica di diverse facoltà quali l'Università degli studi de L'Aquila; Seconda Università degli Studi di Napoli, l'Università degli Studi di Napoli "Federico II. Autore di numerose pubblicazioni scientifiche tra cui: 1 standard internazionale; 27 lavori scientifici pubblicati su riviste internazionali. E' revisore delle più importanti riviste internazionali del settore Misure (IEEE Transaction on Instrumentation and Measurements, Measurement, etc.) e dei progetti scientifici di interesse nazionale.

Mario Luiso è nato a Napoli (NA) il 06/07/1981. Si laurea in Ingegneria Elettronica con votazione di 110/110 e Lode, Dottorato di Ricerca in "Conversione dell'Energia Elettrica" XX Ciclo. Dopo il dottorato di ricerca, dal 2008 ad oggi è stato titolare di diversi assegni di ricerca ed incarichi di collaborazione coordinata e continuativa, presso il Dipartimento di Ingegneria dell'Informazione della Seconda Università degli Studi di Napoli, il Centro Interdipartimentale di Tecnologie per la Diagnostica Ambientale e lo Sviluppo Sostenibile dell'Università degli Studi del Sannio, il Dipartimento di Ingegneria Elettrica dell'Università degli studi di Napoli Federico II. Dal 2013 è ricercatore e professore aggregato del settore ING-INF/07 presso la Seconda Università degli Studi di Napoli. E' autore di: 24 Pubblicazioni su riviste internazionali, 50 Pubblicazioni su atti di conferenze internazionali. Ha conseguito il "Premio Nazionale di Cultura Luigi Casati" per migliore tesi di laurea nel settore scientifico, conferito dall'ONAOISI; ed il "Diploma d'Onore" per la tesi di Dottorato di Ricerca dall'Accademia Ercolanese. È membro della Instrumentation and Measurement Society dell'IEEE. E' membro del comitato editoriale della rivista WSEAS Transactions on Power Systems.

Le principali tematiche di ricerca affrontate dal gruppo di ricerca riguardano: (a) La messa a punto di metodi di misura innovativi per la caratterizzazione, collaudo e diagnostica di componenti, apparecchiature e sistemi elettrici ed elettronici; (b) la progettazione, realizzazione e sviluppo di

strumenti di misura a microprocessori con architetture innovative, idonee a funzionare in tempo reale; (c) i metodi, i componenti ed i sistemi per la misura della potenza ed energia in regime distorto e la valutazione della qualità dell'alimentazione elettrica; (d) caratterizzazione di componenti e di impianti dell'energia elettrica da fonti rinnovabili, e) lo sviluppo di tecniche e dispositivi per la caratterizzazione e l'ottimizzazione delle prestazioni dei sistemi elettrici di potenza, f) la realizzazione di trasduttori di tensione e corrente ad alta precisione e larga banda per la misura di potenza elettrica e della sua qualità.