

M.M.Electronics - <http://www.mmetft.it>



Michele Marino - [mmelectronics@mmetft.it](mailto:mmelectronics@mmetft.it)

## Introduzione al protocollo wireless *ZigBee<sup>TM</sup>*

V 0.1

Settembre 2007

## INFORMATIVA

Come prescritto dall'art. 1, comma 1, della legge 21 maggio 2004 n.128, l'autore avvisa di aver assolto, per la seguente opera dell'ingegno, a tutti gli obblighi della legge 22 Aprile del 1941 n. 633, sulla tutela del diritto d'autore. Tutti i diritti di questa opera sono riservati. Ogni riproduzione ed ogni altra forma di diffusione al pubblico dell'opera, o parte di essa, senza un'autorizzazione scritta dell'autore, rappresenta una violazione della legge che tutela il diritto d'autore, in particolare non ne è consentito un utilizzo per trarne profitto. La mancata osservanza della legge 22 Aprile del 1941 n. 633 è perseguibile con la reclusione o sanzione pecuniaria, come descritto al Titolo III, Capo III, Sezione II. A norma dell'art. 70 è comunque consentito, per scopi di critica o discussione, il riassunto e la citazione, accompagnati dalla menzione del titolo dell'opera e dal nome dell'autore.

## AVVERTENZE

Chiunque decida di far uso delle nozioni riportate nella seguente opera o decida di realizzare i circuiti proposti, è tenuto pertanto a prestare la massima attenzione in osservanza alle normative in vigore sulla sicurezza.

L'autore declina ogni responsabilità per eventuali danni causati a persone, animali o cose derivante dall'utilizzo diretto o indiretto del materiale, dei dispositivi o del software presentati nella seguente opera.

Si fa inoltre presente che quanto riportato viene fornito così com'è, a solo scopo didattico e formativo, senza garanzia alcuna della sua correttezza.

L'autore ringrazia anticipatamente per la segnalazione di ogni errore.

## Indice

<b>1</b>	<b>Introduzione al protocollo ZigBee</b>	<b>5</b>
<b>2</b>	<b>Lo standard ZigBee</b>	<b>5</b>
2.1	Tipi di dispositivi . . . . .	6
<b>3</b>	<b>Configurazioni di rete</b>	<b>6</b>
3.1	Configurazione di rete a stella . . . . .	6
3.2	Configurazione di rete ad albero . . . . .	6
3.3	Reti a maglia . . . . .	7
<b>4</b>	<b>Terminologia del protocollo ZigBee</b>	<b>8</b>
4.1	Tipo di messaggi e binding . . . . .	8
4.2	Formato messaggi del protocollo ZigBee . . . . .	9
4.3	Formato frame del protocollo ZigBee . . . . .	9
4.4	Frame KVP . . . . .	9
4.5	Frame MSG . . . . .	10
<b>5</b>	<b>Indirizzamento di rete</b>	<b>10</b>
5.1	Estensione unica identificatori IEEE - EUI-64 . . . . .	10
<b>6</b>	<b>Indirizzamento di rete</b>	<b>10</b>
6.1	Messaggi unicast . . . . .	10
6.2	Messaggi broadcast . . . . .	11
<b>7</b>	<b>Meccanismo di trasmissione dati</b>	<b>11</b>
7.1	Routing . . . . .	11
<b>8</b>	<b>Associazione di rete</b>	<b>12</b>
<b>9</b>	<b>La stack Microchip</b>	<b>13</b>
9.1	Composizione di un nodo hardware . . . . .	13
9.2	L'analizzatore di reti wireless . . . . .	13
9.3	Struttura della stack . . . . .	13
9.4	Primo esempio applicativo . . . . .	14
9.4.1	Binding sui dispositivi End . . . . .	15
9.4.2	Ricerca di dispositivi . . . . .	15
9.4.3	Invio dei messaggi . . . . .	16
9.5	Utilizzo della stack Microchip . . . . .	16
9.6	Interfacciamento con la stack . . . . .	17
9.7	Creazione e accesso ad una rete . . . . .	17
9.8	Ricezione dei messaggi . . . . .	18
9.9	Invio dei messaggi . . . . .	18
9.10	Richiesta e ricezione dati da un RFD . . . . .	19
9.11	Trasmissioni sicure . . . . .	20
	<b>Bibliografia</b>	<b>23</b>

## Elenco delle figure

1	Rete a stella . . . . .	6
2	Rete ad albero . . . . .	7
3	Rete a maglia . . . . .	7
4	Struttura di un nodo hardware . . . . .	13
5	Menu apertura progetto . . . . .	15
6	Menu compilazione progetto . . . . .	15
7	Struttura base dell'applicazione . . . . .	17
8	Codice ricezione messaggio . . . . .	19
9	Codice trasmissione messaggio . . . . .	20
10	Codice ricezione messaggi RFD . . . . .	21
11	Servizi di sicurezza del ZigBee . . . . .	22

## Elenco delle tabelle

1	Frequenze protocollo ZigBee . . . . .	5
2	Tipi di dispositivi IEEE 802.15.4 (FFD - RFD) . . . . .	7
3	Tipi di dispositivi del protocollo ZigBee . . . . .	7
4	Risorse richieste al PIC18F . . . . .	14

## 1 Introduzione al protocollo ZigBee

ZigBee è un protocollo per reti wireless progettato specificatamente per bassi trasferimenti di dati e per il controllo di dispositivi e sensori su reti wireless. Le applicazioni del protocollo ZigBee sono varie: dalle reti di manifattura automatica, ai sistemi di sicurezza per la casa, ai sistemi di controllo industriale, misure remote e periferiche per personal computer. Comparato con altri protocolli wireless, il protocollo di comunicazione wireless ZigBee ha essenzialmente, tre vantaggi:

- bassa complessità
- richiesta ridotta di risorse
- possiede un insieme di specifiche standard.

Inoltre, ZigBee offre tre bande di frequenze di funzionamento insieme ad un certo numero di configurazioni di rete e caratteristiche di sicurezza opzionali.

## 2 Lo standard ZigBee

Dunque lo ZigBee è un protocollo di comunicazione wireless standard progettato per il controllo su reti a basso trasferimento di dati. ZigBee si basa sulle specifiche IEEE 802.15.4 e fornisce metodologie standard per funzioni, includendo la possibilità di creare reti, messaggi e ricerca di dispositivi su di essa. Dunque lo ZigBee usa le specifiche IEEE 802.15.4 come Medium Access Layer (MAC) e come Physical Layer (PHY). Lo standard definisce tre bande di frequenza di funzionamento: 2.4GHz, 915MHz e 868MHz. Ogni banda di frequenza offre un numero prefissato di canali.

banda	canali	numerazione
2.4GHz	16	11-26
915MHz	10	1-10
868MHz	1	0

Tabella 1: Frequenze protocollo ZigBee

La velocità di trasferimento del protocollo ZigBee dipende dalla frequenza operativa. La banda a 2.4GHz consente un trasferimento massimo di 250Kbps, quella a 915 MHz 40Kbps e quella a 868MHz 20 Kbps. La velocità di trasferimento corrente potrebbe essere inferiore a quelle specificate. Questo è dovuto essenzialmente all'aggiunta iniziale di pacchetti di controllo e al ritardo di processamento dei primi dispositivi in commercio.

La lunghezza massima di un pacchetto IEEE 802.15.4 è di 127 byte, il quale include anche 16-bit per il controllo d'errore (CRC - Cyclic Redundancy Check). Il valore CRC a 16-bit verifica l'integrità del frame. In aggiunta, lo standard IEEE 802.15.4 usa (opzionale) un meccanismo di trasferimento a risposta dei dispositivi di comunicazione (Acknowledged). Con questo metodo, tutti i frame posseggono un bit di richiesta (ACK flag) che viene letto dai dispositivi in ricezione. Se il frame viene trasmesso con il bit ACK settato e non viene ricevuta nessuna risposta dal dispositivo di ricezione entro un certo intervallo di tempo, la trasmissione viene ripetuta per un certo numero di volte prima di segnalare un errore di trasmissione. E' importante notare che la ricezione del segnale di Acknowledgement indica semplicemente che il frame è stato ricevuto correttamente a livello MAC. Chiaramente questo non indica allo stesso tempo, che il frame è stato processato

correttamente. E' possibile infatti che a livello MAC il nodo di ricezione, riceva e trasmetta una risposta corretta (Acknowledgement), ma data la mancanza di risorse di processamento, un frame possa essere scartato al livello superiore. Questo porterebbe ad assumere un'altra risposta di ricezione per il layer superiore, ciò che porta, inevitabilmente ad un overhead di processamento.

## 2.1 Tipi di dispositivi

Lo standard IEEE 802.15.4 definisce due tipi di dispositivi come mostrato nella tabella 2.

La tabella 3 mostra invece tre tipi di dispositivi del protocollo ZigBee in relazione anche ai tipi di dispositivi definiti dall'IEEE.

Il dispositivo Coordinator in genere è unico per ogni rete ed è il dispositivo che forma la rete, allocando gli indirizzi di rete e tenendo memoria della tabella di allocazione (binding table) con l'associazione degli indirizzi dei vari dispositivi di rete. Il dispositivo Router è opzionale e consente di estendere il range della rete consentendo a più nodi di comunicare tra loro. Questo può eseguire anche funzioni di monitoraggio e/o controllo come il dispositivo End.

## 3 Configurazioni di rete

Il protocollo di rete wireless ZigBee può assumere diversi tipi di configurazione. In tutte le configurazioni di rete, ci sono almeno due dispositivi fondamentali:

- nodo Coordinator
- dispositivo End

Il Coordinator del protocollo ZigBee è una variante speciale del dispositivo FFD

che implementa un insieme piuttosto vasto di servizi del protocollo. Un dispositivo End può essere un FFD o anche un RFD. Un dispositivo RFD rappresenta il più piccolo e semplice nodo del protocollo ZigBee. Questo implementa solo un insieme minimo di servizi. Un terzo componente opzionale, il Router, si ritrova in alcune configurazioni di reti ZigBee.

### 3.1 Configurazione di rete a stella

La configurazione di rete a stella (figura 1) consiste di un dispositivo Coordinator e uno o più dispositivi End. In una rete del genere, tutti i dispositivi End presenti nella rete comunicano soltanto con il nodo Coordinator. Se un dispositivo End deve trasferire dati ad un altro dispositivo End, trasferisce i dati al Coordinator e poi quest'ultimo inoltra i dati al dispositivo End di destinazione.

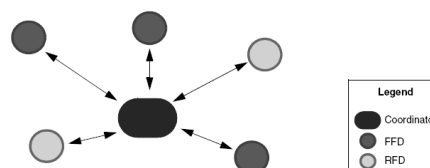


Figura 1: Rete a stella

### 3.2 Configurazione di rete ad albero

Un altro tipo di configurazione di rete è rappresentato dalla tipologia ad albero (figura 2). In questa configurazione, i dispositivi End possono entrare nella rete sia attraverso il nodo Coordinator che attraverso i nodi Router. I nodi Router, in questo caso, svolgono due funzioni: la prima riguarda l'estensione del numero di nodi che possono entrare a far parte del-

Dispositivo	Servizi	Alimentazione	Conf. ricevitore
Full Function Device	molti o tutti	principale	On quando inattivo
Reduced Function Device	limitati	batteria	Off quando inattivo

Tabella 2: Tipi di dispositivi IEEE 802.15.4 (FFD - RFD)

Dispositivo ZigBee	Dispositivo IEEE
Coordinator	FFD
Router	FFD
End	FFD o RFD

Tabella 3: Tipi di dispositivi del protocollo ZigBee

la rete; la seconda riguarda l'estensione fisica del range della rete.

Con l'aggiunta di un Router, un dispositivo End, non necessita di essere nel range radio del nodo Coordinator per comunicare nella rete. Tutti i messaggi in una rete ad albero vengono smistati lungo l'albero di trasmissione.

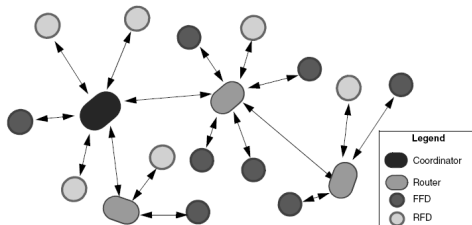


Figura 2: Rete ad albero

### 3.3 Reti a maglia

La configurazione di rete a maglia (figura 3) è simile a quella ad albero, ad eccezione del fatto che in questa configurazione i dispositivi FFD possono smistare messaggi direttamente ad altri FFD senza seguire la struttura ad albero. I messaggi verso i dispositivi RFD passano attraverso

so i nodi padri (Router). Il vantaggio di questa topologia consiste nella riduzione della latenza legata alla trasmissione dei messaggi e ad un incremento di affidabilità.

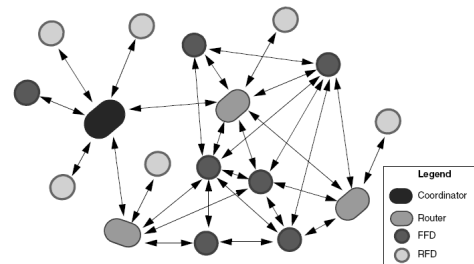


Figura 3: Rete a maglia

Le configurazioni ad albero e a maglia sono anche conosciute come reti multi-hop, date le loro abilità di smistamento dei pacchetti verso dispositivi multipli, mentre la rete a stella viene anche detta rete single-hop. Una rete basata sul protocollo ZigBee è una rete ad accesso multiplo, nel senso che tutti i nodi hanno accesso al mezzo di comunicazione con lo stesso peso. Ci sono due tipi di meccanismi ad accesso multiplo, denominati beacon e non-beacon. In rete con mecca-

nismo non-beacon, tutti i nodi della rete possono trasmettere in qualunque istante non appena il canale di trasmissione è libero o inattivo. Con il meccanismo beacon invece, i nodi della rete possono trasmettere solo in istanti di tempo predefiniti. Il nodo Coordinator, periodicamente, inizializza la rete con un superframe identificato come beacon frame e tutti i nodi della rete si sincronizzano con questo superframe. Ad ogni nodo viene associato uno specifico slot temporale del superframe durante il quale è abilitato a ricevere e trasmettere dati. Il superframe può contenere anche uno slot comune durante il quale tutti i nodi possono accedere al canale di comunicazione.

#### 4 Terminologia del protocollo ZigBee

Nel protocollo ZigBee si definiscono i profili dei dispositivi che non sono altro che delle semplici descrizioni dei dispositivi logici e delle relative interfacce associate. In generale non c'è un codice associato ad un profilo. Ogni blocco di dati che può essere passato tra dispositivi (per esempio lo stato di uno switch o la lettura di un potenziometro) viene chiamato attributo. Ogni attributo viene assegnato ad un identificatore che è unico per ogni dispositivo. Gli attributi vengono raggruppati in cluster. Ad ogni cluster viene associato un identificatore unico. Le interfacce vengono specificate a livello di cluster e non a livello di attributi, anche se gli attributi vengono poi trasferiti individualmente.

Il profilo definisce il valore degli identificatori (ID) degli attributi e dei cluster, così come il formato di ogni attributo. Per esempio, in un controllo domesti-

co di una lampada, profilo Lampada, il cluster OnOffDRC del dispositivo di controllo di luminosità remoto (Dimmer Remote Control) contiene un solo attributo, OnOff, il quale può avere un valore ad 8-bit senza segno, nel quale il valore 0xFF significa ON, il valore 0x00 significa OFF e il valore 0xF0 significa uscita bistabile.

Il profilo descrive anche quali cluster sono obbligatori e quali opzionali per ogni dispositivo. In aggiunta, il profilo potrebbe definire alcuni servizi del protocollo ZigBee come obbligatori.

In base a queste definizioni è possibile scrivere il codice relativo all'applicazione specifica. E' infatti possibile scrivere il codice nel modo che fa più comodo, raggruppando le funzioni e gli attributi in base alle definizioni date nel profilo. In questo modo è possibile avere una certa compatibilità anche tra diversi dispositivi costruiti da diversi produttori.

##### 4.1 Tipo di messaggi e binding

I dispositivi possono comunicare con altri dispositivi presenti nella rete se sono a conoscenza dei relativi indirizzi di rete. Questo tipo di messaggi vengono definiti come messaggi diretti. Comunque, i messaggi diretti portano ad un eccessivo overhead dovuto essenzialmente alla ricerca e al mantenimento di tutti gli indirizzi di destinazione. Il protocollo ZigBee offre una caratteristica che semplifica la comunicazione, nota come binding. Il Coordinator del protocollo ZigBee crea una tabella di accoppiamenti degli indirizzi a livello cluster/endpoint tra i servizi e le richieste dei dispositivi nella rete. Ogni coppia viene chiamata binding. Un binding può essere richiesto dal dispositivo stesso, oppure può essere creato dal Coordinator o da un altro dispositivo.

Una volta creato il binding, due dispositivi possono comunicare tra loro attraverso il nodo Coordinator. Il dispositivo sorgente invia il messaggio al Coordinator, il quale replica il messaggio verso uno o più dispositivi di destinazione. Questo tipo di messaggi vengono definiti come messaggi indiretti.

#### 4.2 Formato messaggi del protocollo ZigBee

Ogni messaggio del protocollo ZigBee consiste di 127 byte suddivisi tra i seguenti campi:

- **MAC Header** - L'intestazione del MAC contiene il campo di controllo frame (Frame Control), il Beacon Sequence Number (BSN) e le informazioni di indirizzamento del messaggio che viene trasmesso. E' importante notare che il MAC Header potrebbe non includere la sorgente o la destinazione finale del messaggio se il messaggio stesso deve passare per un nodo Router. La generazione e l'utilizzo di questo header è trasparente al codice applicativo.
- **Network Layer Header (NWH)** - Questo header contiene, insieme ad altre informazioni, la sorgente attuale e la destinazione finale del messaggio. La generazione e l'uso di questo header è trasparente al codice applicativo.
- **Application Support Sub-Layer Header (APS)** - Questo header include l'identificatore del profilo, l'ID del cluster e il punto finale di destinazione (endpoint) del messaggio corrente. La generazione e l'utilizzo di questo header è trasparente ai fini del codice applicativo.

- **APS Payload** - Questo campo contiene il frame del protocollo ZigBee per l'applicazione del processo. Il codice applicativo è responsabile del contenuto dell'APS Payload.

#### 4.3 Formato frame del protocollo ZigBee

Il protocollo ZigBee definisce due formati di frame: il formato Key Value Pair (KVP) e il formato Message (MSG). Entrambi i frame sono associati con un ID cluster, ma i frame KVP sono pensati per permettere il trasferimento di alcune informazioni associate ad un attributo usando una particolare struttura, mentre i frame MSG trasferiscono informazioni utilizzando una forma strutturale libera. Il profilo dell'applicazione che si va a progettare specificherà il formato del frame che dovrebbe essere usato per trasferire le informazioni e il formato di ogni frame MSG. Date le differenze nel formato del frame, un cluster non può utilizzare entrambi i formati KVP ed MSG.

#### 4.4 Frame KVP

Un frame KVP contiene le seguenti informazioni:

1. Conteggio transazione
2. Tipo frame
3. Transazione
  - numero sequenza transazione
  - tipo comando e tipo dati attributo
  - identificatore attributo
  - codice errore (opzionale)
  - dato attributo (dimensione variabile)

Il tipo di comando indica cosa si è supposto di fare con l'applicazione progettata sulle informazioni manipolate. Per esempio, il comando "Set", richiede di settare il valore dell'attributo indicato dal relativo ID al valore indicato dal dato attributo, e il comando, "Ottieni con Acknowledge", richiede di inviare il valore dell'attributo fornito dall'identificatore dell'attributo stesso.

#### 4.5 Frame MSG

Un frame MSG contiene le seguenti informazioni:

1. Conteggio transazione
2. Tipo frame
3. Transazione
  - numero sequenza transazione
  - lunghezza transazione
  - dati transazione

Sia il dispositivo di trasmissione che quello di ricezione devono conoscere il formato dei dati in transazione.

### 5 Indirizzamento di rete

Ogni nodo nella rete ZigBee ha due indirizzi: un indirizzo MAC a 64-bit e un indirizzo di rete a 16-bit. Oltre a questo, ci sono altre due forme di indirizzamento messaggi disponibili.

#### 5.1 Estensione unica identificatori IEEE - EUI-64

Qualunque dispositivo che comunica utilizzando il protocollo ZigBee deve avere un indirizzo globale unico ovvero, l'indirizzo MAC a 64-bit. Questo indirizzo è costituito da un campo che identifica

in modo univoco l'organizzazione (Organizationally Unique Identifier - OUI) da 24-bit più 40-bit assegnati direttamente dal produttore del dispositivo ZigBee. L'indirizzo OUI è acquistabile dall'IEEE al fine di assicurarne l'unicità. E' possibile ottenere il proprio indirizzo OUI facendone richiesta al seguente indirizzo:

<https://standards.ieee.org/regauth/oui/forms/OUT-form.shtml>

E' evidente che se si è già in possesso di un indirizzo OUI per applicazioni Ethernet, è possibile utilizzare lo stesso OUI anche per le applicazioni che fanno uso del protocollo ZigBee.

### 6 Indirizzamento di rete

I dispositivi utilizzano il loro indirizzo per esteso al fine di comunicare mentre risultano coinvolti nel processo di accesso alla rete. Quando un dispositivo accede con successo ad una rete ZigBee, riceve un indirizzo di rete a 16-bit che viene poi utilizzato per comunicare con gli altri dispositivi presenti nella rete.

#### 6.1 Messaggi unicast

In un messaggio unicast, l'indirizzo del nodo di destinazione viene fornito nell'intestazione a livello MAC del pacchetto di trasmissione. I dispositivi che possiedono quell'indirizzo saranno i soli a ricevere i messaggi.

#### 6.2 Messaggi broadcast

In un pacchetto broadcast, l'indirizzo di destinazione a livello MAC, è 0xFFFF. Ogni ricetrasmittitore abilitato alla ricezione riceverà il messaggio. Questa forma di indirizzamento viene utilizzata

quando si esegue l'accesso alla rete, al fine di ricercare percorsi di comunicazione nella rete e per eseguire altre funzioni di ricerca sul protocollo ZigBee. Il protocollo ZigBee implementa un meccanismo di conferma passivo per i pacchetti broadcast. Questo significa che quando un dispositivo dà origine o ritrasmette un pacchetto broadcast, resta in ascolto di tutti i dispositivi ad esso adiacenti al fine di tramettere nuovamente il pacchetto. Se tutti i dispositivi adiacenti non replicano il messaggio entro un certo intervallo di tempo (`nwkPassiveAckTimeout`), il pacchetto verrà trasmesso nuovamente fino a quando non sentirà la ritrasmissione da tutti i dispositivi adiacenti o fino a quando non si supera il massimo tempo di trasmissione (`nwkNetworkBroadcastDeliveryTime`).

## 7 Meccanismo di trasmissione dati

In una rete tipo non-beacon, quando un dispositivo vuole trasmettere un frame dati, semplicemente resta in attesa fino a quando il canale diventa inattivo. Una volta verificato che il canale di comunicazione è libero, il dispositivo può trasmettere il frame.

Se il dispositivo di destinazione è un FFD il relativo ricetrasmittitore è sempre attivo, e gli altri dispositivi possono trasmettere verso di lui in qualunque istante. Questa capacità consente la creazione delle reti con topologia a maglia. Diversamente, se il dispositivo è un RFD, potrebbe andare in modalità risparmio energetico e quindi disabilitare il proprio ricetrasmittitore. Il dispositivo RFD non sarà quindi in grado di ricevere messaggi in questo stato. Questa situazione viene gestita chiedendo che tut-

ti i messaggi inviati e ricevuti dal dispositivo RFD passino dal dispositivo FFD padre. Quando il dispositivo RFD alimenterà il proprio ricetrasmittitore, richiederà i propri messaggi al dispositivo FFD che lo contiene. Se il nodo padre ha memorizzato un messaggio per il nodo figlio, lo inoltrerà non appena il nodo figlio diventa attivo. Questo consente al dispositivo RFD di risparmiare energia, ma richiede allo stesso tempo al dispositivo FFD di avere più memoria RAM al fine di conservare i messaggi per tutti i nodi figli. Se un nodo figlio non richiede il messaggio per un certo intervallo di tempo (`macTransactionPersistenceTime`), il messaggio andrà in time out e il nodo padre lo eliminerà dalla memoria.

### 7.1 Routing

Il routing consente di espandere la rete consentendo ad un dispositivo di entrare nella rete anche se questo è posizionato in una regione oltre la copertura radio del Coordinator del protocollo ZigBee.

Il tipo di routing desiderato per un messaggio è indicato nel momento in cui viene inviato il messaggio. Esistono tre opzioni di routing:

- **SUPPRESS** - Se in una rete a maglia viene trovato un router, il messaggio viene trasferito attraverso quel router. In caso contrario, il messaggio viene trasferito attraverso l'albero della rete.
- **ENABLE** - Se in una rete viene trovato un router, il messaggio viene trasferito attraverso quel router. Se non viene trovato nessun nodo di smistamento dati, il router può avviare una ricerca per trovare un percorso di smistamento dati. Quando

la ricerca viene completata, il messaggio verrà inviato lungo il percorso calcolato. Se il router non è in grado di smistare i dati, invierà il messaggio lungo l'albero della rete.

- **FORCE** - Se il router è in grado di cercare percorsi di smistamento dati nella rete, avvierà la ricerca di un percorso anche se è già esistente un percorso valido. Quando la ricerca viene portata a termine, il messaggio verrà trasferito lungo il nuovo percorso calcolato. Se il router non ha la possibilità di trasferire il messaggio, questo sarà trasferito sull'albero delle rete ZigBee. Questa opzione dovrebbe essere usata con cautela in quanto genera molto traffico sulla rete. In genere viene utilizzata per riparare un router fuori uso.

## 8 Associazione di rete

La rete basata sul protocollo ZigBee viene inizializzata dal Coordinator. Alla partenza, il Coordinator esegue la ricerca di altri Coordinator di rete ZigBee nel proprio campo di azione. In base all'energia presente nel canale e al numero di reti trovate su ogni canale disponibile, il Coordinator stabilisce la propria rete e seleziona un identificatore PAN ID a 16-bit unico. Una volta creata la nuova rete, i router e i dispositivi del protocollo ZigBee possono entrare nella rete.

Formata la rete, è possibile che a causa di cambiamenti fisici, altre reti possano sovrapporsi creando un conflitto sul PAN ID. In questa situazione, un nodo Coordinator potrebbe avviare una procedura per la risoluzione del conflitto sui PAN ID che porta uno dei Coordinator a cambiare PAN ID e/o canale. Il Coor-

dinator che esegue il cambiamento deve comunicare la modifica a tutti i suoi dispositivi figli in modo che questi possano adeguarsi alla nuova configurazione.

I dispositivi del protocollo ZigBee memorizzano informazioni sugli altri nodi della rete, includendo nodi padri e figli, in un'area di memoria non-volatile chiamata "neighbor table". All'accensione, se un dispositivo figlio determina che faceva parte di una rete attraverso la "neighbor table", potrebbe eseguire una procedura di notifica al fine di localizzarsi nuovamente all'interno della rete. I dispositivi che ricevono la notifica faranno una ricerca nella propria "neighbor table" e verificano se il dispositivo che ne ha fatto richiesta è uno dei figli. Se è così, il dispositivo padre informerà il dispositivo figlio del riconoscimento all'interno della rete. Se la richiesta di notifica fallisce oppure il dispositivo figlio non è presente nella "neighbor table", allora tenterà di entrare nella rete come nuovo dispositivo. Il dispositivo genererà quindi una lista di dispositivi padri potenziali e cercherà di entrare in una rete esistente alla massima profondità.

Una volta entrato in rete, un dispositivo può dissociarsi dalla rete per via di un messaggio di abbandono da parte del dispositivo padre oppure da una richiesta di dissociazione del dispositivo stesso.

Il tempo richiesto da un dispositivo per la determinazione dell'energia del canale e delle reti disponibili in ogni canale è specificato da un parametro presente nel firmware per processori Microchip (ScanDuration). Per la banda di frequenza a 2.4GHz, il tempo di scansione in secondi viene calcolato in base alla seguente relazione:

$$0.01536 \cdot (2^{\text{ScanDuration}} + 1) \quad (1)$$

I dispositivi Router e i dispositivi End del protocollo ZigBee eseguono una sola scansione per determinare le reti disponibili, ma il dispositivo Coordinator ne esegue due, la prima delle quale al fine di determinare l'energia nel canale e la seconda per determinare le reti disponibili. La specifica della durata della scansione definisce quindi il tempo di start-up del sistema complessivo.

## 9 La stack Microchip

Microchip mette a disposizione una stack scritta in C da utilizzare con i propri PIC. Questa permette di implementare e utilizzare in modo semplice e intuitivo il protocollo ZigBee.

### 9.1 Composizione di un nodo hardware

Per creare un nodo del protocollo ZigBee è necessario avere a disposizione i seguenti componenti:

- microcontrollore Microchip con interfaccia SPI
- ricetrasmittitore RF Microchip MFR24J40
- antenna a monopolo o a microstriscia

Come mostra la figura 4, il microcontrollore (master) si connette al modulo RF (slave) mediante il bus SPI e alcuni segnali di controllo. Le risorse hardware richieste dal PIC18F per l'implementazione della stack sono riportate nella tabella 4.

Chiaramente i pin RC0, RC1, RC2 possono essere cambiati a piacere andando a modificare le relative procedure della stack. Il flusso di progetto permette

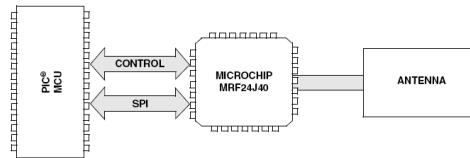


Figura 4: Struttura di un nodo hardware

di utilizzare sia un'antenna a bipolo che un'antenna a microstriscia. La tensione di alimentazione è di 3.3V, che può essere fornita anche mediante una batteria. Tipicamente, sia i Coordinator che i Router vengono alimentati dall'alimentazione principale, mentre i dispositivi End possono funzionare a batteria.

### 9.2 L'analizzatore di reti wireless

Insieme al codice sorgente, la Microchip fornisce anche un software per analizzare le reti wireless chiamato ZENA. Il tool consente inoltre di creare file sorgenti personalizzati per l'applicazione specifica. La versione dimostrativa non permette l'analisi in tempo reale delle reti ma solo di visualizzare il traffico memorizzato in precedenza. Il pacchetto completo della stack si può scaricare e installare liberamente dal sito della Microchip.

### 9.3 Struttura della stack

I file sorgenti utilizzati dalla stack sono numerosi e al fine di mantenere un certo ordine e compatibilità, i file che sono comuni ad altre applicazioni vengono memorizzati in una directory diversa da quelli necessari all'implementazione del protocollo ZigBee. I file relativi alla stack contengono tutta la logica di supporto per tutti i tipi di applicazione del protocollo ZigBee. E' evidente che, in

Risorsa	Descrizione
INT0	accetta interrupt provenienti dal ricetrasmittitore MRF24J40
TMR0	temporizzazione
RC0	selezione chip
RC1	regolatore tensione/wake-up
RC2	reset ricetrasmittitore
RC3	SPI SCK
RC4	SPI SDI
RC5	SPI SDO

Tabella 4: Risorse richieste al PIC18F

base all'applicazione che si va a progettare, verrà utilizzata solo una parte della logica a disposizione. Questo viene stabilito andando a modificare le definizioni nel file *zigbee.def*. Chiaramente ogni applicazione avrà il suo file di definizione. Questo approccio consente di utilizzare file sorgenti comuni per applicazioni diverse e generare il relativo codice macchina facendo riferimento solo al file di definizione.

#### 9.4 Primo esempio applicativo

Per poter implementare il protocollo ZigBee è necessario avere a disposizione il tool di sviluppo software MPLAB, scaricabile gratuitamente dal sito della Microchip. Lo sviluppo della prima applicazione di esempio riguarda i seguenti passi:

- assicurarsi di aver installato correttamente il pacchetto relativo alla stack Microchip
- lanciare il tool MPLAB IDE e aprire il progetto situato in *\*\DemoCoordinator\DemoCoordinator.mcp* per l'applicazione relativa al Coordinator, *\*\ DemoRFD\DemoRFD.mcp* per

l'applicazione relativa ad un RFD, *\*\DemoRouter\DemoRouter.mcp* per l'applicazione relativa all'implementazione di un Router<sup>1</sup> (figura 5)

- impostare tutte le opzioni relative al microcontrollore in uso (tipo oscillatore, configurazione WatchDog, BOREN, ecc.)
- compilare il progetto usando il comando *build*<sup>2</sup> (figura 6)
- se la compilazione non viene portata a termine, controllare le impostazioni di compilazione e le opzioni di progetto
- programmare il microcontrollore con il file *.hex* generato dalla compilazione (per esempio *DemoCoordinator.hex*)

Come impostazione predefinita ai nodi End sono disabilitati i servizi di binding

<sup>1</sup>L'asterisco indica il percorso scelto durante l'installazione del pacchetto. Nel seguito del discorso, tale indicazione verrà omessa.

<sup>2</sup> Assicurarsi che i file sorgenti si trovino nella directory *MpZBee* posizionata nella radice del disco rigido.

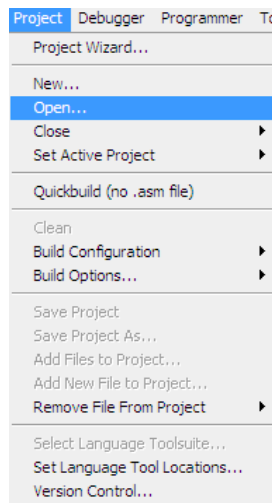


Figura 5: Menu apertuta progetto

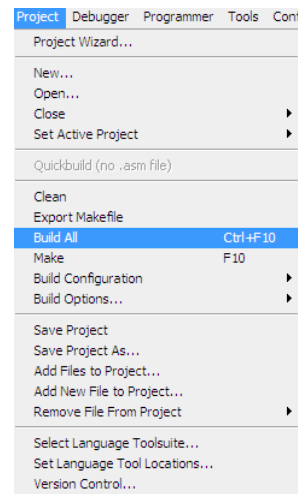


Figura 6: Menu compilazione progetto

e di ricerca. Chiaramente i servizi possono essere abilitati eliminando il commento alle relative stringhe di definizione nel file *.def* (nelle applicazioni dimostrative presentate, si tratta del file *zigbee.def*).

#### 9.4.1 Binding sui dispositivi End

Se entrambi i nodi hanno il binding abilitato (`#define USE_BINDING` non commentato), un nodo<sup>3</sup> switch manda il suo messaggio a uno o più nodi light attraverso il meccanismo del binding e della messagistica indiretta. Prima di inviare un messaggio indiretto è necessario creare un binding tra il nodo switch e il nodo light. La gestione del binding viene effettuata attraverso un pulsante sulla linea RB5. Premendo il pulsante sul nodo Coordinator si avvia la richiesta di binding al dispositivo End. Premendo il pulsante sul nodo RFD (End) si ottiene il binding richiesto. Nel caso in cui non ve-

<sup>3</sup>Con i nomi switch e light ci si riferisce nuovamente all'esempio del controllo di luminosità di una lampada.

nisse premuto il pulsante sul nodo RFD entro circa 5 secondi, la richiesta di binding da parte del nodo Coordinator va in time out e il processo di associazione deve essere ripetuto. Eseguito il binding, il nodo switch può ora inviare messaggi indiretti al nodo light.

#### 9.4.2 Ricerca di dispositivi

Se entrambi i nodi sono abilitati alla ricerca di nuovi dispositivi (`#define USE_BINDINGS` non commentato), il nodo switch può inviare messaggi al nodo light attraverso il meccanismo della messagistica diretta. Prima di inviare un messaggio diretto, il nodo switch deve individuare l'indirizzo di rete del nodo light. Supponiamo, per semplicità, che il nodo switch conosca l'indirizzo MAC del nodo light. Anche in questo caso, la procedura di ricerca viene gestita attraverso la linea RB5. Premendo il pulsante sul nodo Coordinator viene avviata la procedura di ricerca dispositivi attraverso un messaggio broadcast sulla rete. La

logica a basso livello del nodo RFD vedrà e risponderà a questo messaggio senza nessun supporto da parte del layer applicazioni. Quando il Coordinator riceve la risposta da parte del nodo RFD, è in grado di comunicare con il nodo light inviando messaggi diretti. Se entrambi i nodi sono configurati come switch o light, premendo il pulsante sulla linea RB5, entrambi i nodi prenderanno come indirizzo di rete quello del Coordinator. In questa configurazione ogni nodo è in grado di intercambiare messaggi diretti verso l'altro nodo.

#### 9.4.3 Invio dei messaggi

Una volta creato il binding o scoperto un indirizzo di rete, il nodo switch può inviare messaggi al nodo light per accendere o spegnere un led o una lampada. L'invio del messaggio viene gestito dalla linea RB4 del PICDEM Z. Se il messaggio è diretto, l'applicazione del comando verrà notificata quando la stack riceve una risposta MAC per la trasmissione. Se il messaggio è indiretto, l'applicazione del comando verrà notificata quando la stack memorizza il messaggio per l'ultima trasmissione. Quando l'applicazione viene notificata e il nodo light riceve il comando, viene invertito il livello logico della linea RA1 dove risulta essere collegato il carico (un led nel caso del PICDEM Z).

#### 9.5 Utilizzo della stack Microchip

La progettazione di un sistema basato sul protocollo ZigBee si basa sui seguenti passi:

- ottenere un indirizzo OUI

- determinare la copertura radio in base al data rate e alla geografia dell'applicazione richiesta
- selezionare il microcontrollore adatto all'applicazione
- sviluppare l'applicazione ZigBee usando come supporto la stack messa a disposizione dalla Microchip
- ottenere tutte le certificazioni per gli apparati RF

Lo sviluppo dell'applicazione, invece, si basa sui seguenti passi:

1. determinare il profilo del sistema
2. determinare la struttura degli endpoint che ogni dispositivo andrà ad utilizzare
3. creare una nuova directory di progetto nella quale andare a creare tutti i file sorgenti dell'applicazione
4. utilizzare il tool ZENA per generare i file di configurazione in base al tipo di dispositivo, alla configurazione e alla struttura degli endpoint
5. sviluppare l'applicazione partendo dai modelli messi a disposizione dalla Microchip
6. includere il codice e le modifiche nel file modello

#### 9.6 Interfacciamento con la stack

Per poter accedere alle funzioni del protocollo ZigBee, il sorgente dell'applicazione deve contenere il seguente header:  
`#include zAPL.h`

Il Coordinator del protocollo necessita inoltre di variabili di supporto per tenere traccia delle primitive che vengono eseguite dalla stack:

```
ZIGBEE_PRIMITIVE currentPrimitive;
```

Un router del protocollo ZigBee o un dispositivo End necessitano anche loro di tenere traccia delle primitive in esecuzione. In aggiunta, questi necessitano di altre variabili di supporto per implementare le procedure di ricerca e associazione ad una rete:

```
NETWORK_DESCRIPTOR * currentNetworkDescriptor;
ZIGBEE_PRIMITIVE currentPrimitive;
NETWORK_DESCRIPTOR * NetworkDescriptor;
```

Infine è necessario configurare tutti i pin per l'interfacciamento con il modulo ricetrasmittitore. Il processo di configurazione viene automatizzato attraverso il tool ZENA.

Prima di utilizzare la stack, è necessario inicializzarla. Devono essere abilitate le interrupt:

```
ZigBeeInit();
RCONbits.IPEN = 1;
INTCONbits.GIEH = 1;
```

A questo punto l'applicazione è in grado di interfacciarsi con la stack attraverso le primitive definite nel protocollo ZigBee e nelle specifiche IEEE 802.15.4. Le operazioni della stack vengono eseguite chiamando la funzione *ZigBeeTasks()*. L'operazione verrà eseguita fino a quando il percorso di esecuzione della primitiva richiesta non sarà completo. Poiché è possibile eseguire una sola primitiva alla volta, viene utilizzata una singola struttura dati per memorizzare tutti i parametri della primitiva in esecuzione. La struttura è visibile nel fi-

le di header *ZigBeeTasks.h*. Processata una primitiva, è importante fare attenzione a non mandare l'esecuzione in loop, indicando la prossima primitiva da eseguire o mandando l'esecuzione in stallo (*NO\_PRIMITIVE*). I file di modello messi a disposizione dalla Microchip includono le impostazioni di processamento predefiniti per molte primitive. L'utilizzo di due primitive richiedono l'utilizzo di codice aggiuntivo: *APSDE\_DATA\_indication* e *NO\_PRIMITIVE*. La figura 7 mostra la struttura base dell'applicazione.

```
while (1)
{
  CLRWD();
  ZigBeeTasks( &currentPrimitive );

  switch (currentPrimitive)
  {
    // Include cases for each required primitive.
    // Be sure to update currentPrimitive!

    default:
      currentPrimitive = NO_PRIMITIVE;
      break;
  }
}
```

Figura 7: Struttura base dell'applicazione

## 9.7 Creazione e accesso ad una rete

Anche la creazione della rete viene riportata nel file di modello. Il processo viene inicializzato nella primitiva *NO\_PRIMITIVE*. Se il dispositivo è un Coordinator e, se non è ancora stata formata nessuna rete, questo avvierà il processo relativo alla formazione di una rete attraverso la primitiva *NLME\_NETWORK\_FORMATION\_request*. Se il dispositivo non è un Coordinator, tenterà di accedere alla rete. Se un dispositivo era già presente nella rete, tenterà di accedere nuovamente nella rete come orfano attraverso la primitiva *NLME\_JOIN\_request*

con il parametro *RejoinNetwork* settato al valore *TRUE*. Se l'accesso fallisce o il dispositivo non era in una rete, tenterà di entrare come nuovo dispositivo con la primitiva *NLME\_NETWORK\_DISCOVERY\_request*. La primitiva permette di rilevare le reti disponibili nel campo di copertura radio del dispositivo. Il codice esecutivo selezionerà una delle reti disponibili e cercherà di accedervi attraverso la primitiva *NLME\_JOIN\_request* con il parametro *RejoinNetwork* settato al valore booleano *FALSE*.

### 9.8 Ricezione dei messaggi

La notifica della ricezione di un messaggio viene fornita attraverso la primitiva *APSDE\_DATA\_indication*. Quando la primitiva viene richiamata, i parametri utilizzati dalla stessa primitiva contengono informazioni sul messaggio, mentre il messaggio ricevuto viene memorizzato in un buffer. Il messaggio viene successivamente estratto dal buffer attraverso la funzione *APLGet()*. Il parametro *DstEndpoint* indica l'endpoint di destinazione del messaggio. Se l'endpoint è valido, il messaggio può essere processato. La figura 8 mostra il modello di codice utilizzato per la ricezione dei messaggi.

### 9.9 Invio dei messaggi

La stack Microchip per il protocollo ZigBee prevede l'invio di un solo messaggio alla volta. I passi da seguire sono i seguenti:

1. verificare che il layer applicazione sia pronto per l'invio di un nuovo messaggio verificando che il valore ritornato dalla primitiva *ZigBeeReady()* sia al valore booleano *TRUE*.
2. bloccare l'accesso alla primitiva *ZigBeeReady()* attraverso la primitiva *ZigBeeBlockTx()*. Un eventuale accesso alla primitiva *ZigBeeReady()* ritornerà il valore *FALSE*
3. trasferire il contenuto del messaggio nell'array *TxBuffer*, usando la variabile *TxData* per indirizzare le celle dell'array. Completato il trasferimento, la variabile *TxData* dovrebbe puntare alla prima cella dell'array contenente il messaggio
4. caricare la primitiva *APSDE\_DATA\_request*
5. settare il parametro *currentPrimitive* a *APSDE\_DATA\_request*, e chiamare la primitiva *ZigBeeTasks()*.

Tipicamente i messaggi vengono inviati dall'applicazione in due posizioni differenti:

- nel processo *APSDE\_DATA\_indication*, in risposta ad un messaggio ricevuto
- nel processo *NO\_PRIMITIVE*, in risposta ad un evento dell'applicazione.

Il processo di invio messaggi è identico per entrambe le locazioni. E' porre attenzione alle seguenti note:

- ogni frame *APS* deve avere un identificatore di transazione unico. Questo viene ottenuto dalla stack mediante la funzione *APLGetTransID()*
- *TxData* deve puntare alla prossima locazione disponibile, così *TxBuffer* viene caricata usando un indirizzamento ad incrementi successivi

```

case APSDE_DATA_indication:
{
    // Declare variables used by this primitive.

    currentPrimitive = NO_PRIMITIVE; // This may change during processing.
    frameHeader = APLGet();

    switch (params.APSDE_DATA_indication.DstEndpoint)
    {
        case EP_ZDO:
            // Handle all ZDO responses to requests we sent.
            break;

            // Include cases for all application endpoints.
    }
    APLDiscard();
}
break;

```

Figura 8: Codice ricezione messaggio

- l'endpoint di destinazione deve essere già stato identificato e memorizzato nella variabile *destinationEndpoint*
- l'indirizzo di rete a 16-bit del dispositivo di destinazione deve essere stato già identificato e memorizzato nella variabile *destinationAddress*
- esiste la possibilità di richiedere che il messaggio venga smistato nella rete, se possibile.

Lo stato di trasmissione del messaggio verrà restituito dalla primitiva *APSDE\_DATA\_confirm*. Se il messaggio non viene trasmesso con successo, la stack tenterà automaticamente l'invio, un certo numero di volte stabilito nella variabile *apsMaxFrameRetries*. La figura 9 mostra relativo alla trasmissione dei messaggi.

### 9.10 Richiesta e ricezione dati da un RFD

Poiché gli RFD normalmente disabilitano l'alimentazione dei propri ricetrasmitti-

tori quando vanno in modalità risparmio energetico, essi devono richiedere i messaggi quando riattivano le relative sezioni a radio frequenza. Questo avviene attraverso la primitiva *NLME\_SYNC\_request*. La figura 10 mostra una sequenza tipica per portare l'RFD in modalità risparmio energetico e una procedura per la riattivazione mediante timer Watchdog o pulsante esterno. E' possibile mandare in modalità risparmio energetico il dispositivo nelle seguenti condizioni operative:

- non c'è nessuna primitiva del protocollo ZigBee pronta ad essere processata
- la stack non stà eseguendo nessun compito in background
- la richiesta di dati precedente è stata portata a termine
- tutti i processi relativi alla specifica applicazione sono stati completati.

```

if (ZigBeeReady())
{
    if (bLightSwitchToggled)
    {
        bLightSwitchToggled = FALSE;
        ZigBeeBlockTx();

        TxBuffer[TxData++] = APL_FRAME_TYPE_KVP | 1;           // KVP, 1 transaction
        TxBuffer[TxData++] = APLGetTransId();
        TxBuffer[TxData++] = APL_FRAME_COMMAND_SET | (APL_FRAME_DATA_TYPE_UINT8<< 4);
        TxBuffer[TxData++] = OnOffSRC_OnOff & 0xFF;           // Attribute ID LSE
        TxBuffer[TxData++] = (OnOffSRC_OnOff >> 8) & 0xFF;    // Attribute ID MSE
        TxBuffer[TxData++] = LIGHT_TOGGLE;

        params.APSDE_DATA_request.DstAddrMode = APS_ADDRESS_16_BIT;
        params.APSDE_DATA_request.DstEndpoint = destinationEndpoint;
        params.APSDE_DATA_request.DstAddress.ShortAddr = destinationAddress;

        params.APSDE_DATA_request.ProfileId.Val = MY_PROFILE_ID;
        params.APSDE_DATA_request.RadiusCounter = DEFAULT_RADIUS;
        params.APSDE_DATA_request.DiscoverRoute = ROUTE_DISCOVERY_ENABLE;
        params.APSDE_DATA_request.TxOptions.Val = 0;
        params.APSDE_DATA_request.SrcEndpoint = EP_SWITCH;
        params.APSDE_DATA_request.ClusterId = OnOffSRC_CLUSTER;

        currentPrimitive = APSDE_DATA_request;
    }
}

```

Figura 9: Codice trasmissione messaggio

Quando il dispositivo RFD richiede i messaggi al nodo padre, riceve le seguenti risposte:

- se il nodo padre ha messaggi memorizzati per il dispositivo RFD, li invierà uno alla volta, e il dispositivo RFD genererà una primitiva *APSDE\_DATA\_indication*
- se il nodo padre non ha nessun messaggio memorizzato, il dispositivo RFD genererà una primitiva *NLME\_SYNC\_confirm* con uno stato *SUCCESS*.

Se l'RFD non riceve nessuna risposta dal nodo padre, genererà una primitiva *NLME\_SYNC\_confirm* con uno stato *NWK\_SYNC\_FAILURE*.

## 9.11 Trasmissioni sicure

La stack Microchip supporta tutte i sette modi di sicurezza dei pacchetti in transito nella rete, definiti nelle specifiche del protocollo ZigBee. Le modalità di protezione possono essere suddivisi in tre gruppi:

- Message Integrity Code (MIC) - assicura l'integrità del pacchetto. Quando viene incluso nel pacchetto di trasmissione, assicura che il pacchetto, completo di header e dati, non ha ricevuto nessuna modifica durante la trasmissione. I dati non sono criptati in questa modalità
- Encryption - consente di criptare i dati trasmessi. Il testo in chiaro contenuto nei dati trasmessi non può essere ottenuto se non si è a cono-

```

// If we don't have to execute a primitive, see if we need to request
// data from our parent, or if we can go to sleep.
if (currentPrimitive == NO_PRIMITIVE)
{
    if (!ZigBeeStatus.flags.bits.bDataRequestComplete)
    {
        // We have not received all data from our parent. If we are not waiting
        // for an answer from a data request, send a data request.
        if (!ZigBeeStatus.flags.bits.bRequestingData)
        {
            if (ZigBeeReady())
            {
                // Our parent still may have data for us.
                params.NLME_SYNC_request.Track = FALSE;
                currentPrimitive = NLME_SYNC_request;
            }
        }
    }
    else
    {
        if (!ZigBeeStatus.flags.bits.bHasBackgroundTasks &&
            myProcessesAreDone())
        {
            // We do not have a primitive to execute, we've extracted all messages
            // that our parent has for us, the stack has no background tasks,
            // and all application-specific processes are complete.
            // Now we can go to sleep. Make sure that the UART is finished,
            // turn off the transceiver, and make sure that we wakeup from key press.
            while (!ConsoleIsPutReady());
            AFLDisable();
            INTOCONbits.RBIE = 1;
            SLEEP();
            NOP();

            // We just woke up from sleep. Turn on the transceiver and
            // request data from our parent.
            AFLEnable();
            params.NLME_SYNC_request.Track = FALSE;
            currentPrimitive = NLME_SYNC_request;
        }
    }
}
}

```

Figura 10: Codice ricezione messaggi RFD

scenza della chiave usata nella cifratura. Questa modalità non verifica l'integrità o il contenuto dell'header del pacchetto, ma include la sorgente del pacchetto originale e il frame counter

- ENC-MIC - è una combinazione dei gruppi descritti nei primi due punti. In questa modalità i dati vengono cifrati e allo stesso tempo viene garantita l'integrità del pacchetto.

Chiaramente è possibile usare trasmissioni non protette attraverso il codice *0x00*. La capacità di ogni modo di sicurezza viene riportato nella figura 11.

Il protocollo ZigBee definisce supporti diversi per la sicurezza di dispositivi commerciale o residenziali, in base all'uso della chiave di cifratura. La differenza

sostanziale tra i due consiste nel fatto che mentre nella modalità residenziale viene utilizzata una chiave unica di rete tra la rete e i pacchetti, nella modalità commerciale viene richiesta la generazione di una chiave di sicurezza individuale tra ogni coppia di nodi in comunicazione. Allo stato attuale, la stack Microchip prevede l'utilizzo della sola modalità residenziale.

La stack supporta reti con o senza chiavi di sicurezza preconfigurate. La sicurezza viene supportata sia sul layer NWK che sul layer APL, in base alle esigenze dell'applicazione. E' possibile abilitarne il supporto anche sul layer MAC.

Il protocollo di sicurezza ZigBee si specifica dunque combinando tre elementi:

- il frame counter
- l'indirizzo esteso della sorgente (nel

Security Mode		Security Service			MIC Length (Bytes)
Identifier	Name	Access Control	Data Encryption	Frame Integrity	
0x01	MIC-32	X		X	4
0x02	MIC-64	X		X	8
0x03	MIC-128	X		X	16
0x04	ENC	X	X		0
0x05	ENC-MIC-32	X	X	X	4
0x06	ENC-MIC-64	X	X	X	8
0x07	ENC-MIC-128	X	X	X	16

Figura 11: Servizi di sicurezza del ZigBee

caso in cui sia attivo il layer MAC);  
 l'indirizzo non esteso della sorgente  
 nel caso in cui siano attivi i layer  
 NWK e APL

- il key sequence number (per il layer MAC) o il security control byte (per i layer NWK a APL).

Materiale, application note e codici sorgenti possono essere scaricati direttamente dal sito della Microchip ([www.microchip.com](http://www.microchip.com)).

## Riferimenti bibliografici

- [1] Microchip, *‘Microchip Stack for the ZigBee™ Protocol’*, Application note AN965.
- [2] Microchip, *‘MRF24J40 Data Sheet’*.
- [3] ZigBee Alliance, *‘Wireless Control That Simply Works’*, <http://www.zigbee.org>