UNIVERSITÀ DEGLI STUDI DI PADOVA
Corso di Laurea in Ingegneria Elettronica

Relazione di tirocinio

Applicazioni di monitoraggio ambientale con gestione web-based

Relatore:
Paolo Tenti

Correlatore:
Marco Stellini

Laureando:
Daniele Dal Maistro

Anno accademico 2010/2011

Sommario

Il problema energetico è, da diversi anni, un tema centrale nell'ambito della ricerca scientifica. La necessità di sviluppare sistemi di produzione, distribuzione ed utilizzo dell'energia efficienti, unitamente alle attrattive di un mercato in forte crescita, hanno spinto e stanno spingendo aziende, centri di ricerca ed atenei a concentrare gli sforzi in questa direzione, e i risultati sono sotto gli occhi di chiunque: basti pensare a pannelli fotovoltaici, generatori eolici, sistemi di sfruttamento delle biomasse, per citare gli esempi più popolari.

La ricerca, come precisato, non si ferma agli aspetti relativi alla produzione dell'energia, in quanto un altro problema fondamentale sta nell'utilizzo efficiente della stessa. In questo contesto si inserisce il lavoro del Dipartimento di Ingegneria dell'Informazione dell'Università degli Studi di Padova, dove si sta sviluppando un sistema di monitoraggio e controllo ambientale. L'obiettivo è realizzare, nell'ambito di un progetto sulle *smart grid*, una rete di sensori e attuatori che consenta di conoscere nel dettaglio i consumi nelle varie aree degli edifici, di limitare il più possibile gli sprechi ed integrare la fornitura di energia elettrica dalla rete pubblica con la conversione realizzata internamente, sfruttando tecnologie diverse.

Il presente documento contiene una relazione sull'attività svolta dall'autore nel corso del tirocinio interno presso il Dipartimento. Quest'esperienza ha consentito di osservare ed utilizzare alcuni degli strumenti e tecnologie coinvolti nel progetto di monitoraggio e controllo ambientale.

Indice

Capitolo 1

Introduzione

L'energia rappresenta, da almeno due secoli, un settore di enorme importanza nell'interesse della collettività, tale da scatenare conflitti per l'approvvigionamento delle sue fonti. Parliamo di un settore ampio, vario ed in continua evoluzione, sin dalle origini del suo status di interesse primario per ogni nazione. Nel corso degli anni si sono verificate quelle che si possono definire delle rivoluzioni, nella produzione o nell'utilizzo dell'energia, e non soltanto recentemente: per esempio, se all'inizio del '900 l'energia ricavata dal vapore veniva impiegata nell'80% degli azionamenti meccanici, nei trent'anni successivi questa predominanza è stata progressivamente erosa dall'avvento dell'energia elettrica, che nel 1929 (soli 45 anni dopo il suo primo utilizzo in ambito industriale) copriva il 78% del settore suddetto[19].

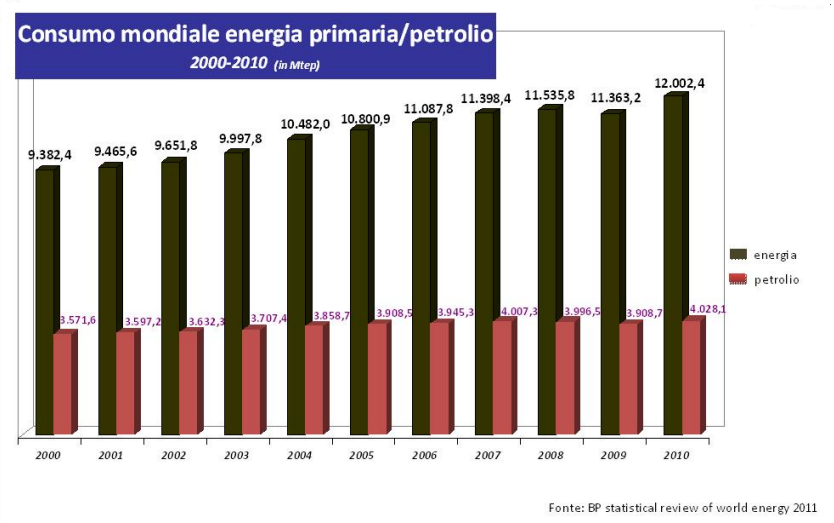
Oggi il mondo sta vivendo, ancora una volta, un'era di profonde trasformazioni nei settori della produzione, del consumo e del mercato energetici. Si tratta di un processo orientato, in linea generale, alla ricerca di nuove fonti energetiche, alla diminuzione dell'inquinamento, e ad un impiego più efficiente dell'energia disponibile, motivato da almeno tre ragioni:

- la necessità di invertire il trend di crescita che ha interessato i consumi energetici negli ultimi decenni, come illustrato in Fig. 1.1a[1];
- la necessità di ridurre le emissioni di gas serra, anch'esse cresciute notevolmente;
- la necessità di ridurre la dipendenza energetica dal petrolio, una fonte notoriamente non rinnovabile ed il cui utilizzo è causa di emissioni inquinanti, che tuttavia rimane tra le più impiegate.

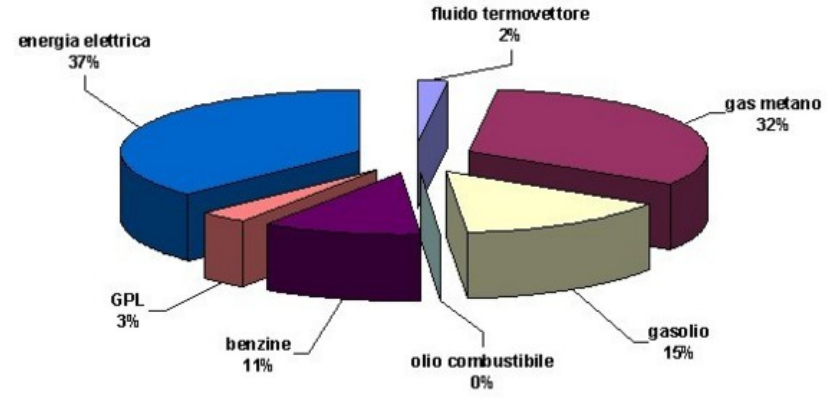
Per far fronte a queste problematiche, nel dicembre del 2008 l'UE ha adottato una strategia integrata in materia di energia e cambiamenti climatici, che fissa obiettivi ambiziosi per il 2020. Lo scopo è indirizzare l'Europa sulla giusta strada verso un futuro sostenibile sviluppando un'economia a basse emissioni di CO_2 improntata all'efficienza energetica. Gli obiettivi della strategia, denominata *Europa 20 20 20*, sono ridurre i gas ad effetto serra del 20%, ridurre i consumi energetici del 20%, e soddisfare il 20% del fabbisogno energetico mediante l'utilizzo delle energie rinnovabili.

[1] I consumi sono espressi in Mtep (mega tonnellate equivalenti di petrolio).

Figura 1.1: Statistiche su consumi energetici ed emissioni di CO_2



(a) Consumi energetici nel decennio 2000-2010



(b) Ripartizione delle emissioni di CO_2 per vettore energetico

L'energia elettrica ricopre, di nuovo, un ruolo centrale in questo processo di evoluzione. In effetti, come evidenziato dalla Fig. 1.1b, la produzione di questa forma di energia è tra le cause principali delle emissioni di CO_2, per effetto dei processi che sfruttano la combustione di materiali come il carbone. Tuttavia essa presenta anche il notevole vantaggio di essere *multi-sourced*, aggettivo che indica la possibilità di ottenerla a partire da fonti diverse: non soltanto dal petrolio o dal carbone, attraverso la combustione della materia prima, ma anche convertendo energia eolica, solare, idroelettrica, nucleare. Questa qualità rende l'energia elettrica una fonte molto versatile, e, assieme alla possibilità di impiegarla unitamente a tecnologie che consentono di elevarne l'efficienza, apre la strada al suo impiego in settori nei quali, tradizionalmente, dominano altre forme energetiche, come quello dei trasporti, che negli ultimi anni ha visto il lancio di vetture a trazione ibrida o addirittura interamente elettrica[3].

La rete elettrica nazionale fino ad oggi sostanzialmente è stata un canale pas-

sivo progettato per la distribuzione dell'energia dalle grandi centrali di produzione agli utenti finali. Il raggiungimento degli obiettivi presentati passa attraverso una modifica sostanziale della sua struttura, che è già in atto da alcuni anni, e che prevede l'integrazione dei flussi di energia generati dalle centrali con la produzione di media o piccola entità proveniente da fonti rinnovabili. La rete elettrica, dunque, non sarà più soltanto elettrotecnica, ma comprenderà anche elementi di elettronica, informatica e telecomunicazioni, che consentiranno di passare dall'attuale controllo centralizzato ad una vera e propria gestione distribuita dell'energia, che comprenda funzionalità di monitoraggio e controllo a livello locale. Si configura, dunque, l'idea di una rete intelligente, una *smart grid*, capace di far interagire produttori e consumatori e di gestire flussi bidirezionali di energia[2]. Benchè l'attività sulle reti intelligenti sorga in relazione diretta all'energia elettrica, questo non è l'unico campo di applicazione disponibile: infatti, una funzionalità estremamente importante in una *smart grid*, come si accennava, è il **monitoraggio ambientale**, inteso come controllo dei consumi di energia, che non è necessariamente soltanto elettrica.

L'Università di Padova, attraverso i dipartimenti DEI (Dipartimento di Ingegneria dell'Informazione) e DIE (Dipartimento di Ingegneria Elettrica), sta sviluppando un progetto di ricerca e sperimentazione finalizzato a realizzare una *micro smart grid* interna ai dipartimenti, che consenta di toccare con mano le potenzialità di queste tecnologie, e di comprenderne i campi di applicazione ed i limiti. Nello specifico, è stata realizzata un'architettura per il monitoraggio e controllo dei consumi, denominata *Pin-Energy*, che si compone di undici punti di acquisizione localizzati nei quadri elettrici più significativi dell'impianto dei dipartimenti. Tali 'nodi' sono rappresentati da dispositivi in grado di interagire con un sistema informatico che si occupa dell'elaborazione e della presentazione dei dati su un'apposita pagina web.

L'architettura descritta può essere anche efficacemente impiegata per monitorare i consumi relativi ad altre fonti energetiche, con l'unico onere di dover aggiungere l'hardware ed il software necessari all'interfacciamento con i nodi di acquisizione. Una parte della tecnologia coinvolta, inoltre, può essere impiegata per applicazioni non direttamente legate ai consumi energetici, come il monitoraggio meteorologico, che comunque si inserisce nell'ambito del monitoraggio ambientale, dato che le informazioni ricavate possono essere utili per gestire più efficientemente, ad esempio, impianti di condizionamento o deumidificazione. Le finalità del progetto sono individuare e quantificare gli sprechi, migliorare l'efficienza energetica dei dipartimenti, e, naturalmente, contribuire allo sviluppo dell'attività di ricerca sulle smart grid.

Questo lavoro di tesi si è concentrato su alcuni aspetti di questa attività di ricerca, presentati e descritti nei capitoli successivi. E' riportata, nel seguito, una sintesi:

- **Capitolo 2**: in questa parte del documento viene descritta ed analizzata l'architettura del sistema di monitoraggio operativo presso i poli didattici DEI e DIE. Dopo la presentazione dell'architettura da un punto di vista strutturale, viene riportata una descrizione dell'hardware preposto alla misura e acquisizione dei parametri relativi all'impianto elettrico dei dipartimenti, per poi passare all'analisi dell'applicazione software che si occupa della elaborazione e presentazione dei dati.

- **Capitolo 3**: vengono presentate le soluzioni studiate al fine di aggiungere il gas metano ai vettori energetici monitorati, ricorrendo alla telelettura del contatore installato all'esterno di uno degli edifici del DEI. Il progetto prevede l'impiego di un trasmettitore, da collegare al misuratore di gas, ed un ricevitore, che si prevede di installare presso il nodo di acquisizione dell'architettura di monitoraggio più vicino. Dopo una breve descrizione del misuratore, viene presentato il protocollo di comunicazione impiegato per realizzare la comunicazione. In seguito vengono descritte nel dettaglio le due soluzioni proposte, con riferimento all'hardware ed al software necessari.

- **Capitolo 4**: descrive il kit per il monitoraggio meteorologico installato presso il dipartimento. In primo luogo, viene presentato l'hardware per la misura dei parametri meteorologici e per la comunicazione con la rete informatica del dipartimento. In seguito, vengono analizzate le caratteristiche della comunicazione wireless tra i due componenti del kit, il software fornito dalla casa produttrice ed i dettagli relativi all'implementazione fisica del sistema di monitoraggio meteorologico.

- **Capitolo 5**: nell'ultimo capitolo viene riassunta l'attività presentata ed analizzata in questo documento. Si evidenziano, inoltre, le difficoltà incontrate, ed i possibili sviluppi futuri.

Capitolo 2

Architettura Pin-Energy

Questo capitolo descrive l'architettura del sistema di monitoraggio installata presso i dipartimenti DEI e DIE, ed illustrata schematicamente in Fig. 2.1. La sua struttura si può suddividere in due parti essenziali, le quali, unitamente, consentono di analizzare i flussi dell'energia elettrica che interessa i due poli didattici:

- una parte legata alla rete di distribuzione dell'energia elettrica, composta dai *quadri elettrici di area Pin-Energy* e dal blocco indicato in figura come *Sorgente*. Le connessioni marcate con tratto nero, in figura, rappresentano le linee di distribuzione dell'energia elettrica;

- una parte costituita dall'infrastruttura informatica di comunicazione, composta da un server del Dipartimento, la sottorete DEI 172.16.10.xxx, e l'interfaccia web degli utenti con la rete (personal computer, palmare, smartphone ecc.). Le connessioni grigie, in figura, rappresentano le linee di comunicazione tra i dispositivi.

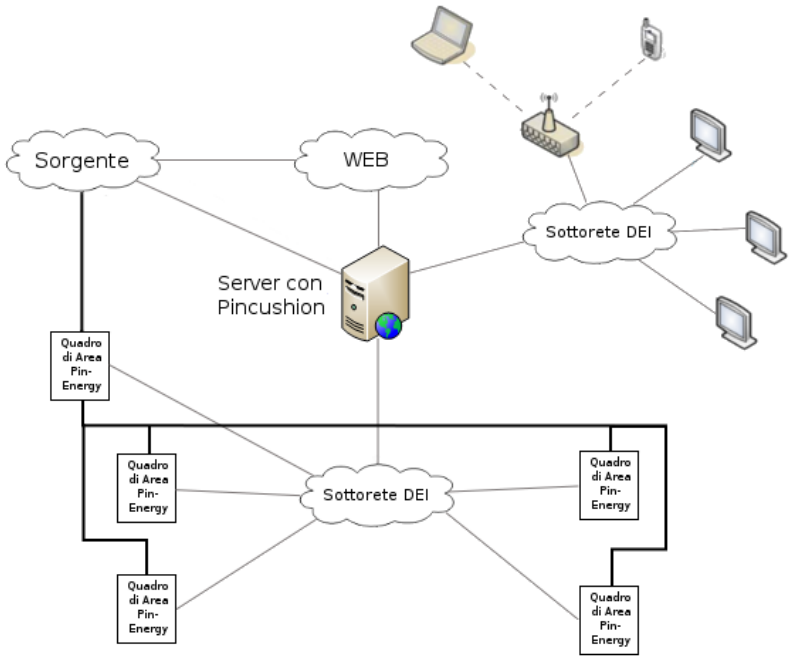
Per quanto riguarda la parte elettrica, il blocco *Sorgente* comprende gli elementi che permettono di realizzare l'interfaccia tra l'ente fornitore dell'energia elettrica e la rete interna del dipartimento (per approfondimenti si faccia riferimento a [1], Cap. 2), mentre i *quadri elettrici di area Pin-Energy* sono i quadri elettrici a cui fanno capo i principali gruppi di utenze. A titolo di esempio, si possono citare il quadro di alimentazione dell'impianto di condizionamento dell'aria o i quadri che alimentano i laboratori. Questi blocchi sono connessi, mediante un'apposita interfaccia, alla sottorete DEI 172.16.10.xxx, e consentono di collegare le due parti dell'architettura, mediante la comunicazione tra i dispositivi di misura localizzati nei quadri e l'infrastruttura informatica.

Focalizzando invece l'attenzione sulla parte inerente la struttura informatica, va evidenziata la funzione del server dipartimentale: su questo calcolatore è installata un'applicazione, denominata Pincushion, che consente di gestire la comunicazione con i quadri Pin-Energy ed i dati raccolti.

Nel dettaglio, i componenti fondamentali che consentono di realizzare le funzioni di monitoraggio sono:

- I moduli Nemo[4] ed iPin-Energy, localizzati all'interno dei quadri Pin-Energy, che svolgono le operazioni di misura, acquisizione e comunicazione;

Figura 2.1: Architettura del sistema di monitoraggio



- L'applicazione Pincushion, che si occupa di elaborare, memorizzare e presentare i dati raccolti dai moduli.

2.1 Moduli Nemo e iPin-Energy

In questa sezione vengono descritti i moduli preposti alla misura ed acquisizione delle grandezze di interesse, ed alla comunicazione con la rete dipartimentale.

2.1.1 Nemo D4-L+ e TA

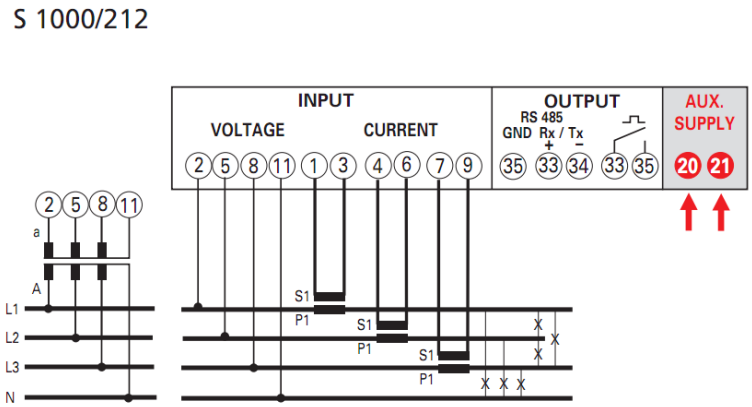
Questi due elementi rappresentano l'hardware di misura, dunque da essi dipende l'accuratezza, la precisione e la banda del sistema.

Il Nemo D4-L+ è uno strumento multifunzione per reti a bassa tensione che permette di misurare diverse grandezze elettriche relative a sistemi monofase e trifase a 3 o 4 fili. I parametri che possono essere rilevati sono i seguenti:

1. tensione di fase e concatenata;
2. corrente di fase e di neutro;
3. corrente media di linea e valore di picco della stessa;
4. frequenza;
5. fattore di potenza;

6. potenza attiva, reattiva o apparente del sistema;
7. potenza attiva e reattiva di fase;
8. potenza media e valore di picco della potenza media;
9. energia attiva e reattiva;
10. energia attiva parziale;
11. tempo di funzionamento;
12. rilevazione della sequenza delle fasi.

Figura 2.2: Schema d'inserzione del modulo per una rete trifase a 4 fili



Le misure vengono effettuate mediante collegamenti voltmetrici ed amperometrici, come evidenziato in Fig. 2.2: i primi, uno per fase, sui morsetti 2, 5, 8 e 11 del dispositivo, i secondi, sulle coppie di morsetti 1-3, 4-6, 7-9. E' possibile realizzare le inserzioni su trasformatori voltmetrici ed amperometrici (in sigla, TV e TA), nel caso in cui i parametri elettrici siano inadatti ad essere misurati direttamente dallo strumento. A questo proposito, il modulo dispone di un tastierino che consente di programmare i rapporti di trasformazione dei trasformatori, e di un display a cristalli liquidi per la visualizzazione delle misure. Nel caso in esame, sono stati impiegati soltanto i TA, mentre le tensioni vengono misurate direttamente. I range di tensione dichiarati per il funzionamento corretto del dispositivo sono [80, 450]V per le tensioni concatenate, [30, 260]V per le tensioni di fase, mentre le correnti devono appartenere all'intervallo [1, 5]A. Se queste condizioni vengono rispettate, la precisione delle misure di tensione e corrente è dichiarata essere $\pm 0.5\%$. Allo stesso modo, la precisione delle misure di frequenza è $\pm 0.2\%$, se si opera nel range [47, 63]Hz. Le misure di correnti e tensioni efficaci vengono effettuate ogni 1.2 s.

Va sottolineato che lo strumento presenta un proprio *autoconsumo*, di tensione e di corrente, ovvero assorbe una certa potenza durante il suo normale funzionamento, cosa che si traduce nell'introduzione di errori nelle misure effettuate. Per approfondimenti si rimanda alla scheda tecnica[4].

Per quanto riguarda l'interfaccia del modulo verso l'esterno, esso dispone di un'uscita RS485, con indirizzo, baud-rate e bit di parità programmabili, che viene utilizzata per la comunicazione con il modulo iPin-Energy destinato ad acquisire le misure.

Trasformatori amperometrici Questi trasformatori, come già precisato, vengono impiegati per consentire al modulo Nemo di misurare correnti il cui valore non rientri nell'intervallo di funzionamento corretto dichiarato dal costruttore, grazie alla conoscenza del rapporto di trasformazione, un coefficiente di proporzionalità che lega le correnti nominali[1] primaria e secondaria. Nello specifico, un TA è progettato affinchè sia rispettata la seguente relazione tra corrente primaria e secondaria:

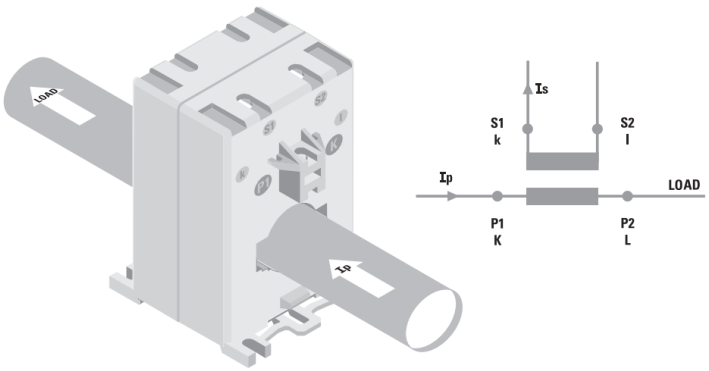
$$I_p \cdot N_p = I_s \cdot N_s,$$

dove I_p ed N_p sono la corrente ed il numero di spire dell'avvolgimento primario, mentre I_s ed N_s sono, analogamente, gli stessi parametri relativi al secondario. Premesso ciò, il rapporto di trasformazione, qui indicato con il simbolo C_t, si ricava dal rapporto tra i numeri di spire nel seguente modo:

$$C_t = \frac{I_p}{I_s} = \frac{N_s}{N_p}.$$

Come si può facilmente immaginare, queste relazioni fanno riferimento ad un modello puramente ideale del trasformatore, che può discostarsi anche notevolmente dalla realtà pratica. Studi più approfonditi portano a ricavare modelli maggiormente complessi e precisi, che tengono conto di alcune delle non idealità dello strumento, per i quali si rimanda al documento di riferimento[1] (Capitolo 2, sottosezione 2.1.2).

Figura 2.3: Schema d'inserzione del trasformatore amperometrico



Dal punto di vista pratico, ogni dispositivo di questo tipo fa riferimento ad uno standard che consente di definirne in modo univoco le prestazioni, per mezzo

[1]Le correnti nominali sono quelle su cui si basano le misure delle prestazioni del dispositivo; è quindi opportuno fare in modo che il dispositivo si trovi a funzionare in condizioni il più possibile prossime a quelle nominali.

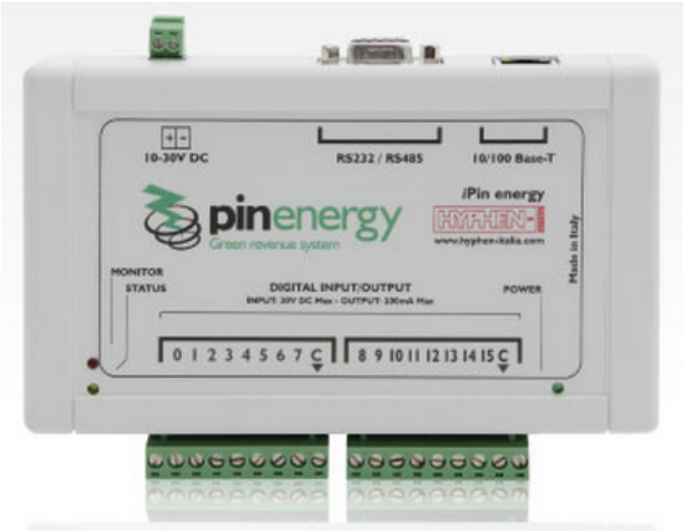
di alcuni parametri, come le correnti nominali o l'errore commesso nella stima della corrente primaria. I TA impiegati nel caso in esame sono degli IME TAID50B800[5], conformi allo standard IEC 60044-1, il cui schema d'inserzione è illustrato in Fig.2.3. Come si può osservare, il primario del trasformatore è rappresentato da una pinza che abbraccia il conduttore, e che rappresenta l'unica "spira" dell'avvolgimento. La pinza stessa è dunque sede del flusso magnetico indotto dalle corrente che fluisce nel cavo, e l'avvolgimento secondario è localizzato direttamente su di essa.

2.1.2 iPin-Energy

I moduli iPin-Energy, prodotti dall'azienda veronese Hyphen-Italia, rappresentano l'anello che lega le due parti in cui si può suddividere, a livello logico, l'architettura di monitoraggio, ovvero la parte elettrica e quella informatica.

Ciascun modulo è dotato di una porta seriale RS485, tramite la quale è posto in comunicazione con un modulo Nemo, dal quale raccoglie le misure. Inoltre esso dispone di un'interfaccia ethernet RJ45, che permette di collegarlo alla rete dipartimentale, e di utilizzarla per inviare le informazioni raccolte all'applicazione Pincushion. Oltre a queste porte di comunicazione, ciascun modulo dispone di un ingresso per l'alimentazione a $10 \div 30$V DC, e di 16 canali digitali optoisolati, configurabili come ingressi o come uscite, e in grado di accettare tensioni in ingresso fino a 30V ed erogare correnti in uscita fino a 200mA.

Figura 2.4: Il modulo iPin-Energy



Le potenzialità di questo dispositivo, però, non si fermano al semplice ruolo di 'ponte' tra i Nemo e Pincushion, in quanto esso è in grado di arricchire le informazioni acquisite con la generazione di statistiche, leggere i valori logici applicati ai canali configurati come ingressi, pilotare i canali configurati come uscite e gestire anche segnali analogici. Tutto questo va aggiunto alla possbilità di controllare il modulo da remoto, per mezzo di Pincushion e dell'applicazione iPinClient, che forniscono la possibilità di passare dal semplice monitoraggio ad applicazioni di vero e proprio

controllo ambientale: in quest'ottica, eventi acquisiti dal modulo iPin ed individuati da Pincushion possono dar luogo ad una reazione, attuata attraverso una delle uscite del modulo.

2.2 Pincushion

Pincushion è una piattaforma web per il building-management che offre una varietà di servizi e funzionalità nel campo dell'automazione e della sicurezza. Esempi delle potenzialità di questo software sono la gestione degli accessi di persone a locali, il controllo della sicurezza delle aree tecniche o, per quanto riguarda il caso in esame, l'ottimizzazione dei consumi energetici.

In questo contesto, Pincushion viene impiegato per configurare i punti di acquisizione, rappresentati dai moduli iPin, e per gestire i dati che questi ultimi raccolgono dai moduli Nemo, salvandoli in un apposito database, ed effettuando le elaborazioni necessarie per la presentazione all'utente.

L'applicazione fornisce un accesso tramite pagina web, che si effettua dal link http://energy.dei.unipd.it tramite un operazione di login che richiede l'inserimento di username e password. La pagina principale, che viene visualizzata una volta effettuato l'accesso, si compone di cinque sezioni:

1. sezione Viste: consente di visualizzare le viste create, ovvero pagine web che riportano del contenuto informativo nella forma stabilita dall'utente;

2. sezione Strumenti: consente di accedere agli strumenti per il controllo degli accessi, il brandeggio delle telecamere, la gestione del calendario, ecc. ;

3. sezione Statistiche: consente di visualizzare le statistiche elaborate sulla base dei dati contenuti nei file di log, che riguardano sia le misure acquisite, sia informazioni riguardanti l'utilizzo della pagina web, come il numero di accessi alla pagina;

4. sezione Amministrazione: consente di gestire le risorse dell'applicazione, ad esempio aggiungere o modificare viste, oggetti web, utenti, o modificare impostazioni come la lingua impiegata dal sistema o i modi di visualizzazione delle pagine;

5. sezione Esci: permette di effettuare il logout dal sistema, operazione che dovrebbe essere effettuata prima di chiudere ka pagina del browser, per consentire all'applicazione di 'accorgersi' del fatto che l'utente ha concluso il suo lavoro, ed identificarlo come non attivo.

Vengono ora chiariti due concetti utili per comprendere la descrizione delle modalità di amministrazione di Pincushion, riportate nella sottosezione successiva.

- Oggetti web: si tratta di immagini, file audio o video, link a pagine web che vengono utilizzati per arricchire la presentazione delle informazioni elaborate dal sistema. Consentono di gestire in modo flessibile le pagine web di Pincushion;

- Viste: sono pagine personalizzate che forniscono informazioni sul sistema monitorato o su altre viste.

2.2.1 Amministrazione di viste e oggetti web

In questa sottosezione vengono riportate alcune istruzioni di base per la gestione delle viste e degli oggetti web di Pincushion. Il presente lavoro di tesi non è finalizzato ad approfondire maggiormente questi aspetti, e rimanda il lettore interessato al documento di riferimento[1], Appendice A.

Figura 2.5: Alcune schermate visualizzate nella procedura di creazione di una nuova vista

(a) Elenco delle viste.

(b) Elenco degli elementi che possono essere inseriti

Creare una vista Per creare una nuova vista è necessario seguire il percorso Amministrazione>Strumenti>Viste, che porterà ad una pagina contenente le voci 'Gruppi' e 'Viste'. La prima serve a gestire raggruppamenti di viste, al fine di ottenere una presentazione più efficace dei dati, ed il suo impiego non viene trattato in questo documento. Selezionando 'Viste' si accede ad un menù che visualizza

l'elenco delle viste attive, mettendo in evidenza anche i gruppi, come illustrato in Fig. 2.5a.

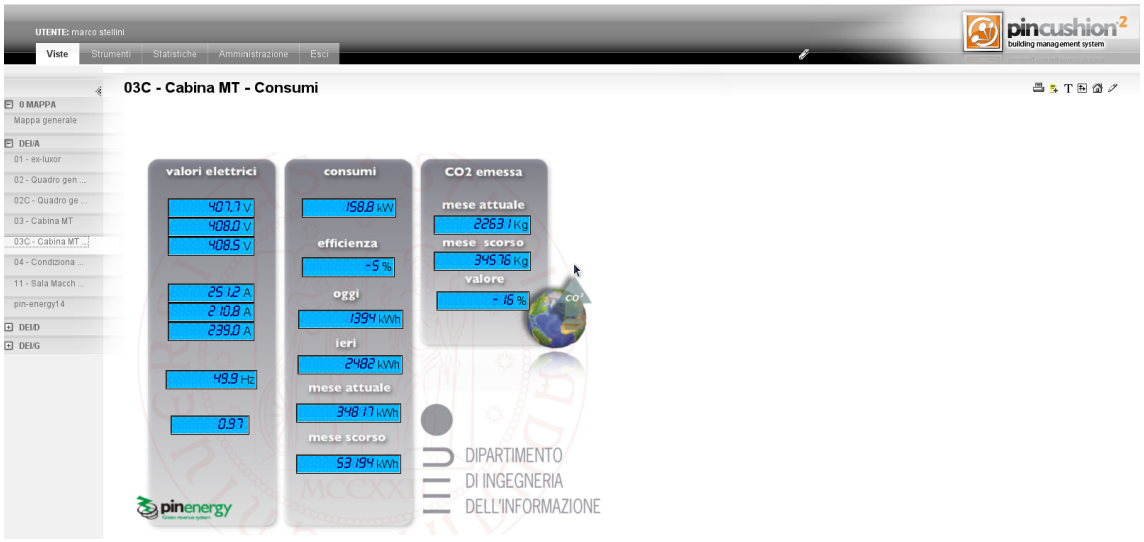
Subito sopra l'elenco, si trovano i comandi 'Nuova vista', 'Duplica viste' e 'Autorizzazioni'. Selezionando il primo, appare una schermata nella quale viene chiesto di inserire il nome della nuova vista ed una sua descrizione sommaria, e che permette di impostare alcune opzioni avanzate.

Terminati questi passaggi, viene offerta la possibilità di aggregare la nuova vista ad uno dei gruppi esistenti, ed infine si passa alla fase in cui è reso possibile inserire nella vista diversi elementi: ad esempio, oggetti web precedentemente caricati, oppure gli stati degli ingressi di uno o più moduli iPin-Energy. Questa operazione viene realizzata attraverso un menù che raggruppa gli elementi in una struttura ad albero, nella quale ogni voce può essere 'esplosa', visualizzando una serie di voci figlie, come mostrato in Fig. 2.5b. In fondo alla pagina, è presente una lista denominata 'Ordine di visualizzazione', comprendente tutti gli elementi inseriti, che regola la gerarchia di visualizzazione degli stessi: gli elementi in cima alla lista vengono posti in secondo piano rispetto a quelli posizionati, via via, in basso. Pertanto conviene spostare le immagini di sfondo in alto, per evitare che esse vadano a coprire altri elementi. Questo stessa schermata è di nuovo accessibile, terminata la procedura di creazione, dal menù 'Viste' premendo il bottone 'Modifica vista', a lato di ogni voce.

L'ultimo passaggio da compiere, per rendere visibile la nuova vista, consiste nel tornare alla pagina 'Viste', sotto Amministrazione>Strumenti>Viste, e selezionare il comando 'Autorizzazioni' precedentemente menzionato: esso consente di stabilire in che misura gli utenti potranno interagire con le viste, ed eventualmente modificarle.

Gestire l'aspetto delle viste In questo paragrafo si descrivono le operazioni da effettuare per gestire l'aspetto di una vista, una volta inseriti gli elementi necessari.

Figura 2.6: La pagina 'Viste' di Pincushion.



Dalla pagina principale, selezionando il comando 'Viste', si accede al menù che visualizza tutte le viste attive per l'utente che ha effettuato il login, la cui schermata

è riportata in Fig. 2.6. In alto a destra è presente una serie di icone che consentono di personalizzare l'aspetto di ciascuna vista. In particolare, il comando 'Visualizza la vista in modalità struttura oggetti', permette di passare ad una modalità in cui è possibile spostare i vari elementi semplicemente trascinandoli, in modo da disporli nel modo più congeniale alla funzione della vista. Terminate le modifiche, l'icona 'Salva la posizione degli oggetti' permette di rendere effettive le modifiche apportate.

Caricare nuovi oggetti web L'inserimento di oggetti web nel sistema Pincushion si esegue dalla schermata che appare seguendo il percorso Amministrazione>Strumenti>Oggetti web, la quale mostra un elenco riportante il nome ed il tipo degli oggetti precedentemente caricati. Subito sopra l'elenco, si trova il comando 'Nuovo oggetto web', che apre una pagina nella quale si possono specificare il nome, la descrizione ed il tipo del nuovo oggetto.

Gli oggetti che il sistema può gestire, come precisato in precedenza, sono

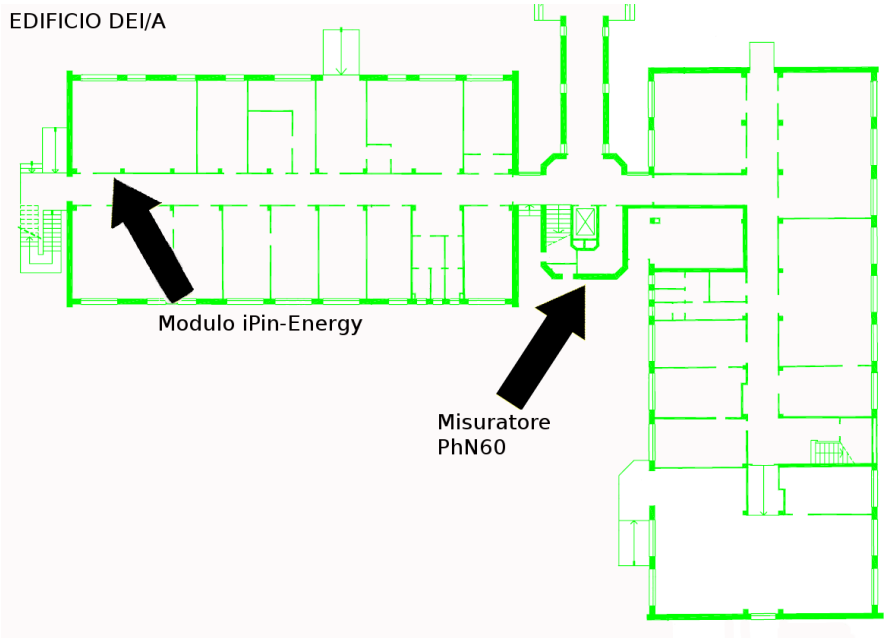
- immagini, file video o file audio, nel qual caso il campo 'Tipo' specificherà il formato del file;
- collegamenti web, nel qual caso si può specificare se aprire il link automaticamente all'ingresso nella vista o se visualizzarlo come testo o pulsante.

Capitolo 3

Telelettura del misuratore di gas

Come anticipato nel precedente capitolo, l'attuale sistema di monitoraggio energetico prevede l'acquisizione dei consumi di potenza provenienti da diversi punti del diparitimento. Uno degli obiettivi del presente lavoro di tesi è estendere il monitoraggio effettuato dall'architettura PinEnergy anche al consumo di gas. Nel presente capitolo verrà dunque presentata l'attività di progettazione e sviluppo necessaria all'implementazione di un sistema di comunicazione che permetta di trasmettere al database di Pincushion i dati relativi al consumo di gas. L'idea di base è stabilire un collegamento tra il contatore, sito all'esterno del dipartimento, ed il più vicino modulo iPin-Energy che si trova a circa 30 m di distanza, come schematicamete illustrato in Fig. 3.1.

Figura 3.1: Ubicazione del misuratore e del modulo iPin nell'edificio DEI/A



Concretamente, il primo problema da affrontare è stato rappresentato dall'impossibilità di stabilire una connessione in cavo, per motivi legati ai costi del materiale

e della manodopera che l'installazione comporta, alle autorizzazioni necessarie ed ai conseguenti tempi. La soluzione più immediata a questo genere di difficoltà è l'utilizzo di una qualche tecnologia wireless, data l'ampia panoramica di dispositivi e protocolli disponibili attualmente sul mercato. L'installazione di una Low Rate - Wireless Personal Area Network (LR-WPAN) è parsa una soluzione adatta a raggiungere l'obiettivo prefissato: si tratta di una rete a basso costo che si caratterizza per la facilità di installazione, l'utilizzo di bande senza necessità di licenza e l'affidabilità del trasferimento dei dati su brevi distanze, utilizzando dispositivi a basso consumo (per esempio alimentati a batteria). La rete, in questo caso, risulta costituita da due nodi: uno che si occupa di trasmettere la lettura del misuratore, situato in prossimità di quest'ultimo, ed uno che si occupa della ricezione e trasferimento al modulo iPin-Energy. Le WPAN a basso rate di trasmissione sono regolamentate dallo standard IEEE 802.15.4.

Per l'implementazione del trasmettitore e del ricevitore sono state studiate e sperimentate due soluzioni:

- la prima prevede l'utilizzo del modulo wireless Jennic JN5148-001-M00 [10], che integra al suo interno un microcontrollore ed un modulo RF. Questa soluzione viene descritta dettagliatamente nella sezione 3.3;
- la seconda si avvale invece di un microcontrollore PIC [16] e del modulo radio MRF24J40MC [15] di Microchip. Questa soluzione viene descritta dettagliatamente nella sezione 3.4.

3.1 Misuratore PhN60

Il contatore in questione è un Sacofgas PhN60. Si tratta di un contatore volumetrico, che effettua la misura isolando fisicamente un volume costante e noto di gas ad ogni ciclo (volume ciclico), sfruttando due appositi contenitori detti camere di misura. La misurazione viene effettuata riempiendo e svuotando ciclicamente le due camere e conteggiando il numero di ripetizioni di tale operazione.

Il PhN60 è equipaggiato con un magnete sull'orologeria, che consente la lettura a distanza attraverso gli impulsi generati dalla commutazione di un relè reed. Sacofgas fornisce un apposito cavo da collegare al contatore, che permette di alimentare il sistema con una tensione fino a 30V, e di ''leggere", attraverso un circuito esterno, gli impulsi generati. La Fig. 3.2 illustra il misuratore.

3.2 Protocollo IEEE 802.15.4

Prima di procedere alla descrizione dell'implementazione hardware, si riportano alcune nozioni di base sul protocollo che regolamenta le LR-WPAN. Lo standard 802.15.4 è stato creato per riempire un vuoto lasciato da altri standard relativi a reti wireless, come IEEE 802.15.1 (Bluetooth), adatto a reti wireless low-rate ma limitato a comunicazioni punto-punto, o IEEE 802.15.3 (High-rate WPAN), poco adatto in molte applicazioni per gli eccessivi consumi di potenza e data-rate.Alcune delle caratteristiche di una LR-WPAN sono:

Figura 3.2: Il misuratore PhN60

1. Bit rate fino a 250kbit/s;
2. Topologie di rete a stella o peer-to-peer;
3. Disponibilità di indirizzi brevi a 16 bit o estesi a 64 bit;
4. Accesso al mezzo fisico di tipo Carrier Sense Multiple Access with Collision Avoidance (CSMA-CA);
5. Utilizzo di tecniche di acknowledgement per migliorare l'affidabilità;
6. Basso consumo di potenza;
7. Funzionalità di Energy detection (ED) del canale;
8. Capacità di indicazione della qualità del collegamento (Link Quality Indication, LQI);
9. Utilizzo di 16 canali nella banda a 2450MHz, 10 canali nella banda a 915MHz, un canale nella banda ad 868MHz.

Il **basso consumo di potenza** è una caratteristica particolarmente importante, perché consente di utilizzare dispositivi alimentati a batteria, eliminando completamente l'esigenza di cablaggi. Tipici esempi d'impiego delle LR-WPAN sono:

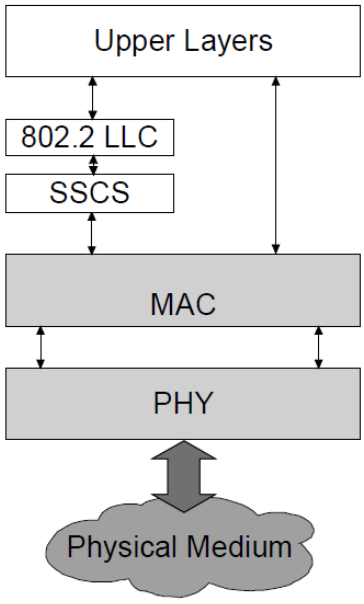
- Domotica: tali reti possono rappresentare una soluzione a basso costo per applicazioni di controllo del riscaldamento, degli impianti di condizionamento, delle luci, delle serrature e di sistemi di intrattenitmento;
- Bioingegneria: sensori e dispositivi diagnostici possono essere collegati per mezzo di una WPAN adibita ad applicazioni medico-sanitarie;
- Monitoraggio del funzionamento di veicoli: le automobili, per esempio, contengono una varietà di sensori e dispositivi che devono poter scambiare informazioni tra di loro, e questo rappresenta un'ambito ideale per le WPAN.

Un esempio notevole è dato dai sensori di pressione degli pneumatici, che non possono, evidentemente, usufruire di collegamenti in cavo;

- Agricoltura: le Wireless PAN sono utili per monitorare le condizioni ambientali e del territorio al fine di massimizzare i raccolti ed ottimizzare i rendimenti.

Architettura Lo studio delle architetture di comunicazione porta a stabilire una suddivisione logica delle operazioni da svolgere in livelli, ognuno dei quali si occupa di gestire specifici aspetti della comunicazione. Il modello OSI (Open System Interconnection) è stato sviluppato dall'ISO allo scopo di fornire una struttura che agevoli lo sviluppo degli standard relativi ai sistemi di comunicazione, e prevede una suddivisione in sette livelli: il più basso, denominato physical layer, si occupa degli aspetti relativi alla trasmissione sul mezzo fisico, mentre procedendo attraverso i livelli superiori si incontrano problematiche, via via, più astratte, come la codifica dei messaggi, il *routing* dei pacchetti attraverso la rete, o ancora la sicurezza della comunicazione, fino ad arrivare al settimo livello, denominato application layer, che si occupa dell'interfaccia con l'utente. In questa sede ci si occuperà soltanto dei due livelli più bassi, ovvero il physical layer ed il data link layer, in quanto sono quelli più interessati da questo tipo di applicazioni. Per approfondimenti consultare il testo di riferimento[18].

Figura 3.3: Livelli del modello OSI utilizzati dal protocollo



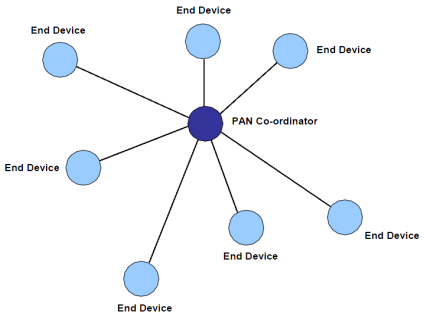
Nello specifico, gli standard IEEE 802.x suddividono il livello data link in due sottolivelli, Medium Access Control (MAC), che si occupa delle modalità di accesso al mezzo fisico, e Logical Link Control (LLC, specificato dallo standard IEEE 802.2), che consente il dialogo con i livelli superiori. Lo standard IEEE 802.15.4 specifica i

livelli physical e MAC, e può sfruttare LLC o interfacciarsi direttamente con i livelli superiori, come illustrato in Fig. 3.3.

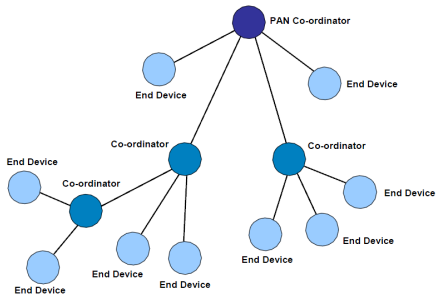
Il livello fisico di IEEE 802.15.4 si occupa di gestire il modulo radio, realizzare le funzioni denominate Energy Detection e Link Quality Indication al fine di selezionare il canale più adatto alla trasmissione, trasmettere e ricevere pacchetti attraverso il mezzo fisico. Il livello MAC, invece, è responsabile delle funzioni di accesso al canale, gestione dei frame, del meccanismo di acknowledge, e della associazione/-disassociazione delle stazioni alla rete. La comunicazione nelle reti IEEE 802.15.4 si basa su un sistema di data frames e MAC command frames, con acknowledge opzionali.

Topologie Nelle reti con topologia a stella (Fig. 3.4a) la comunicazione avviene tra alcuni nodi terminali ed un nodo centrale che svolge il ruolo di *PAN coordinator*. Nelle reti peer-to-peer, invece, vi è solo un coordinatore centrale (il PAN coordinator), ma, almeno in linea di principio, gli altri nodi possono comunicare tra loro senza interessarlo. Questo può dare origine a reti con topologia ad albero (Fig. 3.4b) o ‘’mesh” (Fig. 3.4c). Nelle prime, ciascun nodo ha un padre (tranne la radice, che è il PAN coordinator), può avere dei figli, e comunica soltanto con padre e figli. Nelle reti mesh, invece, tutti i nodi sono identici (con l’eccezione del PAN coordinator), e sono distribuiti secondo una configurazione ad hoc.

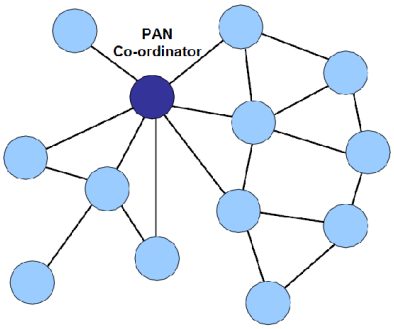
Figura 3.4: Topologie supportate



(a) Topologia a stella



(b) Topologia ad albero



(c) Topologia mesh

Dispositivi In una LR-WPAN troviamo due tipi di dispositivi: full-function device (FFD) e reduced-function device (RFD). Un FFD può comunicare sia con gli RFD che con altri FFD, mentre un RFD può scambiare informazioni soltanto con degli FFD. Evidentemente, i reduced-function devices sono stati pensati come dei dispositivi molto semplici, che non hanno bisogno di scambiare elevate quantità di dati e che possono essere realizzati utilizzando risorse modeste (possono essere per esempio dei sensori). Oltre a questa distinzione, i nodi di una rete IEEE 802.15.4 possono assumere tre ruoli:

- PAN Coordinator: Ha lo scopo di assegnare un ID alla rete, trovare il canale più adatto alla trasmissione, gestire l'aggiunta di nuovi nodi alla rete, trasmettere messaggi da un nodo all'altro (non in tutte le topologie);
- Coordinator: questo tipo di nodo è presente per esempio nelle reti con topologia ad albero. Può essercene più di uno, e in questo caso ciascuno dei coordinatori gestisce i propri nodi figli, amministrando l'ingresso di nuove stazioni nella rete e trasmettendo dati da un nodo all'altro;
- Device (si parla comunemente di *End Device*, anche se questa denominazione non fa parte dello standard): si tratta di un nodo terminale con funzionalità di input/output, ma non di coordinamento.

I dispistivi FFD possono assumere il ruolo di coordinatori o dispositivi terminali, mentre gli RFD non possono assumere funzioni di coordinamento, quindi sono necessariamente degli End Device. La rete deve avere uno e soltanto un PAN coordinator. In alcuni casi un solo dispositivo può assumere tale ruolo, perché è stato determinato precedentemente, o perché è l'unico FFD della rete. Negli altri casi è necessario, nella fase di setting-up, stabilire una politica in base alla quale scegliere il coordinatore centrale. Una volta fatto ciò, va determinato l'ID della rete, che la distingue da possibili altre LR-WPAN localizzate nelle vicinanze. Normalmente, tale operazione viene svolta dal PAN coordinator.

Per quanto riguarda le politiche di ingresso e uscita delle stazioni dalla rete, se un dispositivo intende entrare a far parte della rete, esso deve innanzitutto effettuare un'operazione di *Channel Scan*[1] per individuare un coordinatore. A quest'ultimo il dispositivo invia un *association request*, ed il coordinatore risponde con un acknowledge, accettando o respingendo la richiesta. Diversamente, la richiesta di disassociazione di un dispositivo dalla rete può provenire sia dal dispositivo stesso che dal coordinatore. Quando invece un dispositivo perde la comunicazione con il suo coordinatore, diventa un cosiddetto *orfano*, e dovrà cercare di ristabilire la comunicazione effettuando un'apposita operazione denominata *Orphan Channel Scan*.

Infine, va precisato che ciascun dispositivo in una rete IEEE 802.15.4 può avere due tipi di indirizzo:

- un IEEE MAC address di 64 bit, che lo identifica univocamente nel mondo;

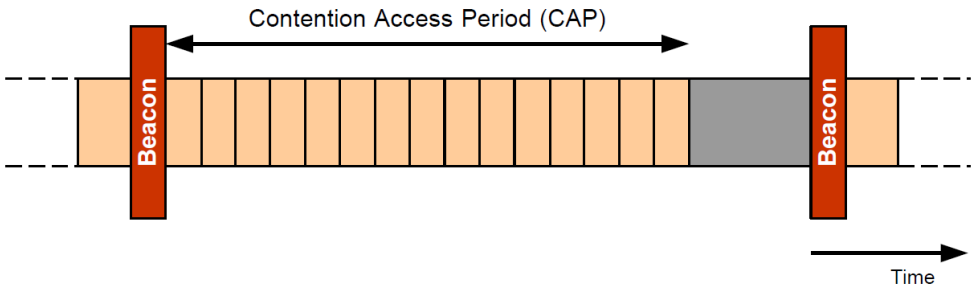
[1] Si distinguono l'Active Channel Scan, in cui il dispositivo invia al coordinatore frame di richiesta di associazione, e il Passive Channel Scan, in cui il dispositivo attende uno dei appositi frame che periodicamente vengono trasmessi dal coordinatore.

- un indirizzo breve a 16 bit, che lo identifica univocamente all'interno della rete (ma non all'esterno: altri dispositivi facenti parte di altre reti potrebbero avere lo stesso indirizzo breve).

Modalità beacon e non-beacon enabled Tutte le reti IEEE 802.15.4 utilizzano dei *beacon*, ovvero frame di controllo che vengono spediti dal coordinatore, ad esempio per consentire l'accesso alla rete di nuovi nodi.

Nella modalità beacon enabled, inoltre, il coordinatore spedisce periodicamente dei beacon contenenti informazioni che consentono alle stazioni di sincronizzare le loro comunicazioni. Normalmente, due beacon successivi marcano l'inizio e la fine di un *superframe*, composto da 16 *timeslots* che i nodi possono utilizzare per comunicare attraverso la rete, utilizzando accesso al mezzo di tipo slotted CSMA/CA. Il tempo totale rappresentato dai timeslots è denominato *Contention Access Period*, come illustrato in Fig. 3.5.

Figura 3.5: Beacon Enabled Mode: Superframe



Nella modalità non-beacon enabled, non avviene trasmissione periodica dei beacon, e le comunicazioni sono asincrone. Questa modalità consente di risparmiare energia, ed è più adatta della precedente alle reti nelle quali ci si aspetta poco traffico.

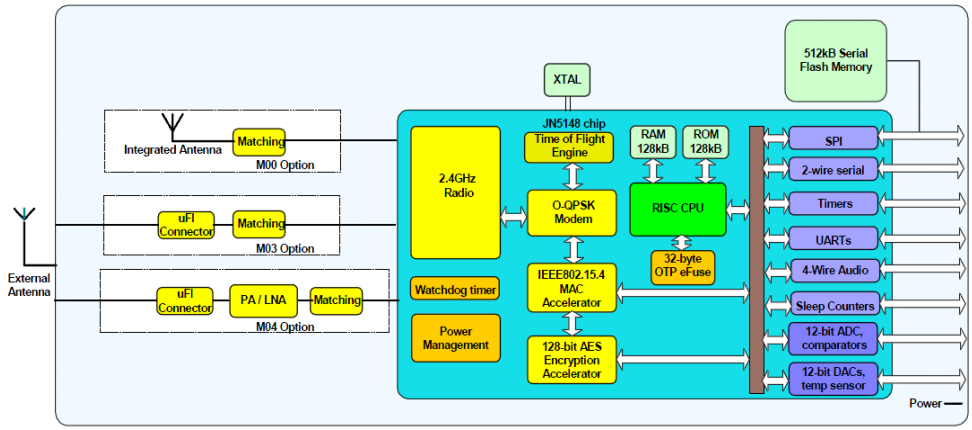
ZigBee Questa denominazione identifica un insieme di protocolli di comunicazione per reti basate sullo standard 802.15.4. Il termine è un marchio registrato di ZigBee Alliance, un gruppo di imprese che si occupa di sviluppare i protocolli e certificare i prodotti che li utilizzano, garantendone l'interoperabilità. Il rapporto esistente tra ZigBee ed IEEE 802.15.4, dunque, è simile a quello che intercorre tra IEEE 802.11 e WiFi Alliance.

3.3 Modulo JN5148-001-M00

Si tratta di un modulo wireless a basso consumo progettato per applicazioni di rete basate su IEEE 802.15.4, ZigBee PRO e JenNet. La caratteristica vincente di questo componente è il fatto che esso integra al suo interno il microcontrollore Jennic JN5148 e la componentistica RF necessaria ad operare in una rete wireless,

limitando al minimo tempi e costi di sviluppo. Il modulo dispone inoltre di una memoria flash, esterna al processore, utilizzata per conservare il software scritto dall'utente, e di un'antenna integrata, con potenza di trasmissione dichiarata di 2.5 dBm. Le Fig. 3.6 illustra lo schema a blocchi del modulo.

Figura 3.6: Modulo JN5148-001-M00: schema a blocchi



Microcontrollore Il cuore del modulo è una CPU di tipo RISC (*Reduced Instruction Set Computer*[2]) a 32 bit. Esso dispone di 15 registri *General Purpose* a 32 bit e di registri *Special Purpose*, utilizzati per gestire le interruzioni e conservare lo stato del processore. Le istruzioni gestiscono dati a 8, 16 o 32 bit, conferendo al processore una certa flessibilità, ma anche la capacità di gestire efficentemente grosse quantità di dati. Il set di istruzioni è progettato per favorire l'implementazione di applicazioni in linguaggi di alto livello, in modo particolare il C, per il quale Jennic mette a disposizione una catena di compilazione ed un debugger. Il sistema di gestione delle interruzioni è vettorizzato internamente, e consente di gestire numerose interruzioni, originate dall'hardware o associate ad operazioni della CPU. Inoltre, per contenere il consumo di energia del dispositivo, il processore è stato dotato di più modalità operative (ad esempio di una modalità "Sleep"), e della possibilità di funzionare a frequenze diverse, da 4 a 32 MHz. Diversamente da altri microcontrollori, il JN5148 dispone di un'architettura di memoria unificata, che raggruppa codice, dati, periferiche e porte di I/O nello stesso spazio di indirizzamento. Per quanto le memorie integrate nel chip, sono presenti:

- una memoria ROM da 128 KB, che include un bootloader per il caricamento nella RAM del codice scritto dall'utente nella memoria flash esterna, la tabella e il gestore delle interruzioni, l'implementazione del MAC IEEE 802.15.4, e le API (*Application Peripheral Interface*) che consentono di gestire in modo semplice le funzioni del protocollo stesso e le periferiche;

[2]Questa espressione identifica architetture dotate di istruzioni semplici, poco numerose e dalla struttura regolare, che vengono tipicamente eseguite in pochi cicli di clock.

- una memoria RAM da 128 KB alla quale il processore è in grado di accedere in un ciclo di clock;
- una memoria OTP (*One Time Programmable*) da 32 KB, che può essere programmata dall'utente.

Tra le periferiche disponibili, vanno evidenziate le seguenti:

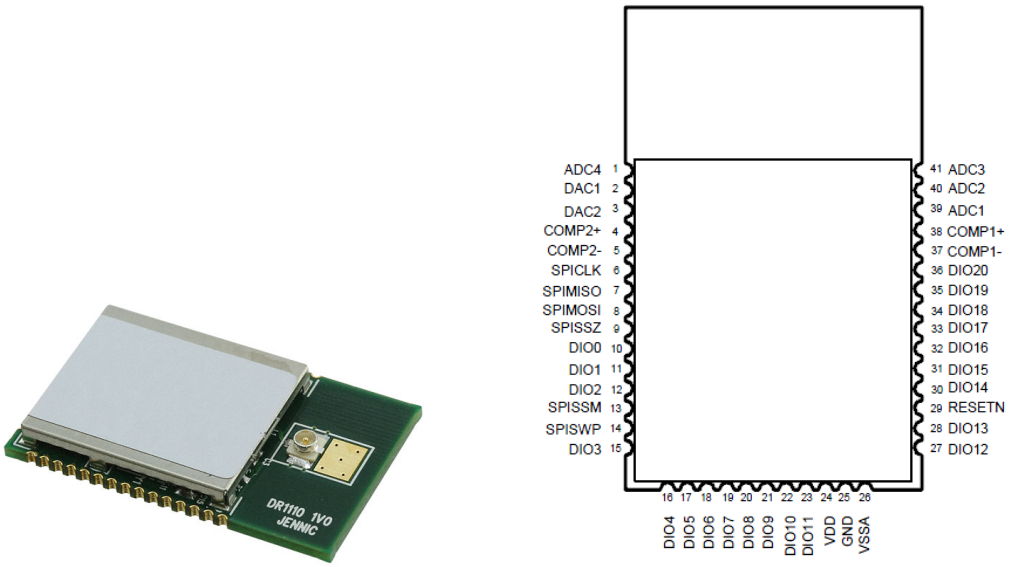
1. Ventuno linee di I/O digitali (multiplexate con altre periferiche, come timer e UART);
2. Un convertitore A/D a 12 bit e quattro canali;
3. Due convertitori D/A a 12 bit;
4. Porte SPI (*Serial Peripheral Interface*) Master e Slave;
5. Tre timer/contatori;
6. Due UART (*Universal Asynchronus Receiver Transmitter*).

Per approfondimenti si faccia riferimento al manuale[9].

3.3.1 Realizzazione delle schede

Il progetto prevede la realizzazione di due board dotate del modulo JN5148: una serve da trasmettitore del segnale fornito dal misuratore, l'altra ricopre il ruolo di ricevitore da interfacciare ad un modulo iPin-Energy. Il trasmettitore ha, inoltre, la funzione di coordinatore della rete, mentre il ricevitore va programmato come un dispositivo terminale.

Figura 3.7: Modulo JN5148-001-M00

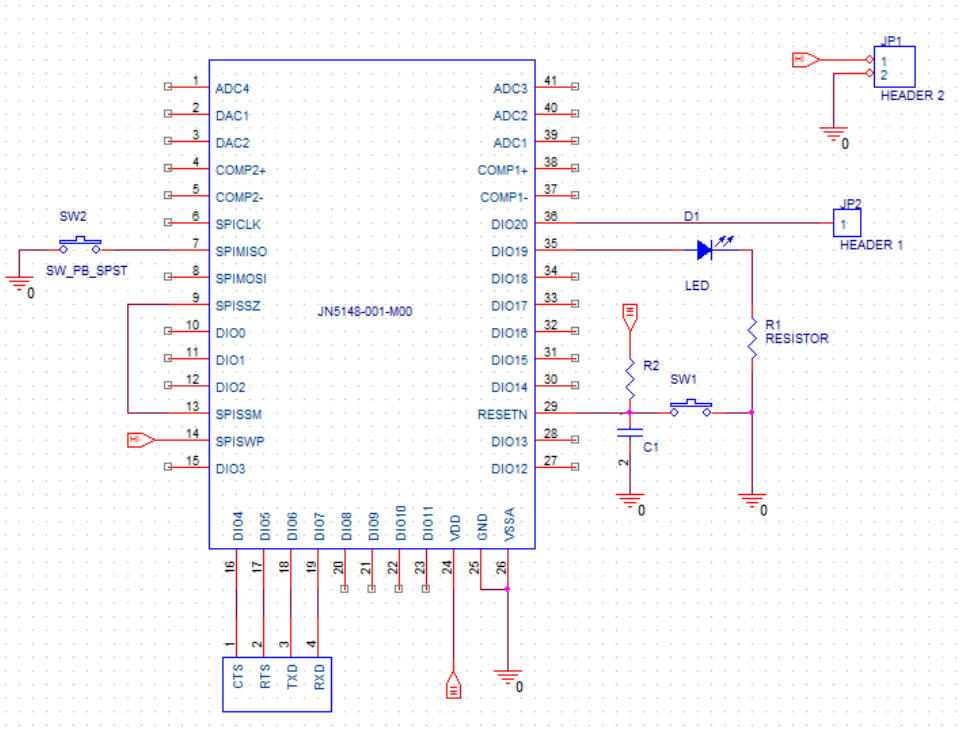


(a) Package

(b) Configurazione pin

Connessioni Nel seguito si riporta una descrizione delle connessioni da realizzare per consentire il corretto funzionamento del modulo. La programmazione può essere effettuata in due modi diversi: utilizzando la porta SPI ed un programmatore esterno, oppure servendosi della porta UART e del Flash Programmer di Jennic. Il progetto prevede la seconda soluzione, per la quale è necessario dell'hardware che consenta il collegamento della scheda ad un personal computer. La Fig.3.7 (b) illustra la configurazione dei pin del modulo. Per effettuare la programmazione del

Figura 3.8: Schema elettrico della board



modulo attraverso la porta UART, in base alle specifiche riportate nel datasheet[10], è **necessario** collegare il pin SPISSZ ad SPISSM, inoltre è consigliabile collegare SPISWP (FLASH Write Protect) a Vdd: questo pin, infatti, inbisce la scrittura della memoria FLASH se interessato da un livello di tensione interpretato come basso. Altri due pin di notevole importanza sono RESETN ed SPIMISO (Master Input Slave Output): al fine di programmare il modulo è infatti necessario che il microcontrollore "veda" un livello di tensione basso sul pin SPIMISO all'accensione o in caso di reset. Il reset avviene quando RESETN viene portato al livello di tensione basso. Collegando questi due pin a GND attraverso gli switch SW1 ed SW2, come illustrato nello schema in Fig. 3.8, la seguente sequenza di operazioni è sufficiente a porre il modulo in modalità di programmazione:

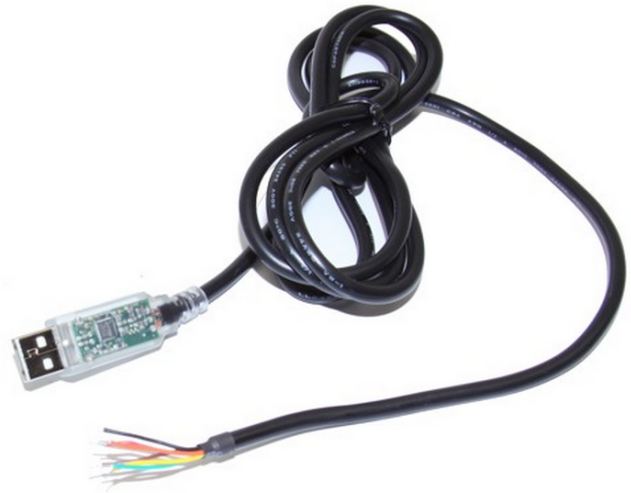
1. Premere (e mantere premuto) SW2;
2. Premere e rilasciare SW1;
3. Rilasciare SW2.

I pin DIO4, DIO5, DIO6 e DIO7 integrano le funzioni di input/output digitali, porta UART e porta JTAG. In questo contesto vengono impiegati come pin CTS (*Clear To Send*), RTS (*Ready To Send*), TXD ed RXD della porta UART, e vanno collegati al relativo connettore. Il pin DIO20 viene impiegato come ingresso per il segnale proveniente dal misuratore (opportunamente adattato), mentre il pin DIO19 pilota un LED con funzioni di debug. Gli altri pin possono essere lasciati flottanti.

La Fig. 3.8 illustra lo schema del circuito da realizzare, disegnato mediante *OrCad Capture CIS*, un software CAD con la funzione di assistere l'utente nella progettazione di circuiti elettronici. Il programma impiega delle librerie di componenti (predefiniti o creati/modificati dall'utente), ciascuno dei quali è identificato, nell'interfaccia grafica, da un blocco cui fanno capo i pin del componente. Capture consente di piazzare i componenti e disegnare le connessioni elettriche, ma non tiene conto delle dimensioni dei componenti o degli effettivi percorsi delle piste. Per questo si utilizzano altri software, come *OrCad Layout*, che, a partire dai *footprint* e dalla *netlist*[3] generata da Capture, consentono di ricavare il disegno della scheda a circuito stampato, completo degli ingombri dei componenti, le geometrie dei pin ed i percorsi delle piste.

Interfaccia con il PC Per consentire la comunicazione del microcontrollore con un personal computer, è necessario aggiungere dell'hardware che faccia da interprete tra la porta UART del JN5148 ed una porta di comunicazione del PC: una scelta agevole è, ad esempio, l'impiego di un cavo della famiglia *USB TTL Serial Cables*, prodotto da FTDI (Fig. 3.9).

Figura 3.9: Un cavo della famiglia FTDI USB TTL Serial Cables



Si tratta di cavi che forniscono la connettività richiesta grazie al chip FTDI FT232R, integrato nel connettore USB. Questo dispositivo fornisce un'ampia gamma

[3]Il *footprint* di un componente rappresenta la disposizione e la geometria delle piazzole che corrisponderanno ai suoi pin, mentre la *netlist* è la lista delle connessioni tra i componenti.

di funzionalità, essendo in grado di gestire rate di trasmissione diversi, con livelli di tensione di 5V/3.3V/2.8V/1.8V, e si adatta ad una ampia gamma di applicazioni: è presente, infatti, nella board integrata all'interno del connettore, una memoria EEPROM che conserva la configurazione del chip. L'estremità del cavo da collegare alla porta UART fornisce i terminali Vcc, Tx, Rx, CTS# ed RTS# (il # indica che queste linee funzionano, per default, in logica negata).

3.3.2 Sviluppo del software

Per lo sviluppo del software Jennic mette a disposizione un SDK (*Software Developer's Kit*) che comprende i seguenti strumenti:

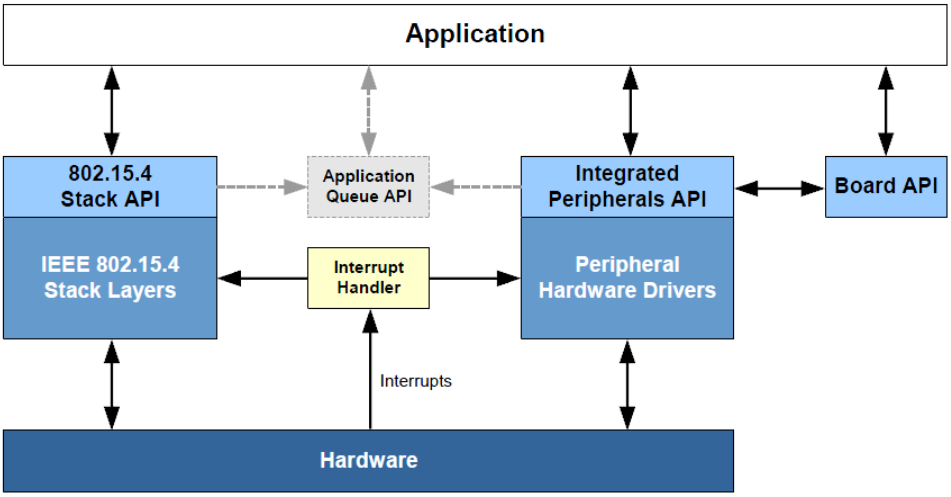
1. Interfaccia a linea di comando Cygwin;
2. Ambiente di sviluppo Eclipse;
3. Strumenti di debug (*standalone* o come plug-in di Eclipse);
4. Catena di compilazione per dispositivi JN51xx;
5. Flash Programmer per dispositivi JN51xx.

Inoltre, vengono fornite delle librerie che facilitano notevolmente lo sviluppo delle applicazioni utente, includendo delle API per l'utilizzo degli stack di protocollo di IEEE 802.15.4, ZigBee PRO e JenNet, e delle periferiche integrate nel modulo. Per ulteriori dettagli, si faccia riferimento alla guida utente[12]. La Fig. 3.10 mette in evidenza il meccanismo delle API ed la loro funzione di intermediazione tra l'hardware e l'applicazione sviluppata dall'utente. Si nota la presenza dell'Application Queue API, il cui utilizzo è opzionale, ma interessante in quanto consente di gestire le interruzioni mediante tre code: la generazione di un interrupt si traduce, cioè, nell'inserimento di una entry nella coda relativa. Grazie a questa caratteristica, l'applicazione può gestire le interruzioni interrogando le code, e dialogando con esse quando necessario.

Infine, ancora nell'ottica di uno sviluppo semplice e rapido del software, viene fornito un'Application Template[4], ovvero del codice che fornisce una base per lo sviluppo di applicazioni basate su IEEE 802.15.4, e che può essere semplicemente modificato per adattarlo alle esigenze del caso. Va sottolineato, però, che **il codice fornito è adatto a reti con modalità non-beacon enabled e con topologia a stella, presuppone che vengano impiegati l'indirizzamento a 16 bit e la comunicazione con acknowledge**. All'interno del archivio .zip scaricabile all'indirizzo citato si trovano le cartelle AN1046_154_Coord e AN1046_154_EndD, le quali contengono il codice sorgente necessario all'implementazione, rispettivamente di un coordinatore o di un dispositivo terminale. Il codice fornito è già completo dal punto di vista del *network setting-up*: entrambi i sorgenti citati dispongono di tutte le funzioni necessarie ad implementare la fase di inizializzazione della rete, che prevede lo svolgimento di una sequenza di operazioni come la creazione del PAN

[4]L'Application Template è disponibile all'indirizzo
http://www.jennic.com/support/application_notes/jn-an-1046_ieee_802154_application_template

Figura 3.10: Ruolo delle API



coordinator, la selezione dell'ID, degli indirizzi, del canale e l'aggiunta di dispositivi alla rete. Nel manuale[11] si trova una descrizione dettagliata dell'Application Template, unitamente alle indicazioni su come modificare il codice.

3.4 PIC18F4620+Modulo MRF24J40MC

Questa sezione illustra l'implementazione del trasmettitore e del ricevitore mediante componenti Microchip. I componenti fondamentali impiegati sono il microcontrollore PIC18F4620 e il modulo radio MRF24J40MC, mentre per la progettazione ed il test del software ci si è avvalsi di due schede di sviluppo PICDEM Z[17].

Microcontrollore Il PIC18F4620 è un microcontrollore con set di istruzioni RISC e dotato di numerose periferiche, come il JN5148 descritto nella sezione precedente, ma che, rispetto a quest'ultimo, presenta notevoli differenze riguardo all'organizzazione interna. Anzitutto l'architettura di memoria è di tipo Harvard, ovvero caratterizzata da memorie distinte per dati e istruzioni. Nello specifico, questo PIC dispone di tre memorie diverse:

- una memoria di programma di tipo FLASH da 64 KB;
- una memoria per dati di tipo SRAM da 3 KB;
- una memoria per dati di tipo EEPROM da 1 KB.

Bus dati e bus istruzioni sono, quindi, separati, il che permette di effettuare accessi contemporanei ai due spazi di memoria, e sono rispettivamente ad 8 e 16 bit. Il set di istruzioni è, anche in questo caso, particolarmente orientato a favorire la compilazione di programmi in linguaggio C. Il dispositivo può funzionare con una frequenza

compresa tra 31 kHz e e 40MHz, e mette a disposizione diverse sorgenti di clock: è presente un oscillatore interno, la cui frequenza è programmabile configurando opportuni registri di controllo, ma è anche possibile fornire il segnale dall'esterno, garantendo la possibilità di utilizzare oscillatori più precisi. E' integrato un circuito PLL (*Phase Locked Loop*) che moltiplica per quattro la frequenza del segnale di clock fornitogli come input. Per quanto riguarda le interruzioni, il PIC18F4620 dispone di numerose sorgenti, che all'occorenza possono essere classificate su due livelli di priorità diversi. Le interruzioni di priorità alta prevalgono su quelle di priorità bassa, e le rispettive routine di servizio sono mappate in modo statico in due zone riservate della memoria. Tra le periferiche di interesse, vanno segnalate le seguenti:

1. Cinque porte di I/O digitali per un totale di 36 ingressi/uscite (nella versione con package a 40 pin). Anche nel caso del PIC, la maggior parte dei pin può essere usato per svolgere compiti diversi;

2. Modulo di conversione A/D a 10 bit, con 13 canali di ingresso;

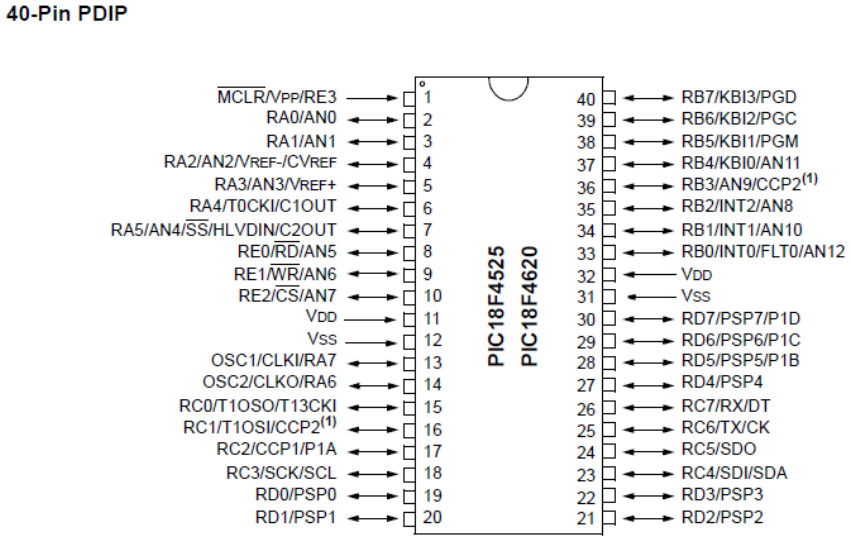
3. Modulo MSSP (*Master Synchronus Serial Port*) che supporta le modalità SPI e I^2C;

4. Quattro timer/contatori ad 8/16 bit;

5. Modulo PWM;

6. Una porta parallela ad 8 bit.

Figura 3.11: Configurazione dei pin del PIC18F4620



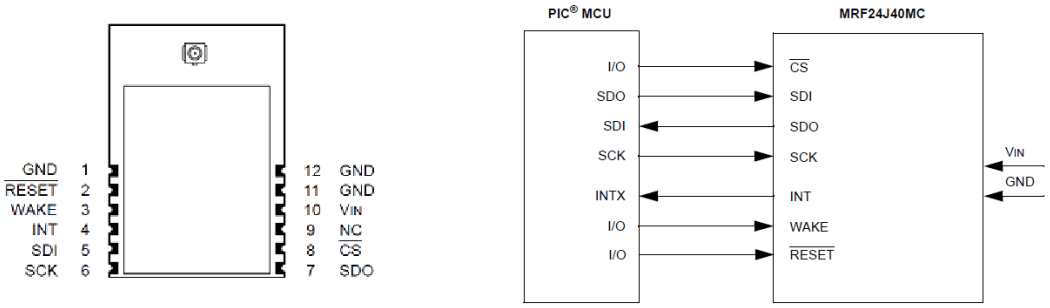
E' il caso di sottolineare una funzionalità molto interessante in questa sede, denominata PORTB On-Change Interrupt: si tratta di un'interruzione che, se abilitata,

interviene in presenza di ogni transizione dal livello di tensione basso a quello alto su uno dei pin RB4, RB5, RB6 o RB7. Questa caratteristica potrebbe essere sfruttata, ad esempio, per l'acquisizione del segnale prodotto dal contatore, al posto di un'alternativa gestione con polling (monitoraggio attraverso la continua lettura dell'ingresso).

Il PIC18F4620 è disponibile con package differenti: quello utilizzato nel caso in esame è la versione a 40 pin, il cui layout è illustrato in Fig. 3.11. Maggiori dettagli sono disponibili nel datasheet del dispositivo[16].

Modulo MRF24J40MC Il dispositvo in questione è un ricetrasmettitore SMD (*Surface Mounting Device*) a radio frequenza prodotto da Microchip e compatibile con lo standard IEEE 802.15.4. E' dotato di un connettore per antenna esterna da 50 ohm, di un oscillatore a cristallo integrato, e si interfaccia ai più comuni microcontrollori PIC attraverso una porta SPI a 4 pin, una linea di interruzione, una linea di wake-up e una di reset. La Fig. 3.12 illustra il modulo e le connessioni necessarie a garantirne il funzionamento. Una descrizione più completa è disponibile nel datasheet del dispositivo[15]. Il cuore del modulo è il circuito integrato MRF24J40,

Figura 3.12: Layout e connessioni del modulo ad un PIC



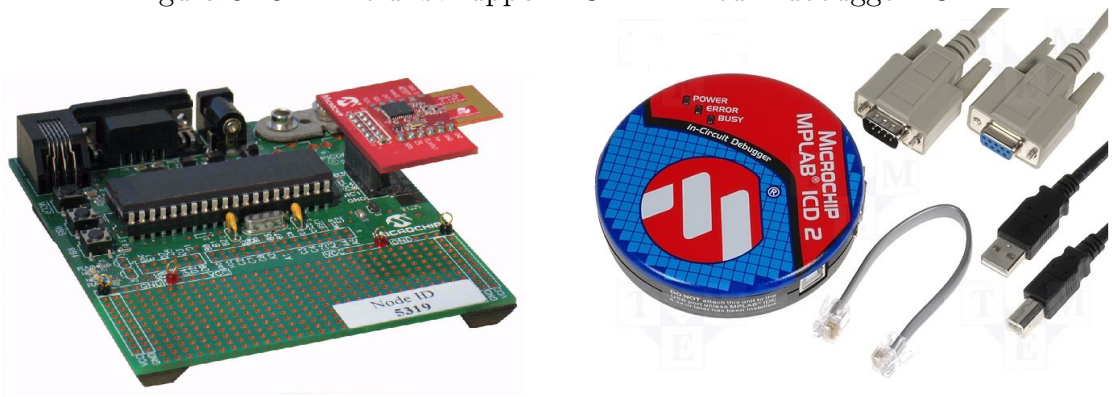
che implementa le funzionalità dei layer MAC e physical dello standard 802.15.4, fornendo il supporto hardware necessario ad effettuare operazioni come l'*energy detection* ed il *sensing* del canale (si ricorda che il protocollo prevede accesso al mezzo di tipo CSMA-CA, il che richiede la capacità di "ascoltare" il canale), o la trasmissione con modalità beacon-enabled. La sua presenza riduce notevolmente il carico di lavoro che il processore deve svolgere, il quale si riduce alla configurazione del modulo e lettura/scrittura dei dati da ricevere/trasmettere, nelle apposite aree di memoria. L'MRF24J40 è compatibile con gli stack di ZigBee e MiWi, un protocollo wireless proprietario basato sullo standard 802.15.4, e sviluppato da Microchip per i processori PIC e dsPIC. Si evidenzia che la potenza di trasmissione massima a radio frequenza è di 19 dBm, qualità che assicura alle trasmissioni su brevi distanze una certa robustezza.

3.4.1 Implementazione su scheda di sviluppo

Nelle fasi preliminari della sperimentazione è stato impiegato il kit di sviluppo PICDEM Z, illustrato in Fig. 3.13a, che comprende:

- una mainboard sulla quale sono montati un microcontrollore PIC18F4620, una porta seriale per la comunicazione con un PC, un connettore RJ-11 che consente il collegamento al debugger, ed altri componenti utili nella fase di sperimentazione;
- una *Wireless Daughter Board* sulla quale è montato il modulo radio MRF24J40, ed un connettore adatto all'installazione sulla mainboard;
- i cavi necessari al collegamento con il PC.

Figura 3.13: Il kit di sviluppo PICDEM Z ed il debugger ICD2



(a) La scheda di sviluppo PICDEM Z

(b) Il debugger ICD2

Per il trasferimento del codice nella memoria del processore è stato impiegato lo strumento di programmazione e debug MPLAB ICD2 (illustrato in Fig. 3.13b), che dispone di un'interfaccia USB/seriale per la comunicazione con il calcolatore, e di una porta RJ-11 per il collegamento alla scheda. Nell'Appendice A sono disponibili le istruzioni per l'installazione dei driver del debugger ICD2. L'ambiente di sviluppo che viene, tipicamente, impiegato quando si ha a che fare con i microncontrollori PIC è l'IDE (Integrated Development Environment) MPLAB[14] di Microchip (nel caso in esame, la versione 8.40), che fornisce una varietà di strumenti utili per la programmazione di questi processori, oltre ad essere particolarmente ottimizzato per la loro gestione. Sono da evidenziare, ad esempio, utility come l'analizzatore logico, che consente di simulare l'esecuzione del programma osservando l'andamento temporale dei segnali sui singoli pin, o l'applicazione *Watch* che permette di valutare il contenuto delle variabili durante l'esecuzione. Altre funzionalità utili sono la possibilità di eseguire il codice istruzione per istruzione, di visualizzare il codice tradotto in istruzioni assembly, mediante il comando *Disassembling List*, di impostare opzioni di compilazione e in generale di determinare in modo dettagliato le operazioni svolte dal compilatore. L'IDE va configurato specificando la catena di compilazione da utilizzare nella costruzione del codice, oltre all'esatta sigla del processore di cui

si dispone. In questo contesto si è fatto uso di una suite messa a disposizione da Microchip, il compilatore C-18[5], progettato specificamente per la famiglia di microcontrollori PIC18, ma è possibile utilizzare anche altri compilatori. Si sottolinea che la suite non è compresa nell'ambiente di sviluppo, ma va installata separatamente. Le istruzioni relative sono disponibili nell'Appendice A.

Per quanto riguarda lo sviluppo del codice, sono state sfruttate le librerie di MiWi disponibili sul sito di Microchip (incluse nel pacchetto Microchip Application Libraries), dopo averle opportunamente modificate e semplificate, al fine di adattarle all'applicazione in esame. Nel dettaglio, per semplificare lo sviluppo delle applicazioni che utilizzano MiWi, sono resi disponibili due layer di interfaccia:

- MiApp: dialoga con l'applicazione utente e con il protocollo di comunicazione proprietario, ad esempio MiWi P2P o MiWi Mesh. Consente di utilizzare protocolli diversi senza dover modificare l'applicazione;

- MiMAC: fornisce le funzionalità del MAC layer, ma è sviluppato appositamente per gestire i transceiver di Microchip.

Data la semplicità dell'applicazione in esame, si è scelto di implementare una prima versione del software utilizzando direttamente le funzioni messe a disposizione da MiMAC, senza cioè fare uso dei protocolli proprietari Microchip: sostanzialmente, sono state implementate solo alcune delle funzionalità previste da IEEE 802.15.4, al fine di ottenere una rete-prototipo funzionante, sulla cui base è possibile sviluppare soluzioni più complete e performanti. Il dispositivo da collegare al misuratore PhN60 è stato programmato in modo da agire implicitamente da PAN Coordinator, e occuparsi solamente della trasmissione di pacchetti che segnalano il completamento di un ciclo di misura da parte del contatore, mentre non è stato previsto che possa ricevere pacchetti dal dispositivo collegato al modulo iPin-Energy. Il PIC18 montato su questa scheda, nella fase di set-up della rete, inizializza il modulo radio al fine di selezionare un canale di comunicazione, un ID per la rete, ed un indirizzo a 16 bit predeterminati, così come il microcontrollore montato sulla scheda di ricezione. Quest'ultima, allo stesso modo, è stata programmata al fine di non trasmettere alcun pacchetto, ma rimanere costantemente in attesa dei frame in arrivo dal misuratore. Nell'Appendice B viene commentato un estratto del codice sorgente scritto per implementare questa applicazione: riassumendo, entrambi i programmi (sia quello per il trasmettitore che quello per il ricevitore) contengono una funzione *main*, eseguita dal processore all'accensione o in caso di reset, all'interno della quale vengono chiamate una serie di funzioni di inizializzazione, come *BoardInit* e *MRF24J40Init*, che impostano i registri di controllo del PIC e del modulo radio in modo da ottenere il funzionamento desiderato. Dopo queste funzioni preliminari, i microcontrollori passano all'esecuzione di un ciclo infinito, all'interno del quale il trasmettitore chiama la funzione *SendPacket* solo se il pin connesso al misuratore si trova al livello logico alto, mentre il ricevitore interroga il modulo radio, e, in caso di ricezione di nuovi pacchetti, procede alla lettura degli stessi.

[5]E' stata impiegata la versione 3.36 Lite del compilatore C-18, che pur essendo freeware è sufficiente per la compilazione dell'applicativo in questione.

3.5 Conclusioni

Le due soluzioni sperimentali presentate nei precedenti paragrafi sono di fatto equivalenti. Si è scelto tuttavia di sviluppare ed applicare la configurazione basata su componenti Microchip. Si tratta di una soluzione in linea di principio più complessa di quella basata sul modulo Jennic, perchè richiede l'utilizzo di almeno due componenti, anzichè uno soltanto. Questa scelta è stata motivata da ragioni legate alla praticità nella realizzazione delle due schede:

- dal punto di vista della realizzazione fisica, il PIC ed il modulo radio sono dotati di terminali più comodi da installare sulla scheda, rispetto al modulo Jennic: il PIC può essere comodamente inserito in un apposito zoccolo, mentre il modulo radio ha un package SMD, ma con pin più grandi e distanziati rispetto a quelli del modulo Jennic;

- dal punto di vista dell'implementazione del software, i vantaggi principali della seconda soluzione sono stati la familiarità con l'ambiente di sviluppo MPLAB, e con la programmazione dei PIC in generale, data l'ampia diffusione di questi microcontrollori, e la disponibilità presso il Dipartimento delle schede di sviluppo PICDEM Z e del debugger ICD2, che hanno reso il procedimento molto più comodo.

Capitolo 4

Meteo DEI

4.1 Stazione Meteorologica Davis Vantage Vue

Un aspetto del monitoraggio ambientale, benchè non direttamente legato a questioni di efficienza energetica, è sicuramente costituito dal rilievo delle condizioni meteorologiche.

Da questa constatazione è nata l'idea di dotare il Dipartimento di un sistema che consentisse di misurare i parametri meteorologici *in situ*, e che disponesse di un'intefaccia adeguata ad immettere i dati in rete, in modo da fornire un servizio di consultazione informatizzato. Una soluzione semplice e relativamente economica a questa necessità è stata individuata nella stazione meteorologica Vantage Vue di Davis Instruments, uno strumento largamente usato dai meteo-amatori. La stazione meteorologica è composta da due elementi:

- un gruppo sensori integrato (*Integrated Sensor Suite*[6]), che include la strumentazione necessaria ad acquisire le misure sulle condizioni meteorologiche esterne;
- una console[7] che fornisce l'interfaccia verso l'utente, un PC oppure la rete.

I due blocchi comunicano a distanza sfruttando un trasmettitore ed un ricevitore che implementano un protocollo di comunicazione dedicato, in modo che il gruppo sensori possa essere installato all'esterno. Al fine di comunicare correttamente, il gruppo sensori deve essere associato ad un *transmitter ID* univoco (sono disponibili gli ID da 1 a 8), e la console deve essere impostata per ricevere i dati dal trasmettitore associato a quello specifico ID. Questo permette di utilizzare più stazioni e console nella stessa area. Per default all'ISS è assegnato l'ID 1 e la console è configurata di conseguenza.

Gruppo sensori integrato Il gruppo sensori è alimentato da un piccolo pannello fotovoltaico (potenza nominale 0.5 W), e dispone di una batteria al litio da 3V con funzione di backup, che permette di far fronte ad eventuali insufficienze energetiche. Il dispositivo è progettato per funzionare, senza sostituire la batteria, da un minimo di circa 8 mesi (in assenza di alimentazione da parte del pannello) fino ad oltre 2 anni. Esso dispone dei seguenti strumenti di misura:

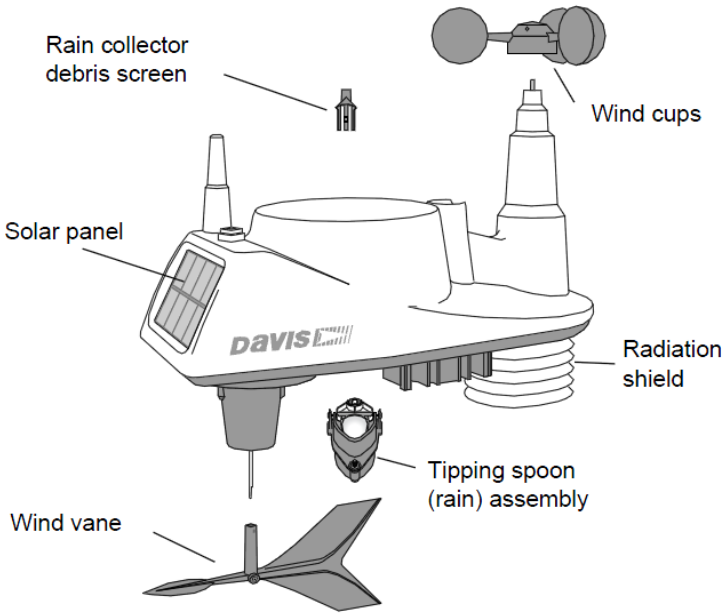
1. un anemometro, ovvero un sensore di velocità del vento, costituito da un perno dotato di pale eoliche, e da uno switch magnetico, che viene fatto commutare attraverso la rotazione del perno stesso. La frequenza di commutazione permette di determinare la velocità di rotazione;

2. un segnavento collegato ad un encoder magnetico, che consente di determinare la direzione del vento;

3. un pluviometro, costituito da un collettore che convoglia l'acqua su di un cucchiaio, il quale, inclinandosi, fornisce un'indicazione sull'intensità delle precipitazioni. E' fornito, inoltre, un filtro per eventuali detriti, che falserebbero le misure;

4. un sensore di temperatura, costituito da un diodo a giunzione PN, ed un sensore di umidità, localizzati all'interno di uno scudo che minimizza l'impatto della radiazione solare sulle misure.

Figura 4.1: Vantage Vue ISS



Le misure vengono effettuate con frequenza costante, dipendente dalla variabile misurata: per esempio, le misure sulla velocità del vento vengono aggiornate ogni 2.5 s, mentre quelle sulla pressione ogni minuto. Uno schema dell'ISS è illustrato in Fig. 4.1. Il gruppo dispone di un *Sensor Interface Module* (SIM), che raccoglie i dati acquisiti dai sensori e li trasmette alla console.

Il gruppo sensori può trasmettere i dati acquisiti a più console, non soltanto Vantage Vue, ma anche Vantage Pro2 e dispositivi Davis Weather Envoys.

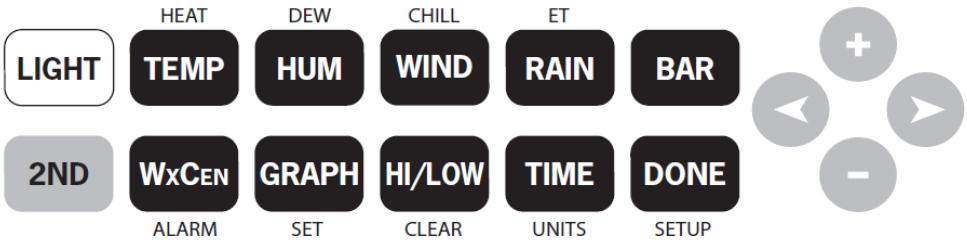
Figura 4.2: Console Vantage Vue

Console Wireless La console, illustrata in Fig. 4.2, dispone di uno schermo LCD e di una tastiera. Essa permette di visualizzare ed immagazzinare i dati ricevuti dal gruppo sensori, di produrre grafici e funzioni di allarme, e può essere collegata, a seconda della versione acquistata, ad un PC (tramite cavo seriale od USB) oppure ad una rete Ethernet. Quest'ultimo è il caso della console presente al DEI, che dispone di un *data logger*: si tratta di un dispositivo che si occupa di raccogliere le misure ricevute in una memoria interna, e che implementa il protocollo Ethernet, disponendo di una porta RJ-45 e di un MAC address. Il data logger è in grado di contenere fino a 2560 record (52 byte per record, la capacità della memoria è quindi approssimativamente di 128 KB), e gli intervalli di archiviazione selezionabili vanno da 1 a 120 minuti. La console funziona attraverso l'apposito alimentatore, ma è prevista la possibilità di sfruttare, anche in questo caso, 3 batterie tampone (durata approssimativa dichiarata di 9 mesi). E' da evidenziare il fatto che questa console è in grado di ricevere dati da più trasmettiori (ognuno identificato da un ID univoco), e non soltato dal gruppo sensori del pacchetto Vantage Vue, ma anche dall'ISS di una stazione meteorologica Davis Vantage Pro2. Inoltre, in presenza di altre console Vantage Vue o Vantage Pro2, è possibile impostare una modalità di "ritrasmissione" nella quale la console reinvia i dati ricevuti alle altre, divenendo essa stessa un trasmettitore, associato ad un ID univoco. In questo modo si realizza una vera e propria rete per il monitoraggio meteorologico. Infine, questo dispositivo dispone di un sensore di temperatura, in modo da poter evidenziare la temperatura interna (relativa al locale nel quale è installata) oltre che quella esterna, misurata dal gruppo sensori.

La tastiera è composta da 12 pulsanti di comando, e quattro di navigazione. Ciascun pulsante di comando è associato ad una variabile meteorologica, o ad un

comando della console, selezionabili direttamente premendo il pulsante. Inoltre, ciascuno di tali pulsanti ha una seconda funzione: per selezionarla è sufficiente premere e rilasciare 2ND, e subito dopo (entro 7-8 secondi) premere il pulsante relativo. I pulsanti di navigazione +, -, < e > sono utilizzati per inserire valori, selezionare opzioni e fornire funzioni addizionali quando usati in combinazione con un pulsante di comando (Fig. 4.3).

Figura 4.3: Tastiera della console Vantage Vue



Ciò che viene visualizzato dal display dipende dalla modalità nella quale si sta operando. A tal proposito, esistono cinque modalità di funzionamento:

- Setup;
- Current Weather;
- High/Low;
- Alarm;
- Graph.

La modalità Setup è prevista per configurare il funzionamento della stazione, ed è richiamata automaticamente quando la console viene accesa. Per entrare nella modalità Setup è necessario premere e rilasciare 2ND e poi premere il tasto DONE (che ha funzione ausiliaria SETUP). Fatto ciò, per spostarsi avanti o indietro tra le schermate visualizzate si utilizzano rispettivamente i tasti DONE e BAR, mentre i pulsanti di comando permettono di effettuare le azioni desiderate. Per uscire dalla modalità di Setup è necessario mantenere premuto DONE fino a quando la console passa alla modalità Current Weather. La modalità Setup è composta da 19 schermate, alle quali si accede sequenzialmente, che consentono di:

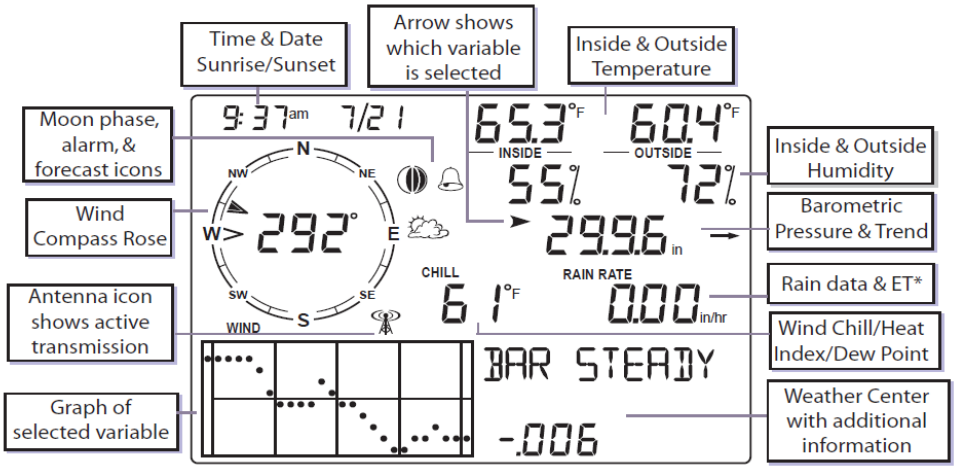
1. Impostare data ed ora visualizzati, fuso orario del luogo d'installazione e configurare la gestione dell'ora legale;
2. Visualizzare e configurare gli ID dei trasmettitori dai quali la stazione sta ricevendo (ci sono 8 ID disponibili), aggiungere e rimuovere trasmettitori;
3. Impostare latitudine, longitudine ed altitudine del luogo d'installazione;
4. Configurare il metodo con cui la pressione barometrica viene determinata e calcolata;

5. Impostare il tipo di pale dell'anemometro montato sull'ISS, e il tipo di cucchiaio del sensore per le precipitazioni;

6. Configurare altri parametri, tra cui l'inizio della stagione delle piogge, la quantità di informazioni aggiuntive visualizzate a commento dei dati meteorologici, il baud rate del collegamento al PC od alla rete;

La modalità Current Weather, la cui schermata è riportata in Fig. 4.4, permette di visualizzare sul display le letture del gruppo sensori. E' possibile impostare le unità di misura e visualizzare fino ad otto variabili meteorologiche, oltre ad ora e data, fase lunare, un'icona che inidica le previsioni meteorologiche per le prossime 12 ore ed un grafico della variabile selezionata. Selezionando una variabile essa viene stampata a video se non era già precedentemente visibile, oppure ne viene visualizzato il grafico. Per selezionare una variabile è sufficiente premere il tasto relativo (preceduto dalla pressione di 2ND, se si tratta di una funzione ausiliaria). Premendo ripetutamente il tasto WxCEN dopo aver selezionato una variabile, vengono visualizzate informazioni aggiuntive relative alla variabile stessa, nel riquadro Weather Center in basso a destra. Le variabili che possono essere visualizzate, come

Figura 4.4: Schermata associata alla modalità Current Weather



evidenziato in Fig. 4.4, sono:

1. Ora e data, orari di alba e tramonto, fase lunare;

2. Velocità e direzione del vento;

3. Temperatura interna ed esterna;

4. Umidità;

5. Pressione barometrica e relativo andamento (calcolato sulla base delle letture nelle ultime tre ore);

6. Fattore di *wind chill* [1];

[1] parametro legato all'influenza del vento sulla temperatura percepita da una persona

7. Punto di rugiada;

8. Indice di calore;

9. Precipitazioni (intensità e totale giornaliero, mensile ed annuale).

Premendo il tasto HI/LOW la console si porta ad operare nella modalità relativa: il display visualizza i massimi giornalieri per tutti i campi disponibili, ed il grafico mostra il massimo corrente e quelli dei 25 giorni precedenti. Premendo i tasti + e - si naviga attraverso le viste dei massimi e minimi giornalieri, mensili ed annuali, mentre i pulsanti < e > permettono di scorrere i 26 valori memorizzati per la variabile correntemente selezionata. Massimi e minimi vengono aggiornati automaticamente al termine di ogni periodo: per esempio, i massimi giornalieri vengono aggiornati ad ogni mezzanotte, quelli mensili alla mezzanotte dell'ultimo giorno del mese.

La stazione meteorologica Vantage Vue permette di programmare 22 condizioni di allarme, ciascuna agganciata al valore di una delle variabili. L'allarme scatta quando la lettura eccede o scende al di sotto della soglia programmata, con le eccezioni dell'allarme relativo alla pressione barometrica, che scatta non sulla base del valore istantaneo, ma del trend nelle ultime ore, e dell'allarme temporale, che scatta alla data ed ora prefissate. Quando sussiste una condizione d'allarme, la console emette un segnale sonoro (per un tempo massimo di due minuti se è alimentata solo dalle batterie), visualizza un'icona lampeggiante ed una descrizione della condizione d'allarme nel campo Weather Center. Al fine di gestire gli allarmi è necessario premere 2ND seguito da WxCEN (che ha funzione ausiliaria ALARM, appunto): si passa quindi alla modalità Alarm, che consente di visualizzare e settare le soglie (premere HI/LOW per passare da soglie superiori a inferiori e viceversa). La funzione SET permette di attivare l'allarme associato ad una determinata variabile, e di fissare la soglia. La pressione del tasto DONE comporta il ritorno alla modalità Current Weather.

La pressione del tasto GRAPH in seguito alla selezione di una variabile comporta il passaggio alla modalità Graph, nella quale vengono evidenziati il valore attuale della variabile, il grafico ed eventuali informazioni aggiuntive pertinenti. Per quanto riguarda il grafico, l'asse orizzontale rappresenta il tempo, ed è suddiviso in 26 intervalli, che rappresentano i 26 campioni memorizzati. Anche in questo caso la funzione HI/LOW consente di passare dalla visualizzazione dei massimi a quella dei minimi, e viceversa, i tasti < e > di passare da un campione all'altro, mentre i tasti + e - hanno l'effetto di modificare l'asse dei tempi (passando dal grafico costruito con i campioni delle ultime 25 ore a quello con gli ultimi 25 giorni o mesi).

Per maggiori informazioni sulla console wireless della stazione meteorologica Vantage Vue, si consulti il manuale[7].

4.2 Trasmissione dei dati tra ISS e console

Gruppo sensori e console comunicano sfruttando una tecnica di modulazione digitale di tipo *Frequency Hopping Spread Spectrum*, che opera nella banda 902-928 MHz per i modelli venduti negli Stati Uniti, 868.0-868.6 MHz per i modelli venduti in Europa, con potenza di trasmissione inferiore a 8 mW. Si tratta di bande

che possono essere utilizzate senza bisogno di licenze particolari o tasse, certificate rispettivamente da FCC (*Federal Communications Commission*, l'agenzia statunitense che si occupa di regolamentare le comunicazioni non governative) e CE. In particolare, la banda 868-869 MHz è specificamente designata per l'utilizzo da parte di dispositivi LPD/SRD (*Low Power Devices/Short Range Devices*) con potenze di trasmissione inferiori a 10 mW. La trasmissione dei pacchetti avviene ogni 2.5 s, e il range di trasmissione va da un minimo di 75 m in presenza di ostacoli, ad un massimo di 300 m in campo libero.

In questo paragrafo si vuole presentare una breve descrizione delle tecniche di modulazione attualmente utilizzate, per poi focalizzare la trattazione sulla tecnica FHSS. Anzitutto, si richiama che l'espressione *modulazione digitale* indica il processo di trasformazione di una sequenza di bit in un segnale continuo adatto alla trasmissione attraverso un canale di comunicazione. Il dispositivo che si occupa di effettuare questa operazione prende il nome di *modulatore digitale*, e deve essere associato ad un *demodulatore* che esegua l'operazione opposta per completare la trasmissione. Concettualmente, il modulatore mappa la sequenza b_l di bit in ingresso in una sequenza di simboli a_k, nella quale ciascun simbolo appartiene ad un alfabeto di M elementi (quindi ogni gruppo di $\log_2$M bit in ingresso viene mappato in un simbolo in uscita). A ciascuno di tali simboli, poi, viene associata in modo biunivoco una forma d'onda s_{a_k}(t), in modo che il segnale trasmesso risulta essere descritto dall'espressione

$$s_{T_x}(t) = \sum_{k=-\infty}^{+\infty} s_{a_k}(t - kT),$$

dove T è il periodo di simbolo, che si può facilmente ricavare da T_b, il periodo di trasmissione dei singoli bit, nel seguente modo:

$$T = T_b \log_2 M.$$

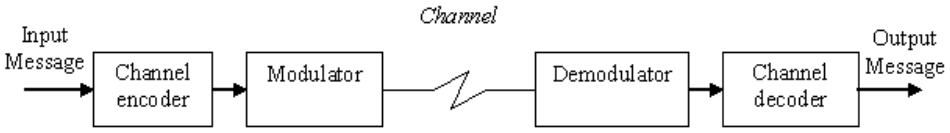
Introducendo il *bit rate* R_b come l'inverso di T_b, si ottiene il *symbol rate*

$$F = \frac{1}{T} = \frac{R_b}{\log_2 M},$$

che si misura in baud. Il canale di comunicazione introduce sia distorsione delle forme d'onda trasmesse, sia *rumore*, il cui effetto può essere approssimato a quello di un segnale che si somma a quello trasmesso, e tutto ciò si traduce in possibili errori nella fase di demodulazione, quindi nell'errata ricostruzione del messaggio originario. Un parametro che consente di caratterizzare la bontà di una tecnica di modulazione digitale è la **probabilità di errore**, sul simbolo oppure sul bit ricostruiti al ricevitore: essa è legata all'energia del segnale trasmesso e alla dimensione M dell'alfabeto utilizzati.

Uno schema di principio è rappresentato in Fig. 4.5, dove sono posti in evidenza anche il *codificatore* ed il *decodificatore* di canale. Il primo si occupa di trasformare il messaggio digitale in ingresso in un secondo messaggio digitale, più robusto nei confronti degli errori che il canale può introdurre, mentre il secondo esegue l'operazione inversa. In questa sede non si ritiene opportuno approfondire questi aspetti, per i quali si consiglia la consultazione del testo[18].

Figura 4.5: Sistema di trasmissione digitale



Nei moderni sistemi di comunicazione vengono utilizzate prevalentemente due tecniche di modulazione digitale:

- OFDM (*Orthogonal Frequency Division Multiplexing*);
- Spead spectrum.

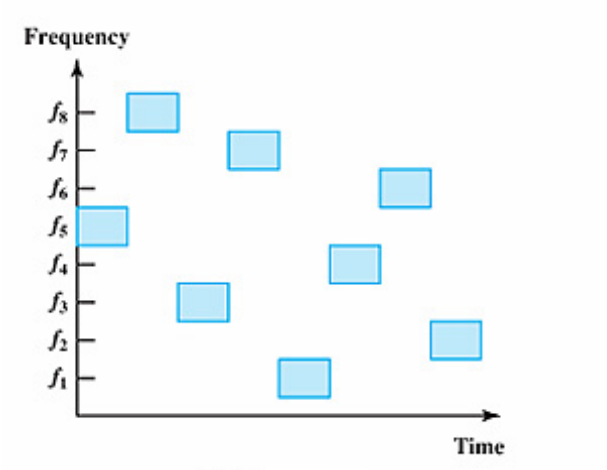
OFDM è una una tecnica di modulazione largamente utilizzata nei sistemi di comunicazione che forniscono bit rate elevati su canali dispersivi; è impiegata, per esempio, nella trasmissione digitale di programmi televisivi (*Digital Video Broadcasting*). Diversamente, la strategia spread spectrum è comunemente usata nelle applicazioni wireless a basso bit rate e bassa potenza di trasmissione: essa rappresenta quindi una scelta naturale per applicazioni come quella in esame.

Una caratteristica peculiare delle modulazioni spread spectrum è l'utilizzo di una banda nettamente più larga rispetto a quella minima richiesta per la trasmissione del segnale. Questo fatto comporta alcuni notevoli vantaggi:

1. **Bassa probabilità d'intercettazione** La larghezza della banda utilizzata e le ridotte potenze in gioco fanno sì che la densità spettrale di potenza assuma valori molto bassi, persino minori di quella del rumore. Questo comporta che la demodulazione del segnale risulti molto complicata se i parametri di modulazione non sono noti al ricevitore;
2. **Robustezza nei confronti di rumore e disturbi**;
3. **Accesso multiplo al canale di comunicazione** Ciò significa che più comunicazioni possono avere luogo contemporaneamente sullo stesso canale;
4. **Capacità di localizzazione** Queste tecniche di modulazione consentono una stima molto accurata dei tempi di arrivo dei simboli modulati, e questo può essere sfruttato per misurare con precisione la distanza tra gli attori della trasmissione.

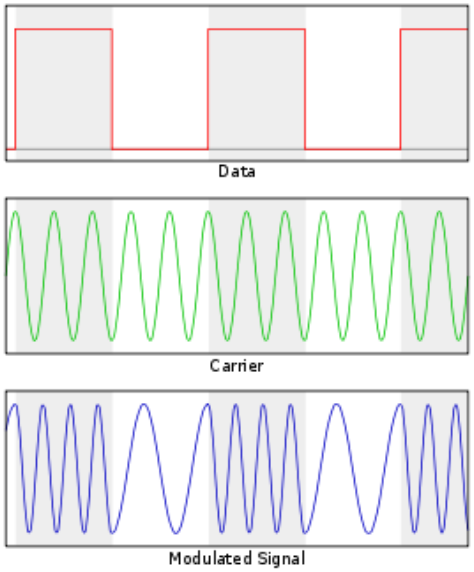
L'idea alla base del frequency hopping, che rientra nella categoria delle modulazioni spread spectrum, consiste nel suddividere la banda disponibile in N fette, con frequenze centrali f_i, $i=1, \ldots, N$, come illustrato in Fig. 4.6. Ad ogni periodo di simbolo kT la sottobanda di trasmissione viene scelta in base ad un codice pseudo-random precedentemente generato $c_{u,\,k}$, dove u identifica l'utente, cosicchè $f_{c_{u,\,k}}$ sia la frequenza centrale scelta. E' possibile trasmettere più simboli per una data frequenza centrale, nel qual caso si parla di FH *lento*; se invece si trasmette

Figura 4.6: Esempio del paradigma frequency hopping con 8 sottobande



un simbolo ad ogni "salto" si parla di FH *veloce*. L'ortogonalità tra le trasmissioni di utenti diversi si ottiene costruendo i codici $c_{u,\,k}$ in modo che la probabilità di trasmettere nello stesso istante e sulla stessa sottobanda sia limitata.

Figura 4.7: Esempio di modulazione FSK con M=2



In associazione al frequency hopping viene spesso utilizzata la modulazione FSK (Frequency Shift Keying) *non coerente*, che utilizza le seguenti forme d'onda

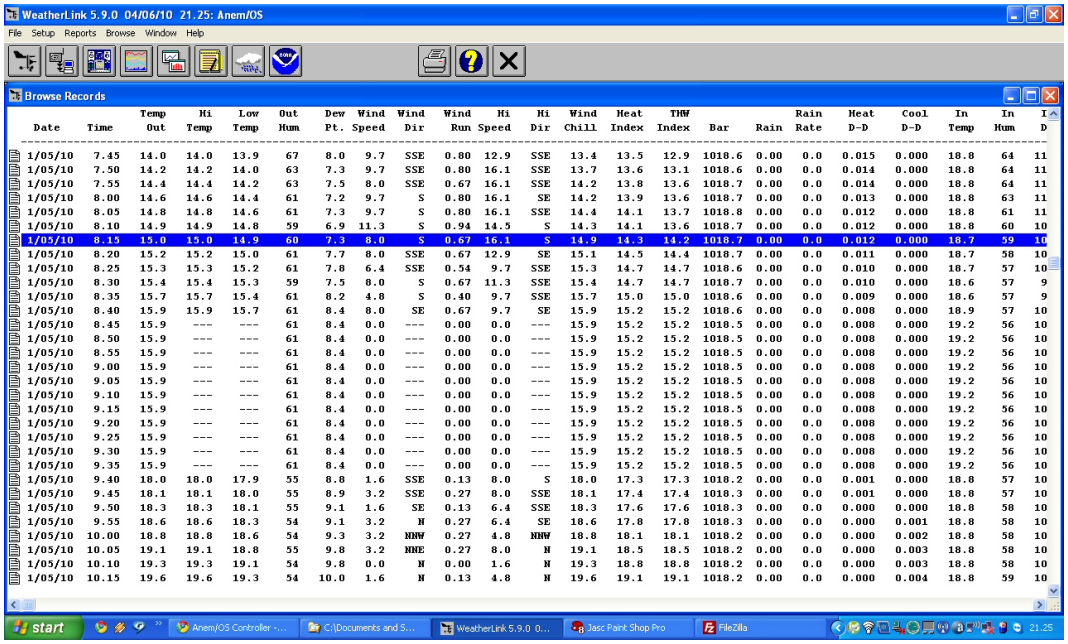
$$s_{a_k} = \begin{cases} A\sin(2\pi f_n t + \varphi_n), & 0<\mathrm{t}<\mathrm{T} \\ 0, & \text{altrimenti} \end{cases} \quad , n = 1, ..., M,$$

ciascuna con una differente frequenza portante f_n e fase φ_n. Le frequenze f_n sono scelte in modo da garantire l'ortogonalità tra le forme d'onda [2]. La versione *coerente* di FSK si differenzia da quest'ultima per il fatto che le fasi delle forme d'onda sono sincronizzate $\varphi_n = \varphi$. Un esempio della modulazione FSK è illustrato in Fig. 4.7. Alternativamente alla FSK, vengono usate altre modulazioni, come ad esempio QAM o PAM, per le quali si rimanda al testo[18].

4.3 Software

Davis fornisce il software WeatherLink per la connessione di un personal computer alla console della stazione meteorologica. Questo applicativo consente di raccogliere, attraverso il collegamento alla rete Ethernet, i dati meteorologici forniti dalla stazione, di creare un database delle misure e di visualizzare una varietà di report e grafici. Inoltre, la console è configurata per inviare automaticamente i dati anche ad un server web messo a disposizione da Davis, che consente di accedere ai propri dati meteorologici senza dover creare una propria pagina web. In Fig. 4.8 è riportata l'interfaccia grafica di WeatherLink.

Figura 4.8: Software WeatherLink



Date	Time	Temp Out	Hi Temp	Low Temp	Out Hum	Dew Pt.	Wind Speed	Wind Dir	Wind Run	Hi Speed	Hi Dir	Wind Chill	Heat Index	THW Index	Bar	Rain	Rain Rate	Heat D-D	Cool D-D	In Temp	In Hum	I D
1/05/10	7.45	14.0	14.0	13.9	67	8.0	9.7	SSE	0.80	12.9	SSE	13.4	13.5	12.9	1018.6	0.00	0.0	0.015	0.000	18.8	64	11
1/05/10	7.50	14.2	14.2	14.0	63	7.3	9.7	SSE	0.80	16.1	SSE	13.7	13.6	13.1	1018.6	0.00	0.0	0.014	0.000	18.8	64	11
1/05/10	7.55	14.4	14.4	14.2	63	7.5	8.0	SSE	0.67	16.1	SSE	14.2	13.8	13.6	1018.7	0.00	0.0	0.014	0.000	18.8	64	11
1/05/10	8.00	14.6	14.6	14.4	61	7.2	9.7	S	0.80	16.1	SE	14.2	13.9	13.6	1018.7	0.00	0.0	0.013	0.000	18.8	63	11
1/05/10	8.05	14.8	14.8	14.6	61	7.3	9.7	S	0.80	16.1	SSE	14.4	14.1	13.7	1018.8	0.00	0.0	0.012	0.000	18.8	61	11
1/05/10	8.10	14.9	14.9	14.8	59	6.9	11.3	S	0.94	14.5	S	14.3	14.1	13.6	1018.7	0.00	0.0	0.012	0.000	18.8	60	10
1/05/10	8.15	15.0	15.0	14.9	60	7.3	8.0	S	0.67	16.1	S	14.9	14.3	14.2	1018.7	0.00	0.0	0.012	0.000	18.7	59	10
1/05/10	8.20	15.2	15.2	15.0	61	7.7	8.0	SSE	0.67	12.9	SE	15.1	14.5	14.4	1018.7	0.00	0.0	0.011	0.000	18.7	58	10
1/05/10	8.25	15.3	15.3	15.2	61	7.8	6.4	SSE	0.54	9.7	SSE	15.3	14.7	14.7	1018.6	0.00	0.0	0.010	0.000	18.7	57	10
1/05/10	8.30	15.4	15.4	15.3	59	7.5	8.0	S	0.67	11.3	SSE	15.4	14.7	14.7	1018.7	0.00	0.0	0.010	0.000	18.6	57	9
1/05/10	8.35	15.7	15.7	15.4	61	8.2	4.8	S	0.40	9.7	SSE	15.7	15.0	15.0	1018.6	0.00	0.0	0.009	0.000	18.6	57	9
1/05/10	8.40	15.9	15.9	15.7	61	8.4	8.0	SE	0.67	9.7	SE	15.9	15.2	15.2	1018.6	0.00	0.0	0.008	0.000	18.9	57	10
1/05/10	8.45	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	8.50	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	8.55	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.00	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.05	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.10	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.15	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.20	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.25	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.30	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.35	15.9	---	---	61	8.4	0.0	---	0.00	0.0	---	15.9	15.2	15.2	1018.5	0.00	0.0	0.008	0.000	19.2	56	10
1/05/10	9.40	18.0	18.0	17.9	55	8.8	1.6	SSE	0.13	8.0	S	18.0	17.3	17.3	1018.2	0.00	0.0	0.001	0.000	18.8	57	10
1/05/10	9.45	18.1	18.1	18.0	55	8.9	3.2	SSE	0.27	8.0	SSE	18.1	17.4	17.4	1018.3	0.00	0.0	0.001	0.000	18.8	57	10
1/05/10	9.50	18.3	18.3	18.1	55	9.1	1.6	SE	0.13	6.4	SSE	18.3	17.6	17.6	1018.3	0.00	0.0	0.000	0.000	18.8	58	10
1/05/10	9.55	18.6	18.6	18.3	54	9.1	3.2	N	0.27	6.4	SE	18.6	17.8	17.8	1018.3	0.00	0.0	0.000	0.001	18.8	58	10
1/05/10	10.00	18.8	18.8	18.6	54	9.3	3.2	NNW	0.27	4.8	NNW	18.8	18.1	18.1	1018.2	0.00	0.0	0.000	0.002	18.8	58	10
1/05/10	10.05	19.1	19.1	18.8	55	9.8	3.2	NNE	0.27	8.0	N	19.1	18.5	18.5	1018.2	0.00	0.0	0.000	0.003	18.8	58	10
1/05/10	10.10	19.3	19.3	19.1	54	9.8	0.0	N	0.00	1.6	N	19.3	18.8	18.8	1018.2	0.00	0.0	0.000	0.003	18.8	58	10
1/05/10	10.15	19.6	19.6	19.3	54	10.0	1.6	N	0.13	4.8	N	19.6	19.1	19.1	1018.2	0.00	0.0	0.000	0.004	18.8	59	10

WeatherLink fornisce le seguenti funzionalità:

1. visualizzazione dei dati correnti in un "bollettino" real-time o in formato testuale (su file .txt);

2. possibilità di configurare la console dal proprio PC;

[2] si ricorda che due forme d'onda sono ortogonali se il loro prodotto interno è nullo

3. visualizzazione dei dati sotto forma di grafici su base oraria, giornaliera, mensile o annuale;

4. costruzione di report in formato NOAA (*National Oceanic and Atmospheric Administration*);

5. collezione dei dati da più stazioni sullo stesso PC;

6. consente agli aderenti al *Citizen Weather Observer Program* (CWOP) di inviare dati meteorologici al National Weather Service [3].

Figura 4.9: Ubicazione del gruppo sensori, sul tetto del DEI/G.

Il software funziona su computer con sistema operativo Windows 98 SE, 2000, XP, Vista o 7 con Microsoft .NET 2.0 framework. Per quanto il database, lo spazio richiesto su disco varia in funzione dell'intervallo d'archiviazione. Ciascun record nel database occupa 88 byte (sono inoltre necessari, ogni giorno, 2 record aggiuntivi per informazioni di sommario), quindi, se i dati vengono archiviati ogni minuto, lo spazio richiesto è approssimativamente di 3.9 MB. Non si ritiene opportuno, in questa sede, approfondire ulteriormente le funzionalità di WeatherLink (si rimanda

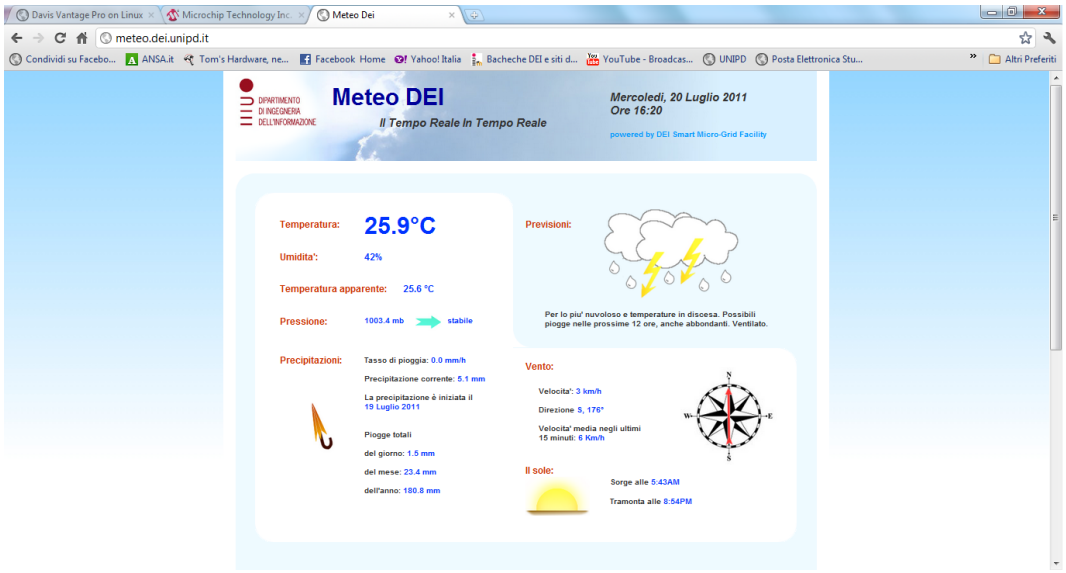
[3]Il CWOP è una partnership pubblico-privata con lo scopo di raccogliere i dati forniti dagli aderenti e renderli disponibili a servizi meteorologici ed enti per la protezione civile.

alla guida[8]), in quanto tale software non è stato impiegato nella realizzazione del sistema di monitoraggio meteorologico Meteo DEI.

4.4 Realizzazione

In questo paragrafo sono riportati alcuni dettagli relativi all'implementazione di Meteo DEI. Il gruppo sensori è stato installato sul tetto dell'edificio DEI/G, nella posizione illustrata in Fig. 4.9. Per approfondimenti sul montaggio e sull'installazione del gruppo sensori, si faccia riferimento all'appendice.

Figura 4.10: La pagina web "Meteo DEI"



Come precisato nel paragrafo precedente, non è stato utilizzato WeatherLink, poichè tale software è pensato per un utenza domestica, e non è particolarmente flessibile. Inoltre, WeatherLink è realizzato per sistemi operativi Microsoft, mentre i server del Dipartimento utilizzano sistemi basati su Linux. Queste difficoltà sono state superate grazie ai sistemisti del DEI, che si sono serviti di alcuni applicativi disponibili in rete all'indirizzo http://www.joejaworski.com per acquisire i dati dalla console. Questa pagina web contiene il software e le indicazioni necessarie per utilizzare la stazione meteorologica attraverso un server Linux. In particolare, un programma denominato VPROWEATHER si occupa di dialogare periodicamente con la console, acquisendo le letture in tempo reale, ed accedendo alla memoria EEPROM del data logger per ottenere i massimi/minimi, e i dati memorizzati al fine di costruire i grafici. L'output di questo programma può essere rediretto in un file di testo (privo di formattazione, ovvero con soli caratteri ASCII) che apparirà come di seguito:

Listing 4.1: Output del software VPROWEATHER

```
1 ...
2 hlWindHiMonth = 28
```

```
3 hlWindHiYear = 56
4 hlOutTempHiDay = 78.8
5 hlOutTempLoDay = 68.9
6 hlOutTempHiTime = 4:26PM
7 ...
```

In questo esempio, la velocità più elevata del vento durante il mese scorso è stata di 28 miglia orarie. Sono stati, poi, creati degli script per la conversione in unità metriche, e per la traduzione delle stringhe contenenti le previsioni meteorologiche (le quali, ovviamente, sono in inglese), che hanno permesso lo sviluppo della pagina web Meteo DEI, disponibile all'indirizzo http://meteo.dei.unipd.it/. La pagina si presenta come illustrato in Fig. 4.10.

Attualmente, si sta cercando di migliorare le previsioni meteorologiche fornite dalla stazione (che si sono rivelate non molto accurate), seguendo due strade:

- sfruttando un algoritmo che cerca di formulare delle previsioni sulla base dei dati grezzi forniti dalla console;
- incrociando previsioni di diversa origine (quelle fornite dalla console con quelle generate dall'algoritmo citato, per esempio).

Capitolo 5

Conclusioni

Il presente lavoro di tesi presenta e descrive tre aspetti del sistema di monitoraggio ambientale operativo presso i dipartimenti DEI e DIE dell'Università di Padova: l'architettura generale del sistema, il progetto di estensione del monitoraggio al consumo di gas, e l'implementazione del servizio meteorologico *Meteo DEI*. Si è trattato di un'esperienza preziosa dal punto di vista formativo e tecnico, perchè ha consentito di lavorare su diverse tecnologie appartenenti ai settori dell'Elettronica, dell'Informatica e delle Telecomunicazioni, ha comportato un confronto con le difficoltà tecniche che si presentano spesso nelle fasi realizzative, ed ha permesso la collaborazione con numerose persone, tra cui sistemisti e personale dei Dipartimenti, tecnici di Hyphen e Sacofgas, nonchè altri tesisti.

L'architettura Pin-Energy è stata installata e collaudata nel corso del precedente anno accademico, ed è tuttora in fase di studio ed evoluzione. Tra i possibili sviluppi futuri è previsto, a breve, il rilascio di una nuova versione del software Pincushion, che offrirà maggiori efficenza e flessibilità, e l'estensione delle funzionalità del sistema al controllo web-based: in quest'ottica, Pincushion piloterebbe i moduli iPin-Energy in modo da farli reagire all'acquisizione di determinati eventi impostando adeguatamente le linee di uscita. Un primo passo in questo senso potrebbe consistere nel controllare via Pin-Energy l'impianto luci di un ufficio o un laboratorio, configurando l'architettura in modo da far spegnere o accendere automaticamente le luci a determinati orari, o in presenza/assenza di persone, con l'ausilio di alcuni sensori. L'integrazione della funzionalità di controllo ambientale rappresenterà un significativo passo in avanti nell'ambito dello studio e della ricerca sulle *smart grid*.

Per quanto riguarda il progetto finalizzato ad ottenere il monitoraggio del consumo di gas, si è trattato in primo luogo di capire se e come fosse possibile effettuare la telelettura del misuratore, cosa che ha comportato un lavoro di ricerca e di confronto con i tecnici di Sacofgas, dopodichè si è passati alla fase di progettazione e sviluppo. In questo senso, si è proseguito con un'analisi delle possibili soluzioni tecniche per realizzare la comunicazione, che ha portato alla scelta del protocollo IEEE 802.15.4, e all'individuazione delle soluzioni presentate per realizzare il trasmettitore ed il ricevitore. L'ultima fase è stata l'implementazione dell'applicazione sulle schede di sviluppo, che ha comportato un notevole lavoro di sperimentazione ed una indispensabile collaborazione con altre persone, al fine di superare le difficoltà che si sono presentate nel corso dell'attività. I passi successivi, sulla via della realizzazione, sa-

ranno l'assemblaggio delle schede che andranno montate sul contatore e sul modulo iPin-Energy, e l'ulteriore sviluppo del firmware, che allo stato attuale è funzionante ma rudimentale. Potranno essere implementati, per esempio, un meccanismo di *acknowledge* ed una tecnica di controllo e correzione degli errori, in modo da raffinare e rendere maggiormente robusta la comunicazione tra le due board.

La realizzazione del servizio meteorologico è iniziata con l'assemblaggio ed il test del gruppo sensori e della console, accompagnato naturalmente da un'opera di documentazione tecnica su questi dispositivi, finalizzata a comprendere come funzionassero e come fosse possibile utilizzarli. Successivamente, è stato portato avanti un lavoro di sperimentazione con l'obiettivo di sfruttare il software di Davis Instruments per raccogliere i dati dalla console e consentirne l'uso. Questa soluzione si è però rivelata troppo macchinosa, ed è stato decisivo l'aiuto dei sistemisti del DEI, che hanno sviluppato il software necessario all'implementazione della pagina web, come descritto nel capitolo precedente.

In conclusione, questo lavoro di tesi ha contribuito a documentare ed arricchire, seppur modestamente, l'attività di ricerca dei Dipartimenti sulle *smart grid* e più specificamente sul monitoraggio ambientale, aspetti che saranno, nei prossimi anni, di primaria importanza nel settore della gestione energetica.

Appendice A

MPLAB ed ICD2

Questo capitolo descrive i passi compiuti per implementare il trasmettitore ed il ricevitore descritti nel Capitolo 3 su due schede di sviluppo PICDEM Z.

In primo luogo vengono presentate le istruzioni necessarie all'installazione dell'ambiente di sviluppo MPLAB, della toolchain C-18 e dei driver per l'utilizzo del debugger ICD2. In seguito, viene presentata una guida rapida all'uso dell'ambiente di sviluppo e del debugger.

Questa descrizione fa riferimento al sistema operativo Windows 7, ma le istruzioni sono analoghe anche per le versioni precedenti del software Microsoft.

A.1 Installazione del software necessario

Se il personal computer con il quale si sta lavorando non dispone del software indicato, la prima cosa da fare è procurarsi le installazioni e seguire i passi indicati nel seguito.

Ambiente di sviluppo MPLAB L'archivio contenente il setup di MPLAB è disponibile sul sito di Microchip, nella sezione Home>Products Home Page>Development Tools Main Page>MPLAB IDE.

Una volta scaricato l'archivio, lo si decomprima e si lanci l'installer *setup.exe*, che determinerà l'avvio della consueta procedura guidata. I files dell'installazione vengo copiati, salvo modifiche, nella cartella C:\Program Files\Microchip\. La versione di MPLAB utilizzata per l'implementazione in questione è la 8.40, ed è a licenza freeware.

Catena di compilazione C-18 La toolchain, come precisato nel Capitolo 3, va installata separatamente rispetto all'IDE. L'eseguibile è disponibile, di nuovo, sul sito di Microchip, nella sezione Home>Products Home Page>Development Tools Main Page>Compilers>MPLAB C Compiler for PIC18 MCUs. In questa pagina sono disponibili una versione *LITE* completamente libera ed una versione standard di valutazione, che offre la possibilità di testare liberamente la catena di compilazione per un periodo di 60 giorni: la prima è assolutamente sufficiente per la compilazione del codice impiegato per questa applicazione. Il percorso standard per l'installazione è C:\Program Files\Microchip\MCC18\.

L'installazione di questa toolchain determina l'aggiornamento automatico delle variabili d'ambiente[1], che si ritiene utile riportare in seguito, in caso di errori o malfunzionamenti:

- la directory contenente gli eseguibili del compilatore, normalmente C:\Program Files\Microchip\MCC18\bin\viene aggiunta all'inizio della variabile PATH;
- la directory contenente gli eseguibili dell'assemblatore MPASM, normalmente C:\Program Files\Microchip\MCC18\mpasm\viene aggiunta all'inizio della variabile PATH;
- la directory contenente i file header, normalmente C:\Program Files\Microchip\MCC18\h\viene aggiunta all'inizio della variabile MCC_INCLUDE;

Per verificare la corretta installazione di C-18, eseguire il file di batch C:\Program Files\Microchip\MCC18\example\an696\buildit.bat. Se il processo è stato portato a termine senza errori, comparirà il file 18motor.out.

Infine, l'IDE MPLAB viene aggiornato con i percorsi del compilatore, del linker MPLINK e dell'assemblatore MPASM. Anche in questo caso, in caso di malfunzionamenti, è possibile impostare i percorsi manualmente, come verrà evidenziato nella sezione successiva.

Configurazione del debugger ICD2 Il debugger ICD2 può comunicare con il PC in due modi:

- tramite la porta seriale, nel qual caso il dispositivo necessita di un'alimentazione esterna;
- tramite la porta USB, nel qual caso l'alimentazione è ottenuta attraverso la porta stessa.

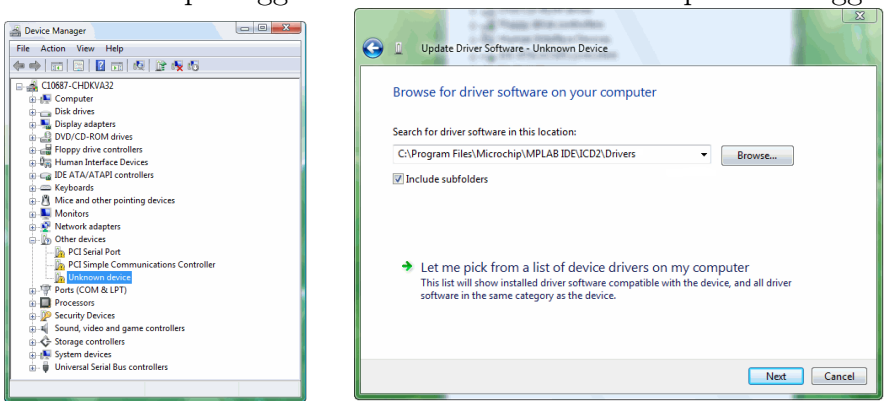
Nel caso in esame si è fatto uso della comunicazione tramite USB. La comunicazione con la scheda PICDEM Z, invece, avviene tramite un cavo con connettori RJ-11, e la scheda necessita di alimentazione esterna.

Se, come riportato in questa guida, l'IDE MPLAB è stato già installato, a questo punto si può connettere il dispositivo ICD2 al computer (non collegare l'ICD2 alla scheda, per il momento). Comparirà il consueto messaggio, riportante il fatto che il computer ha rilevato un nuovo hardware, tuttavia Windows non sarà in grado di trovare un driver corretto. Si eseguano dunque le seguenti operazioni:

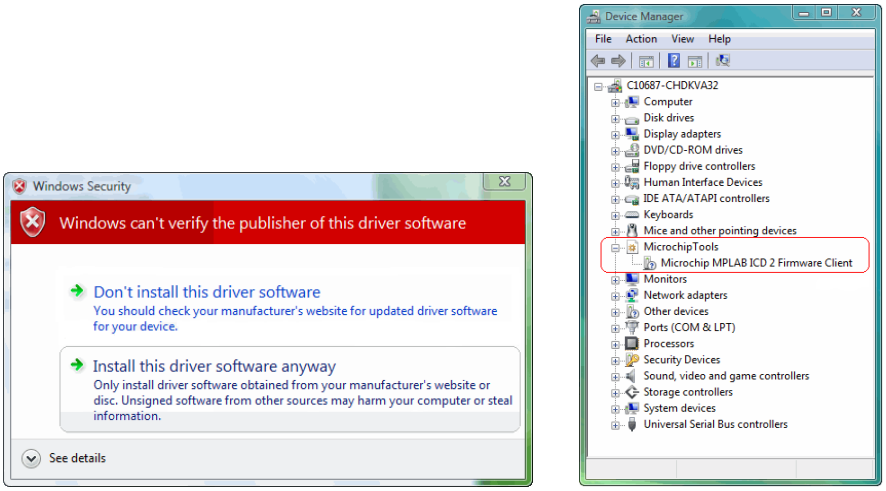
1. entrare in Pannello di Controllo>Gestione dispositivi;
2. cercare la voce 'Altri dispositivi', sotto la quale si trova 'Dispositivo sconosciuto', come illustrato in Fig. A.1a;
3. cliccare con il tasto destro su 'Dispositivo sconosciuto': compariranno una serie di voci, tra cui 'Aggiornamento software driver...'. Selezionare quest'ultima;

[1]Su Windows 7 la scheda delle variabili d'ambiente è accessibile da Pannello di Controllo>Sistema>Impostazioni di sistema avanzate>Variabili d'ambiente...

Figura A.1: Alcuni passaggi dell'installazione dei driver per il debugger ICD2



(a) (b)



(c) (d)

4. selezionare 'Cerca il software del driver nel computer';

5. compare una finestra nella quale si chiede di inserire il percorso dei driver. Selezionare 'Sfoglia' e inserire il percorso, che normalmente è C:\Program Files\Microchip\MPLAB IDE\ICD2\Drivers, come illustrato in Fig. A.1b, e premere Avanti.

6. compare ora la finestra di Windows Security, nella quale si viene avvisati del fatto che Windows non riconosce il produttore del driver. Selezionare 'Installa il software driver comunque', come illustrato in Fig. A.1c;

7. la procedura è completa. Per verificare la corretta installazione, controllare che, in Gestione dispositivi, sia presente la voce Microchip Tools>Microchip MPLAB ICD2 Firmware Client, come evidenziato in Fig. A.1d.

A.2 Guida rapida all'uso

In questa sezione si descrivono le operazioni necessarie per usare MPLAB, assieme al dispositivo ICD2, per programmare il PIC montato sulla scheda di sviluppo. Sono riportate, nell'ordine:

- le istruzioni per configurare l'IDE;
- la sequenza di operazioni da eseguire per alimentare l'ICD2 e la scheda, ed assicurare la correttezza delle comunicazioni nella configurazione considerata, ovvero connessione USB tra ICD2 e PC, ed alimentazione esterna della scheda[2];
- alcune istruzioni di base per l'uso di MPLAB;

Configurazione dell'IDE Dopo aver lanciato l'IDE, è necessario configurarlo con le impostazioni relative al debugger in questione, eseguendo le seguenti operazioni:

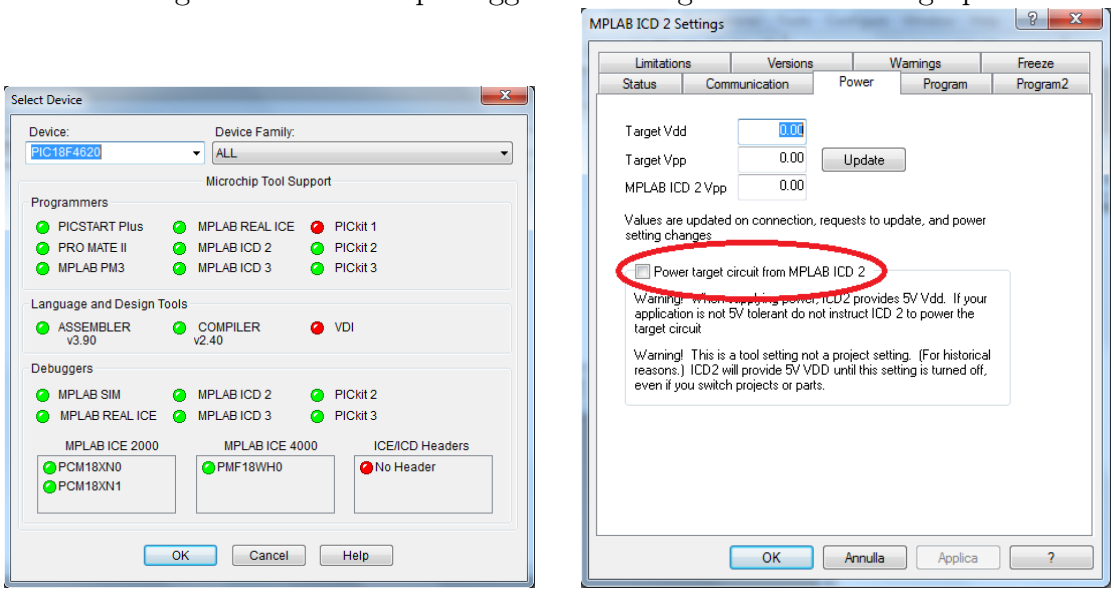
1. selezionare Configure>Select Device, dopodichè, dal menù a tendina 'Device', impostare il PIC18F4620: si noterà che, sotto le voci 'Programmers' e 'Debuggers', MPLAB ICD2 è contrassegnato da un pallino verde, che indica la compatibilità con il microcontrollore (Fig. A.2a);
2. selezionare Debugger>Select Tool>MPLAB ICD2, per impostare il dispositivo come debugger: la finestra 'Output' si aprirà sulla scheda MPLAB ICD2, che evidenzia lo stato della connessione con il PC;
3. selezionare Debugger>Settings, e la scheda 'Communications', per impostare la comunicazione via USB;

Sequenza di power-up Si eseguano le seguenti azioni per alimentare correttamente ICD2 e scheda:

1. collegare l'ICD2 al PC e lanciare MPLAB, se questo non è già stato fatto;
2. assumendo che MPLAB sia già stato configurato come descritto nel paragrafo precedente, impostare l'ICD2 come debugger dal menù Debugger>Select Tool;
3. selezionare Debugger>Settings, e la scheda 'Power', quindi assicurarsi che la voce 'Power target circuit from MPLAB ICD2' NON sia spuntata, come evidenziato in Fig. A.2b;
4. alimentare la scheda e selezionare Debugger>Connect. Nella finestra 'Output' compariranno dei messaggi che indicano la stato della connessione con la board.

[2]Esistono configurazioni diverse: l'ICD2 può comunicare con il PC tramite connessione RS232, nel qual caso va alimentato separatamente; con questa configurazione l'ICD2 può alimentare una piccola scheda. Si faccia rifrimento alla guida[13] per maggiori dettagli.

Figura A.2: Alcuni passaggi della configurazione e setting-up



(a) (b)

Utilizzo di MPLAB Il modo più semplice per creare un progetto con MPLAB è cliccare su Project>Project Wizard. Si avvia quindi una procedura guidata che permette di:

1. scegliere il microcontrollore (PIC18F4620 nel caso in esame);

2. scegliere la catena di compilazione, attraverso il menù a tendina 'Active Toolsuite': va selezionata la voce Microchip C18 Toolsuite[3]. Si osservi la Fig. A.3a;

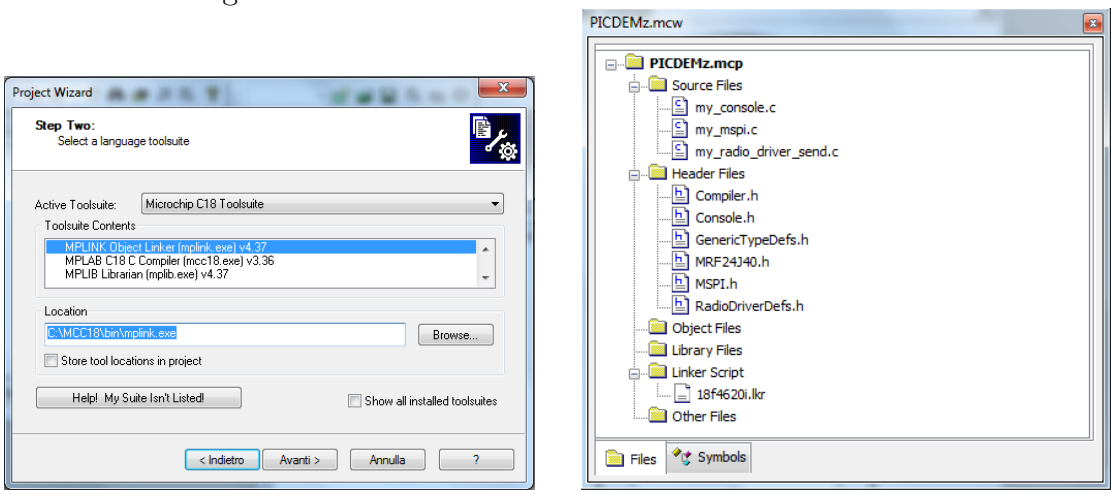
3. impostare il percorso del progetto;

4. aggiungere subito del codice già scritto.

Terminato il wizard, viene creato il progetto, e compare la finestra <nome_progetto>.mcw, che illustra la Fig. A.3b, una struttura ad albero con la directory principale e delle sub-directory come 'Source Files' o 'Header Files', le quali andranno a contenere, chiaramente, i file .c ed i .h coinvolti nel progetto. Va sottolineato che, al fine di permettere al programma di generare correttamente il codice caricabile, è necessario aggiungere alla directory 'Linker Script' il file .lkr relativo al microcontrollore in questione, disponibile normalmente nella cartella C:\Program Files\MCC18\bin\LKR\. Si precisa che, al fine di programmare il PIC è necessario come minimo includere, nel codice sorgente, il file header relativo, disonibile nella cartella C:\Program Files\MCC18\h\, e che vanno impostati, tramite direttive inserite nel sorgente, oppure dal menù Configuration>Configuration Bits, alcuni registri

[3]Se compare una croce rossa in corrispondenza dei componenti della toolsuite, assicurarsi che il campo 'Location' contenga il percorso esatto.

Figura A.3: Alcuni finestre relative all'uso di MPLAB



(a) (b)

fondamentali del microcontrollore, come quelli che impostano il funzionamento dell'oscillatore. Data l'impronta di 'Guida rapida' data a questa sezione, non si ritiene opportuno approfondire l'argomento, e si rimanda alla guida[14].

Le varie opzioni di compilazione si possono impostare dal menù accessibile selezionando Project>Build Options>Project, nella scheda MPLAB C18, mentre con il comando Project>Set Language Tool Locations si possono impostare i percorsi degli eseguibili, ed i path di ricerca predefiniti per i file header, le librerie, i file .lkr.

Per la compilazione del codice si utilizza il comando Project>Make (o il tasto rapido F10): se il processo va a buon fine, la scheda 'Build' della finestra 'Output', riporterà la stringa *BUILD SUCCEDED*. Al fine di caricare il codice macchina nella memoria del PIC, analogamente, si impiega il comando Debugger>Program: a questo punto il software può essere eseguito impiegando le funzioni di debug fornite dall'IDE. La funzione Debugger>Run (tasto rapido F9) consente di eseguire il programma normalmente, mentre i comandi 'Step Into', 'Step Over' e 'Step Out', sempre sotto il menù Debugger, permettono di osservare l'esecuzione un'istruzione alla volta. Molto utili sono i *breakpoints*, che consentono di marcare le linee di codice presso le quali si desidera che l'esecuzione si blocchi, al fine di osservarne gli effetti. Per impostare un breakpoint è sufficiente fare un doppio click a fianco della linea di codice di interesse.

Utili funzionalità per il debug delle applicazioni sono, inoltre, l'applicazione accessibile da View>Watch, che consente di monitorare il contenuto di registri e variabili, ed il comando View>Dissassembly Listing, tramite il quale si può osservare il codice compilato ma non assemblato, in istruzioni assembly, dunque.

Una volta debuggato il codice, per effettuare la programmazione vera e propria, è necessario deselezionare l'ICD2 come debugger, ed impostarlo come programmatore, cliccando Programmer>Select Programmer>MPLAB ICD2. I comandi, a questo punto, sono assolutamente analoghi a quelli del menù Debugger: sarà necessario effettuare la connessione con la scheda, mediante il comando Connect, configurare le impostazioni del programmatore dal menù Settings, mentre il comando Program

consentirà, di nuovo, di effettuare la programmazione.

La differenza tra la programmazione effettuata in modalità debugger, e quella in modalità programmer, è il fatto che nel primo caso il codice può essere eseguito solo con il debugger collegato, e con il vantaggio di poter sfruttare le funzioni di debug, mentre nel secondo caso l'esecuzione inizia subito (non c'è un comando 'Run') e prosegue anche se si disconnette la scheda dal dispositivo ICD2.

Appendice B

Telelettura: implementazione su PICDEM Z

Questo capitolo intende presentare una panoramica sul codice sorgente impiegato per programmare i PIC sulle due schede, necessariamente riassuntiva, data la dimensione dello stesso: verranno commentati alcuni estratti relativi al processore PIC18F4620[1].

L'applicazione è stata sviluppata sulla base del codice sorgente disponibile come esempio nel MiWi Development Environment, appartenente al pacchetto Microchip Application Libraries[2]. Tale codice è molto ampio, ed è stato progettato per essere adatto a più processori e schede di sviluppo: pertanto è stato modificato e semplificato per adattarlo all'applicazione in esame, e ne sono stati ricavati due progetti, radio_driver_send e radio_driver_receive, rispettivamente per il trasmettitore e per il ricevitore. La prima sezione richiama alcuni concetti sul compilatore e sui files coinvolti, utili a comprendere quanto verrà trattato in seguito, mentre la seconda presenta la struttura dei progetti e i files in essi contenuti. Segue un analisi del setting dei registri di configurazione, comune ad entrambe le schede, una presentazione delle funzioni impiegate, e la descrizione del *main*, la funzione principale.

B.1 Concetti utili

Quando si parla del software che si occupa di tradurre un codice sorgente nel corrispondente codice eseguibile dal sistema di elaborazione in esame, si usa tipicamente il termine *compilatore*. In realtà, il codice eseguibile è il risultato di elaborazioni eseguite da una serie di programmi (tra cui il compilatore propriamente detto), che costituiscono la *catena di compilazione*, ognuno dei quali lavora sull'output del precedente.

Il programmi scritti in linguaggio C contengono istruzioni scritte con la sintassi propria del linguaggio stesso, ed altri costrutti sintattici, denominati direttive, macro o meta-istruzioni, che NON sono comprensibili al compilatore. Queste par-

[1] Il codice, infatti, contiene parti dedicate a vari processori delle serie PIC18 e PIC24.

[2] La versione più recente è disponibile sul sito di Microchip seguendo il percorso Home>Products Home Page>16-bit PIC MCUs & dsPIC Digital Signal Controllers>Microchip Application Libraries.

ticolari istruzioni vengono trattate dal *preprocessore*, il primo software della catena di compilazione, che si occupa di espanderle in linguaggio C puro. Le direttive sono fondamentalmente di tre tipi:

- direttive di definizione: hanno sintassi composta dalla parola chiave #define, seguita da un'identificatore e da un elemento sintattico del linguaggio C. Ad esempio, la direttiva *#define TRUE 1*, comporta che il preprocessore sostituisca tutti i 'TRUE' che compaiono nel codice con la costante 1;

- direttive di inclusione: la loro sintassi si caratterizza per la presenza della parola chiave #include, seguita dal nome di un file *header*, ad esempio *#include <stdlib.h>*. Questi files, che hanno estensione .h, contengono le dichiarazioni di funzioni e variabili, e l'inclusione di uno di essi comporta l'espansione, nell'unità di compilazione corrente, di tali istruzioni. Il vantaggio di questo meccanismo sta nel fatto che esso consente di non dover riscrivere le istruzioni ovunque esse servano, e di rendere più modulare ed ordinata l'organizzazione del progetto;

- direttive condizionali: sono molto simili ai costrutti if-else presenti in ogni linguaggio di programmazione, con la differenza che la sintassi, tipicamente è *#if defined <nome> <blocco di codice in C puro> #end if*. Il loro effetto è l'inserimento nell'unità di compilazione corrente del blocco di codice se <nome> è stato definito in precedenza.

Oltre a queste direttive universali, ne esistono altre impiegate solo da alcune specifiche toolchain, come la direttiva #pragma del C-18, il cui significato verrà spiegato nel seguito, con riferimento ad esempi concreti.

L'output del preprocessore, come già evidenziato, è un file in C puro, che viene elaborato dal compilatore vero e proprio. Funzione di quest'ultimo è tradurre il codice in C, in un codice nel linguaggio assembly dell'architettura in esame. Interverrano poi almeno altri tre programmi: l'assemblatore, che traduce il file assembly nel codice oggetto, contenente il programma in linguaggio macchina, ma in una forma non ancora caricabile; il linker che si occupa di collegare il file oggetto prodotto ad altri files oggetto e librerie preassemblati; il loader, la cui funzione è organizzare il codice ricevuto come input in modo che sia effettivamente caricabile nella memoria del PIC.

Un progetto in MPLAB, come spiegato nell'Appendice A, è strutturato in modo da raggruppare tutti i files necessari per categorie. La cartella 'Header Files' contiene i files .h inclusi mediante le apposite direttive nei file .c, i quali sono raggruppati nella cartella 'Source Files'[3].

B.2 Struttura dei progetti

I progetti radio_driver_send e radio_driver_receive sono in realtà molto simili: la sola differenza sta, sostanzialmente, nella funzione *main*.

[3]MPLAB si occupa di gestire i percorsi dei file aggiunti al progetto, sollevando il programmatore dall'onere di specificarli manualmente ad ogni compilazione.

I files header usati sono i seguenti:

1. Compiler.h: contiene le direttive per l'inclusione dei files header della libreria standard e della libreria dei microcontrollori PIC18, oltre ad una serie di direttive di definizione che consentono di fare riferimento ai bit di alcuni registri del PIC, o ad alcune funzioni, attraverso denominazioni 'comode': ad esempio, la direttiva *#define ADON ADCON0bits.ADON*, consente di fare riferimento ad un bit del registro ADCON0, destinato a stabilire se il convertitore A/D è acceso o spento, digitando 'ADON' anzichè 'ADCON0bits.ADON';

2. GenericTypeDefs.h: contiene le definizioni di alcuni tipi di dato che vengono usati frequentemente nell'applicazione in esame. Ad esempio, l'istruzione *typedef unsigned char BYTE;* consente di definire variabili intere ad 8 bit dichiarandole di tipo BYTE, ed aumentando la chiarezza del codice;

3. RadioDriverDefs.h: contiene altre definizioni utili perchè consentono di fare riferimento a registri e strutture dati in modo chiaro e semplice. Queste definizioni sono contenute in un file separato rispetto a quelle di Compiler.h perchè sono relative a registri e strutture dati relativi all'applicazione radio;

4. MRF24J40.h: contiene delle direttive di definizione che associano denominazioni mnemoniche agli indirizzi delle locazioni di memoria impiegate dal PIC per dialogare con il modulo radio;

5. MSPI.h: contiene le direttive di inclusione di Compler.h e GenericTypeDefs.h, ed alcune dichiarazioni e definizioni utili alla gestione della porta SPI, usata dal PIC per comunicare con il modulo radio;

6. Console.h: è un header file pensato per la gestione dell'applicazione radio attraverso una *console*, che consenta una maggiore interazione con l'utente. Questa funzionalità non è stata impiegata nel caso in esame. Contiene le direttive di inclusione di Compler.h e GenericTypeDefs.h, e RadioDriverDefs.h.

La cartella 'Source Files' contiene, invece, i seguenti tre files:

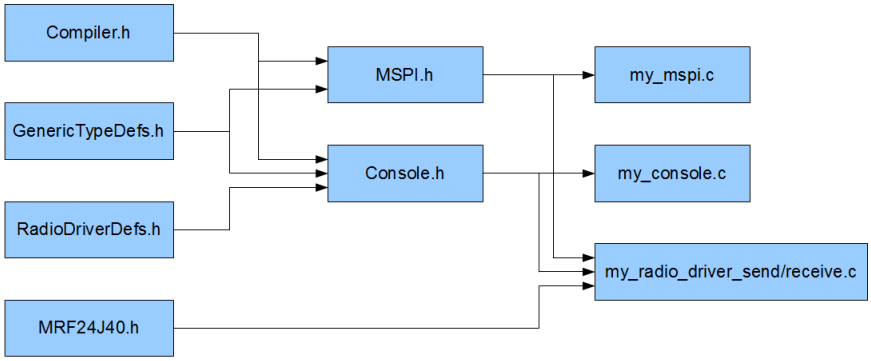
1. my_console.c: contiene le implementazioni delle funzioni per gestire la console, che, come già sottolineato, non viene usata nel caso in esame;

2. my_mspi.c: contiene le implementazioni delle funzioni *SPIPut* ed *SPIGet*, per la scrittura e lettura di un byte sul bus SPI;

3. my_radio_driver_send.c/my_radio_driver_receive.c: è il sorgente che contiene la funzione *main*, il punto di partenza per l'esecuzione dell'applicazione. Inoltre, contiene l'implementazione di numerose funzioni, ognuna delle quali gestisce un aspetto dell'applicazione, che verranno analizzate in seguito.

La Fig. B.1 evidenzia le relazioni esistenti tra i file del progetto, con le frecce a rappresentare le direttive di inclusione: il blocco da cui parte ogni freccia indica il file incluso, mentre quello in cui la freccia arriva rappresenta il file nel quale si trova la direttiva. Infine, va segnalato l'utilizzo del linker script 18f4620i.lkr, necessario al linker per completare la compilazione dell'eseguibile.

Figura B.1: Relazioni tra i file dei progetti



B.3 Configuration bits e funzioni principali

Questa sezione descrive le direttive usate per configurare il funzionamento di base del PIC, e le funzioni più importanti usate per ottenere il funzionamento voluto. In Listing B.1 è riportato un'estratto del codice relativo al microntrollore in esame. La meta-istruzione #if defined(__18F4620) comporta l'inserimento delle istruzioni seguenti, nel codice che viene passato dal preprocessore al compilatore, solo se il processore sulla scheda collegata all'ICD2 è un PIC18F4620. In caso contrario, il preprocessore semplicemente ignora le istruzioni del blocco che segue, e passa ad eseminare la macro #elif defined. Le direttive #pragma, una caratteristica peculiare del compilatore C-18, in questo caso servono ad allocare delle costanti nella memoria di programma, grazie alla parola chiave *romdata* (esistono altre direttive che consentono l'allocazione nella memoria dati). Seguono tale parola chiave il nome dell'area di memoria, in stampatello maiuscolo, il suo indirizzo e la definizione della costante. Queste aree di memoria vengono interpretate come registri di configurazione del processore[4], ed i valori in esse contenuti definiscono il suo funzionamento: ad esempio, l'assegnazione del valore binario 0b00000110 al registro CONFIG1H determina il funzionamento dell'oscillatore in modalità HS con PLL abilitato. Per conoscere nel dettaglio i valori da assegnare a tali registri si faccia riferimento ai fogli tecnici del processore[16].

Listing B.1: Direttive per l'assegnazione dei registri di configurazione del PIC

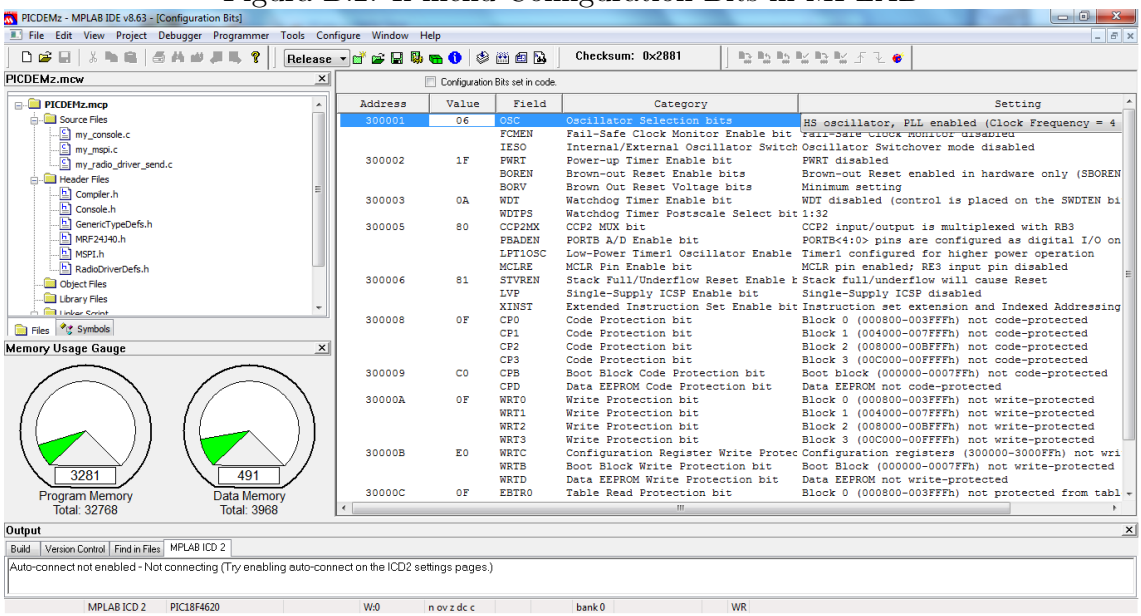
```
1     ...
2     //Define the configuration bits for each processor
3    #if defined(__18F4620)
4        //define the configuration bits for PICDEM Z board - PIC18F4620
5       #pragma romdata CONFIG1H = 0x300001
6       const rom unsigned char config1H = 0b00000110; // HSPLL oscillator
7
8       #pragma romdata CONFIG2L = 0x300002
9       const rom unsigned char config2L = 0b00011111; // Brown-out Reset Enabled
10                                                     // in hardware @ 2.0V, PWRTEN disabled
11      #pragma romdata CONFIG2H = 0x300003
```

[4]Con riferimento alla linea di codice 5, il processore *sa* che all'indirizzo 0x300001 della memoria di programma troverà la configurazione di bit corrispondente al registro CONFIG1H.

```
12          const rom unsigned char config2H = 0b00001010; // HW WD disabled, 1:32 prescaler
13
14          #pragma romdata CONFIG3H = 0x300005
15          const rom unsigned char config3H = 0b10000000; // PORTB digital on RESET
16
17          #pragma romdata CONFIG4L = 0x300006
18          const rom unsigned char config4L = 0b10000001; // DEBUG disabled,
19                                                         // XINST disabled
20                                                         // LVP disabled
21                                                         // STVREN enabled
22      //define the configuration bits for PIC18 Explorer - PIC18F87J11
23      #elif defined(__18F87J11)
24       ...
```

Un altro meccanismo che può essere impiegato, in alternativa, per configurare il funzionamento del processore è utilizzare il menù accessibile selezionando Configure>Configuration Bits..., e togliendo la spunta all'opzione 'Configuration Bits, come illustrato in Fig. B.2.

Figura B.2: Il menù Configuration Bits in MPLAB



Si passa ora a descrivere le funzioni che permettono il funzionamento dell'applicazione, i cui prototipi sono messi in evidenza in Listing B.2.

Listing B.2: Funzioni impiegate in my_radio_driver_send/receive.c

```
1  ...
2  /********************** FUNCTION PROTOTYPES ************************/
3  void BoardInit(void);
4  void PHYSetLongRAMAddr(WORD address, BYTE value);
5  void PHYSetShortRAMAddr(BYTE address, BYTE value);
6  BYTE PHYGetShortRAMAddr(BYTE address);
7  BYTE PHYGetLongRAMAddr(WORD address);
8  void MRF24J40Init(void);
9  void SetChannel(BYTE channel);
10 void PHYSetDeviceAddress(WORD PANID, WORD shortAddress);
```

```
11 BYTE RSSIlookup(BYTE RSSIval);
12 void PrintStatus(void);
13 void PrintHeader(ROM char *app);
14 void PrintSubHeader(ROM char *app);
15 void ResetMRF24J40(void);
16 void ControlFunctions(BYTE data);
17 void Send_Packet(BYTE Packet_Type);
18 void msDelay(unsigned int ms);
19 ...
```

1. la funzione *BoardInit*, come indica il suo nome, ha il compito di inizializzare la scheda, e contiene una serie di assegnazioni che interessano i registri del PIC riguardanti la gestione delle interruzioni, del modulo SPI e delle porte di I/O. Nel dettaglio, PORTB, ai cui pin 4 e 5 sono connessi gli switch viene impostata come porta di ingresso digitale; PORTA, ai cui pin RA0 ed RA1 sono connessi i LED, viene configurata in modo che tali terminali siano uscite digitali; PORTC viene configurata in modo da pilotare correttamente il bus SPI; vengono attivate le interruzioni con priorità alta;

2. le funzioni *PHYSet/GetLong/ShortRAMAddr*, servono a scrivere/leggere la locazione della memoria del chip MRF24J40 con l'indirizzo specificato (che può essere ad 8 o 16 bit), attraverso la porta SPI, pilotato grazie alle funzioni *SPIPut* e *SPIGet*. Tale spazio di memoria è diviso in registri di controllo e data buffer, la cui lettura e scrittura consente al PIC di gestire il modulo;

3. *MRF24J40Init* inizializza il modulo radio attraverso una serie di chiamate alle funzioni *PHYSet/GetLong/ShortRAMAddr*;

4. *SetChannel* seleziona il canale passato come parametro, di nuovo impiegando le funzioni per la lettura/scrittura nella memoria del dispositivo;

5. la funzione *PHYSetDeviceAddress* imposta il PAN ID e l'indirizzo a 16 bit con i quali si intende lavorare;

6. la funzione *RSSIlookup* converte il valore RSSI in decibel. Questo valore indica l'intensità media del segnale dell'ultimo pacchetto ricevuto;

7. le funzioni *PrintStatus*, *PrintHeader* e *PrintSubHeader* riguardano la console, pertanto il loro ruolo non viene approfondito;

8. la funzione *ResetMRF24J40* ha, ovviamente, lo scopo di ripristinare il modulo radio;

9. *ControlFunctions* riguarda la console, pertanto il suo ruolo non viene approfondito;

10. *Send_ Packet* si serve delle funzioni per scrivere la memoria del chip MRF24J40 per spedire il pacchetto definito da 'Packet_Type';

11. *msDelay* è una funzione che, banalmente, ottiene un ritardo del numero di millisecondi passato come parametro iterando l'istruzione *nop*, che semplicemente non ha alcun effetto.

Per concludere questa sezione, si accenna al meccanismo di segnalazione che il modulo radio usa per comunicare al PIC la ricezione di un nuovo pacchetto, basato sullo sfruttamento delle interruzioni.

Il microcontrollore in questione gestisce interruzioni su due livelli di priorità. Quando si verifica un'interruzione di bassa priorità il processore memorizza l'attuale valore del Program Counter e di alcuni altri registri, e salta ad un'indirizzo prestabilito nella memoria di programma, dove troverà un'istruzione di salto alla routine di servizio delle interruzioni di bassa priorità, programmata dall'utente. Il meccanismo è analogo per le interruzioni di alta priorità.

Nell'applicazione in questione, le interruzioni di alta e bassa priorità sono collegate alla stessa routine di servizio, denominata *HighISR*. Non si ritiene opportuno approfondire le operazioni svolte da tale funzione, precisando soltanto che essa assegna il valore TRUE alla variabile booleana *PacketReceived*, il cui valore viene poi monitorato nella funzione *main*.

B.4 Funzione *main*

Il main è la funzione principale, e deve far parte di ogni progetto, poichè l'esecuzione inizia proprio da essa. Pertanto, tutte le chiamate alle altre funzioni devono essere riconducibili al main.

Come già precisato, i due progetti, di fatto, si differenziano proprio in questo blocco di codice. Il Listing B.3 riporta la parte del main comune ad entrambi i progetti, che contiene le istruzioni necessarie ad inizializzare le due schede.

Listing B.3: La parte iniziale della funzione main, comune ai due progetti.

```
1  void main(void){
2      BYTE index;
3      BYTE power;
4      BYTE i;
5      #if !defined(__18F4620)
6          char buf[32];
7      #endif
8
9      // decide if it is a hard reset or soft reset
10     #if defined(__18F4620) || defined(__18F87J11)
11         if((RCON & 0b00000011) != 0b00000011)
12     #endif
13     {
14         // hard reset, intialize the variables
15         UseExternal = FALSE;
16         ReceivingMode = FALSE;
17         RCON |= 0b00000011;
18     }
19     #if defined(__18F4620) || defined(__18F87J11)
20         else
21         {
22             // soft reset, keep them the way they were
23             RCON |= 0b00000011;
24         }
25     #endif
```

```
26
27      numPacketsRx.Val = 0;
28      PERnumFailed.Val = 0;
29      PERnum.Val = 0;
30      foundPERPacket = FALSE;
31      numRSSISamples = 1;
32      packetDelay = 1;
33      BoardInit();
34      ConsoleInit();
35      nl();
36      MRF24J40Init();
37      PrintHeader((ROM char *)"Main␣Menu");
38     ...
```

Dopo le dichiarazioni delle variabili di tipo *BYTE* index, power ed i, e dell'array di caratteri buf, si nota la presenza di una struttura if-else[5], che interessa il registro RCON. Quest'ultimo ha la funzione di memorizzare gli eventi di reset del dispositivo. Il PIC in questione dispone di numerose modalità di reset, che in questa sede non vengono analizzate nel dettaglio (si faccia riferimento al datasheet[16], Sezione 4). Le istruzioni presenti nel codice interrogano i primi due bit del registro RCON, che riguardano i casi di reset Power-On, cioè l'accensione del dispositivo, e reset Brown-Out, che si verifica quando Vdd scende sotto una soglia determinata: se uno di questi due eventi si è verificato, c'è stato un 'hard reset', e le variabili UseExternal e ReceivingMode vengono inizializzate con FALSE; in caso contrario, siamo in presenza di un 'soft reset', e tali variabili rimangono inalterate.

Superato il controllo sull'eventualità di reset, viene eseguita una serie di assegnazioni che interessano alcune variabili di tipo numerico o booleano, che riguardano principalmente la trasmissione dei pacchetti, dopodichè vengono chiamate le funzioni di inizializzazione della scheda, della console (che in questo caso non hanno effetto), e del modulo radio.

Nel seguito vengono commentate le parti del main che non coincidono nei due progetti, in quanto scritte per implementare, rispettivamente, le funzioni di trasmissione e ricezione.

Progetto my_radio_driver_send L'implementazione della trasmissione, a questo punto, è molto semplice. Il pin 4 del registro LATB viene settato al valore logico 1, perchè questo determina l'attivazione del relativo pull-up interno al processore: in questo modo, la pressione dello switch che collega questo pin a massa, comporta un cambiamento del valore logico ad esso associato, che può essere rilevato dal software. Di seguito, il processore passa all'esecuzione di un ciclo while infinito, nel quale la funzione *msDelay* determina un ritardo di 500 ms, ed il blocco if controlla lo stato del pin RB4, determinando l'invio di un pacchetto se lo switch è stato premuto. Il dato inviato è un byte il cui valore non ha interesse, in questa prima versione dell'applicazione: la semplice ricezione di un pacchetto, infatti, è sufficiente al ricevitore per sapere che il misuratore ha completato un ciclo. Il Listing B.4 riporta il codice descritto.

[5]Si ricorda che il processore in esame è il PIC18F4620, per cui la condizione delle direttive #if defined è verificata.

Listing B.4: Implementazione della funzione di trasmissione.

```
    ...
    LATB |= 0x10;
    while(1){
        msDelay(500);
        if(!RB4){
               RA0=!RA0;
               Send_Packet(0x11);
        }
    }
}
```

Progetto my_radio_driver_receive Il Listing B.5 illustra la parte della funzione main che realizza la funzione di ricezione. In primo luogo, il flag ReceivingMode viene impostato a TRUE, ed il pin RA0 della porta PORTA viene asserito, fatto che determina l'accensione del LED ad esso collegato. Segue un ciclo while con condizione sempre verificata, nel quale la ricezione di un pacchetto, segnalata dal valore TRUE della variabile booleana *PacketReceived*, determina la lettura delle aree di memoria del chip MRF24J40 contenenti il pacchetto stesso, per mezzo della funzione *PHYGetLongRAMAddr*. In questa prima, molto semplice, versione del firmware, in realtà non viene compiuta alcuna elaborazione: la ricezione del pacchetto determina semplicemente il cambiamento di stato del pin RA0, che sulla scheda di sviluppo corrisponde all'accensione o spegnimento del LED. Molte delle altre istruzioni presenti, come le chiamate alla funzione *PrintChar*, sono relative alla console, quindi non hanno effetto.

Listing B.5: Implementazione della funzione di ricezione.

```
    ...
    ReceivingMode = TRUE;
    RA0=1;
    while(1){
        if( ReceivingMode & PacketReceived )
          {
               BYTE i,j, RSSI_VAL;
               i = PHYGetLongRAMAddr(0x300);
               RSSI_VAL = PHYGetLongRAMAddr(0x300+i+2);
               PacketReceived = FALSE;
               if(RxPrintMenu++ == 20)
               {
                      c("\r\nRSSI␣|␣Packet␣Length␣|␣Packet␣Data\r\n\r\n");
                      RxPrintMenu = 0;
               }
                      PrintChar(RSSI_VAL);
               c("␣|␣");
                      PrintChar(i);   //print the packet length
               c("␣|␣");
                        for(j=1;j<=i;j++)
                   {
                       PrintChar(PHYGetLongRAMAddr(0x300 + j));
                       //read out the rest of the buffer
                   }
                RA0=!RA0;
```

```
26                      c("\r\n");
27          }
28  }
```

Bibliografia

[1] Tommaso Caldognetto. "Monitoraggio dei consumi di energia elettrica dei poli didattici DEI e DIE mediante architettura Pin-Energy". Tesi di Laurea per il Corso di Ingegneria Elettronica. 2010.

[2] ENEL. *Le Smart Grids*. 2011. URL: http://www.enel.it/it-IT/reti/enel_distribuzione/qualita/progetti_smart_grids/.

[3] Andy Grove. *Our Electric Future*. 2008. URL: http://www.american.com/archive/2008/july-august-magazine-contents/our-electric-future.

[4] IME. *Nemo D4-L*. 2010. URL: http://www.imeitaly.com/docs/NT604.pdf.

[5] IME. *TAI233, Trasformatore di misura per reti a bassa tensione*. 2009. URL: http://www.imeitaly.com/docs/NT521.pdf.

[6] Davis Instruments. *Vantage Vue Integrated Sensor Suite Installation Manual*. 2011. URL: http://www.vantagevue.com/product_documents/weather/manuals/07395-262_IM_06357.pdf.

[7] Davis Instruments. *Vantage Vue Wireless Console Manual*. 2011. URL: http://www.vantagevue.com/product_documents/weather/manuals/07395-261_IM_06351.pdf.

[8] Davis Instruments. *WeatherLinkIP for Vantage Stations*. 2011. URL: http://www.vantagevue.com/product_documents/weather/spec_sheets/6555_spec_WeatherLinkIP.pdf.

[9] Jennic. *Data Sheet: JN5148-001 IEEE802.15.4 Wireless Microcontroller*. 2010. URL: http://www.jennic.com/files/support_files/JN-DS-JN5148-1v6.pdf.

[10] Jennic. *Data Sheet: JN5148-001-Myy JenNet, ZigBee PRO and IEEE802.15.4 Module*. 2010. URL: http://www.jennic.com/files/support_files/JN-DS-JN5148MO-1v4.pdf.

[11] Jennic. *IEEE 802.15.4 Application Development Reference Manual*. 2010. URL: http://www.jennic.com/files/support_files/JN-RM-2024-IEEE802.15.4-App-Dev-2v0.pdf.

[12] Jennic. *JN5148 Software Developer's Kit Installation and User Guide*. 2010. URL: http://www.jennic.com/files/support_files/JN-UG-3064-JN5148-SDK.pdf.

[13] Microchip. *MPLAB ICD2 In-Circuit Debugger User's Guide*. 2007. URL: http://ww1.microchip.com/downloads/en/devicedoc/51331b.pdf.

[14] Microchip. *MPLAB User's Guide*. 2006. URL: http://ww1.microchip.com/downloads/en/devicedoc/51519a.pdf.

[15] Microchip. *MRF24J40MC Data Sheet*. 2011. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/75002A.pdf.

[16] Microchip. *PIC18F2525/2620/4525/4620 Data Sheet*. 2008. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/39626e.pdf.

[17] Microchip. *PICDEM^{TM} Z Demonstration Kit User's Guide*. 2008. URL: http://ww1.microchip.com/downloads/en/AppNotes/PICDEM%20Z%20Users_Guide_51524C.pdf.

[18] M. Zorzi N. Benvenuto. *Principles of Communications Networks and Systems*. John Wiley & Sons Ltd., 2010.

[19] Jr. Warren D. Devine. *From Shafts to Wires: Historical Perspective on Electrification*. 1983. URL: http://www.j-bradford-delong.net/teaching_folder/Econ_210c_spring_2002/Readings/Devine.pdf.

Ringraziamenti

Desidero ringraziare anzitutto i miei genitori, per l'incoraggiamento e per avermi permesso di frequentare questo Corso di Laurea, il professor Paolo Tenti per la disponibilità dimostrata e Salvatore Brundo per avermi aiutato durante la fase di programmazione dei microcontrollori. Un ringraziamento particolare va a Marco Stellini e Tommaso Caldognetto per avermi costantemente assistito ed aiutato nella realizzazione di questo lavoro.