Università di Pisa
Facoltà di Ingegneria

Corso di Laurea Specialistica in
Ingegneria Dell'Automazione

Integrazione di reti eterogenee RFID e di sensori wireless IEEE 802.15.4 per applicazioni di sicurezza in ambienti industriali automatizzati

Candidato:

Cosimo Salvatore

Relatori:

Prof. Giuseppe Lipari

Dr. Paolo Pagano

Dr. Matteo Petracca

Anno accademico 2011/2012

Abstract

The concepts of *Internet Of Things* and *Pervasive Computing* are based on the idea that in the future the information processing will be integrated with the environment and the objects will be able to communicate and share information among them.

The Radio Frequency Identification (RFID) and Wireless Sensor Networks (WSN) technologies are two key components in the *Internet Of Things* and *Pervasive Computing* research fields, since both of them can be used for coupling the physical and th virtual world.

The new opportunities allowed by these technologies enable applications in many areas, more in particular in this thesis work safety issues in industrial intelligent environments have been addressed. The main goal of the work is to provide an advanced system for the management of endangered areas and protective objects by means off a software and hardware integration between RFID and WSNs.

Sommario

I concetti di *Internet Of Things* e *Pervasive Computing* si basano sull'idea che in futuro l'elaborazione delle informazioni sarà integrata con l'ambiente e che gli oggetti saranno in grado di comunicare tra loro e condividere informazioni.

Le tecnologie Radio Frequency Identification (RFID) e Wireless Sensor Network (WSN) sono due componenti fondamentali nel campo delle *Internet Of Things* e del *Pervasive Computing*, dal momento che entrambe possono essere utilizzate per l'associazione tra mondo fisico e mondo virtuale.

Le nuove opportunità permesse da questo tipo di tecnologie trovano applicazioni in molti ambiti, in particolare in questo lavoro di tesi è stato affrontato il tema della sicurezza in ambienti industriali intelligenti per fornire un sistema avanzato di gestione delle aree a rischio e dei dispositivi di protezione individuale basato sull'integrazione software e hardware di RFID e WSN.

Indice

Abstract i

Sommario ii

Indice iii

Elenco delle figure vi

1 Introduzione 1
- 1.1 Valutazione del rischio in ambito lavorativo 2
- 1.2 Tematiche relative alla sicurezza affrontate nel lavoro di tesi . 5
  - 1.2.1 Dispositivi di Protezione Individuale 5
  - 1.2.2 Controllo accessi nelle aree a rischio 6
  - 1.2.3 Ottimizzazione di tecnologie già esistenti 7

2 Wireless Sensor Networks 9
- 2.1 Requisiti tipici 9
- 2.2 Dispositivi utilizzati nelle WSN 10
- 2.3 Lo standard IEEE 802.15.4 11
  - 2.3.1 Tipi di dispositivi e ruoli 12
  - 2.3.2 Tipologia di rete 12
  - 2.3.3 Architettura del protocollo 13
  - 2.3.4 Il livello fisico 13
  - 2.3.5 Il livello MAC 15
  - 2.3.6 Politica di gestione degli indirizzi 15
- 2.4 Applicazioni 16

3 Radio Frequency Identification 18
- 3.1 Componenti del sistema RFID 18
- 3.2 Classificazione per frequenze e applicazioni 21
  - 3.2.1 TAG induttivi LF 120-145kHz 23
  - 3.2.2 TAG indutivi HF 13,56 MHz 23
  - 3.2.3 TAG elettromagnetici UHF Media 860-950 MHz . . . 24
  - 3.2.4 TAG elettromagnetici UHF Alta e SHF 2,4GHz 24

3.3 Funzionamento dei tag passivi e semi-passivi 24
3.3.1 Accoppiamento induttivo 25
3.3.2 Accoppiamento elettromagnetico 25
3.3.3 Funzionamento dei tag attivi 29
3.4 Applicazioni 29
3.4.1 Lo standard Electronic Product Code 32

4 Integrazione tra WSNs e RFID 34
4.1 Integrazione di TAG RFID con sensori 35
4.2 Integrazione tra tag RFID e nodo WSN 36
4.3 Integrazione tra reader RFID e nodo WSN 37
4.4 Sistemi ibridi RFID-WSN 39

5 Scenario di riferimento 41
5.1 Requisiti del sistema 41
5.2 Overview del sistema e building blocks 42
5.3 Esempio: cabina di trasformazione 44

6 Componenti del sistema 46
6.1 Descrizione del software 46
6.1.1 ERIKA Enterprise 46
6.1.2 Tools Microchip per il supporto alla programmazione . 48
6.1.3 Framework Qt 49
6.2 Descrizione dell'Hardware 50
6.2.1 Seed-eye board 50
6.2.2 Tag RFID basato su chip IDS sl900a 51
6.2.3 Tag RFID UHF passivi 53
6.2.4 Reader RFID UHF 55
6.2.5 Antenna RFID UHF 56
6.3 Integrazione dell'hardware 57
6.3.1 Nodo WSN integrato con reader RFID UHF 57
6.3.2 Nodo WSN integrato con tag RFID UHF 57
6.3.3 Gestione dell'interrupt mediante segnale a radiofrequenza 59
6.4 Implementazione dei driver per ERIKA Enterprise 62

7 Progettazione del sistema 65
7.1 Data Management 65
7.1.1 Organizzazione della memoria EPC 67
7.1.2 Organizzazione della memoria USER 67
7.2 Meccanismo di autenticazione 69
7.3 Protocollo di comunicazione della rete WSN e localizzazione . 69
7.4 Database e GUI su PC 71

8 Prestazioni del sistema **73**
8.1 Affidabilità e tempi necessari per il processo di autenticazione 73
8.2 Riduzione del consumo energetico dei nodi 74

9 Conclusioni **78**

A Driver per Erika Enterprise **79**
A.1 Driver per il reader RFID 79
A.2 Driver per il tag IDS sl900a 90
A.3 Implementazione della funzione di interrupt in ERIKA 95

B Inizializzazione della memoria **96**

C Meccanismo di autenticazione **98**

D Implementazione del sistema **101**
D.1 Implementazione del codice del nodo coordinator 101
D.2 Implementazione del codice del nodo mobile 110

E Implementazione della GUI **117**

Bibliografia **128**

Ringraziamenti **131**

Elenco delle figure

1.1 Smart factories. 7

2.1 Schema a blocchi di uno smart sensor. 11
2.2 Tipologie a stella e peer to peer (mesh e cluster-tree). 14
2.3 Lo stack IEEE 802.15.4. 14

3.1 Alcuni esempi di tag RFID passivi. 19
3.2 Alcuni esempi di tag RFID semi-passivi. 20
3.3 Alcuni esempi di tag RFID attivi. 21
3.4 Frequenze di riferimento per la tecnologia RFID. 22
3.5 Accoppiamento induttivo (campo vicino). 26
3.6 Accoppiamento elettromagnetico (campo lontano). 26
3.7 Circuito equivalente del tag. 27
3.8 Applicazioni della tecnologia RFID HF. 30
3.9 Applicazioni della tecnologia RFID UHF. 30
3.10 Codice EPC. 32
3.11 Stack EPC global. 33

4.1 Integrazione di TAG RFID con sensori. 36
4.2 Integrazione tra tag RFID e nodo WSN. 37
4.3 Schema di un dispositivo integrato reader RFID e nodo WSN. 38
4.4 Integrazione tra reader RFID e nodi WSN. 39
4.5 Architettura mista RFID-WSN. 40

5.1 Esempio di layout di un impianto industriale. 42
5.2 Schema di funzionamento del sistema. 43
5.3 Esempio cabina di trasformazione. 45

6.1 Architettura di *ERIKA RTOS*. 47
6.2 Architettura del sistema di comunicazione μWireless. 48
6.3 Seed-eye board. 50
6.4 Tag RFID basato su chip IDS sl900a. 52
6.5 Schema del chip IDS sl900a. 53
6.6 Tag passivi. 54

6.7 Reader RFID UHF. 55
6.8 Antenna RFID UHF. 56
6.9 Nodo WSN integrato con reader RFID UHF. 58
6.10 Integrazione del chip sl900a con il micro PIC32. 58
6.11 Livelli di tensione in caso di scrittura sul chip sl900a. 59
6.12 Board di connessione. 61
6.13 Nodo WSN integrato con tag RFID UHF. 62

7.1 Indirizzi della memoria del tag IDS sl900a. 66
7.2 Schema del meccanismo di autenticazione. 70
7.3 Interfaccia grafica su pc. 72

8.1 Scenario di testing. 74
8.2 Numero di tentativi di autenticazione. 75
8.3 Tempi rilevati per l'autenticazione. 75
8.4 Tempi rilevati senza errori di lettura. 76
8.5 Durata delle batterie. 77

Capitolo 1

Introduzione

Ogni anno mediamente il 6% dei lavoratori italiani subisce un incidente sul lavoro [1]. Si tratta di quasi un milione di incidenti di diversa natura e gravità, dei quali oltre 27 mila determinano una invalidità permanente nell'infortunato, e più di 1.300 ne causano la morte. Ciò equivale a dire che ogni giorno tre persone perdono la vita per disgrazie legate alla propria attività lavorativa. Tali numeri rendono la sicurezza sul lavoro un tema molto sentito nella società odierna.

Secondo un'accezione diventata ormai comune per sicurezza nei luoghi di lavoro si intendono tutte quelle precauzioni che devono essere prese per garantire ai lavoratori un luogo di lavoro sicuro e sano, con lo scopo finale di evitare infortuni e/o malattie professionale mentre si lavora.

L'entità del problema e la particolarità della materia hanno reso necessario, nel tempo, una adeguata legislazione. In Italia, la sicurezza sul lavoro è oggi definita da un nuovo decreto, il *D. Lgs. 81/2008*, comunemente conosciuto come Testo Unico Sicurezza Lavoro entrato in vigore, per la quasi totalità degli articoli, il 15 maggio 2008. Esso abroga le precedenti norme in materia *(p.e. la 626/94)* ed introduce importanti innovazioni nel campo della tutela della sicurezza e salute dei lavoratori, focalizzando l'attenzione sulle regole e gli strumenti operativi. Tali regole determinano, tra l'altro, l'obbligo per tutte le aziende, pur con un solo dipendente, ad organizzare e porre in essere procedure altrettanto complesse ed articolate, con obiettive e consistenti difficoltà per gran parte di esse, costringendole ad affrontare, impreparate, nuovi problemi. La complessa e ampia legislazione introdotta dal *D. Lgs. 81/2008* impone precisi e vincolanti obblighi per tutte le aziende, prevedendo anche una capillare sorveglianza sull'osservanza delle norme esercitata da organismi vari (ASL, VVF, Ispettorato del Lavoro, etc.), con una possibile conseguente azione sanzionatoria nei confronti delle aziende inadempienti.

Nel contesto appena descritto va ad inserirsi il presente lavoro di tesi, ponendosi come obiettivo quello di fornire strumenti tecnologici in grado di

aumentare il livello di sicurezza di ambienti industriali soggetti a rischio, andando inoltre ad ottimizzare procedure di sicurezza già esistenti. L'attenzione sarà focalizzata sui *dispositivi di protezione individuale* in dotazione agli operai, sul *controllo accessi delle aree a rischio* e sulla ottimizzazione di tecnologie di ultima generazione installate nei luoghi di lavoro per aumentare la sicurezza.

1.1 Valutazione del rischio in ambito lavorativo

La valutazione del rischio negli ambienti di lavoro è normata dal Testo unico sulla sicurezza sul lavoro quale il *D.Lgs 81/2008* (integrato con il *Decreto Legislativo 106/2009*). Essa consiste nella determinazione dei rischi fisici, chimici, biologici, da movimentazione carichi, da videoterminali, incendio, indoor, outdoor, ecc. a cui i lavoratori possono essere sottoposti L valutazione del rischio è realizzata quantificando la probabilità che si verifichi un effetto dannoso a carico dei lavoratori e valutando la gravità di tale evento denominata magnitudo.

L'obbiettivo principale di una procedura di valutazione del rischio è quello di prevenire due tipologie di eventi rischiosi, classificandoli come:

- *Rischio infortunistico*: rischio di incorrere in un danno che ha cause da ricercarsi all'interno del posto di lavoro e che si manifesta nell'immediato o comunque entro l'orario di lavoro stesso;
- *Rischio igienistico*: rischio di incorrere in un danno dovuto ad una esposizione prolungata e a livelli elevati ad un agente chimico, fisico, biologico che causa danni che si manifestano a distanza di tempo.

Nella valutazione del rischio sono coinvolte varie fasi che richiedono il contributo di discipline differenti. La valutazione inoltre, deve tenere conto del tipo di ambiente di lavoro, dei processi che intervengono al suo interno e della loro complessità. Tuttavia è possibile riassumere il processo di valutazione nei seguenti passaggi:

- *Individuazione delle sorgenti di pericolo*: è finalizzata ad individuare gli elementi in grado di causare un effetto avverso (mediante monitoraggio ambientale e/o biologico e sorveglianza sanitaria), definendone la dose-risposta e valutando la possibilità di esposizione. In pratica si procede alla caratterizzazione del rischio ovvero si considera la dose di esposizione e la sua correlazione con la tipologia, severità e prevalenza dell'effetto avverso nella popolazione in oggetto. Poiché diversi fattori possono influenzare la quantità di contaminante che viene ricevuta, viene prodotta una distribuzione dello spettro dei possibili valori. Particolare cura viene data alla determinazione dell'esposizione dei lavoratori suscettibili.

- *Individuazione dei soggetti esposti*: mira ad identificare i soggetti che potranno essere esposti ad un particolare pericolo, tenendo conto delle differenze di sesso, età, etnia ecc; in questo modo si caratterizza il personale permettendo di individuare coloro che possiedono le caratteristiche fisiche-psicologiche adatte (mediante visita medica).

- *Stabilire la priorità dei rischi*: i risultati delle due fasi vengono poi combinati per produrre una stima del rischio (Risk Assessment). In questo modo si è in grado di stilare una classifica in base a cui si stabilisce l'ordine degli interventi da eseguire.

- *Scelta degli interventi*: in base alle priorità stabilite in precedenza, alle informazioni che si possono acquisite dalle esperienze passate o dalla bibliografia e alle informazioni relative al luogo oggetto della valutazione, si scelgono degli interventi che devono essere efficaci, efficienti e adatti al contesto in cui devono essere applicati, facendo anche una valutazione costo-beneficio. Da notare che si devono preferire interventi alla fonte del pericolo, alle misure collettive, che comunque sono preferibili alle misure individuali.

- *Attuazione le misure di controllo sugli interventi*: una volta messi in pratica gli interventi, deve essere controllata periodicamente la loro effettiva funzionalità, con controlli statistici, ambientali, biologici, ecc. Il controllo è meglio se programmato e descritto minuziosamente cosi da poter verificare successivamente eventuali modifiche.

- *Valutazione dell'efficacia dell'intervento*: una volta acquisiti i dati relativi ai controlli sugli interventi , è necessario discutere della loro efficacia ed efficienza cosi da poter prendere in considerazione eventuali miglioramenti o altri accorgimenti da prendere. Nel caso di cambiamenti interni, è necessario valutare se gli interventi che erano stati attuati inizialmente, siano ancora efficaci con le nuove modifiche.

La valutazione del rischio, e quindi la successiva stesura del Documento di Valutazione del Rischio (DVR), è uno degli obblighi non delegabili del datore di lavoro. Con esso dovranno partecipare anche il Responsabile del Servizio di Prevenzione e Protezione (RSPP), il Medico Competente (nei casi previsti) e il Responsabile dei Lavoratori per la Sicurezza (RLS). A tal fine poi, potranno partecipare anche figure professionali specifiche che potranno collaborare nella definizione dei rischi.

Il Documento di Valutazione del Rischio (DVR) è la relazione stesa a seguito della valutazione del rischio. Questo documento deve riguardare tutti i rischi per la sicurezza e la salute dei lavoratori, compresi quelli riguardanti gruppi di lavoratori esposti a rischi particolari come stress lavoro-correlato, lavoratrici in stato di gravidanza, nonché quelli connessi a differenze di genere, età, etnia e tipologia contrattuale. Il documento può presentarsi in

formato cartaceo o informatico ma deve sempre essere custodito all'interno dell'azienda (in caso di supporto informatico è necessario possedere anche l'applicazione per poter aver accesso al documento); il documento inoltre, per essere valido, deve essere munito di data certa o attestata dalla sottoscrizione, oltre che del datore di lavoro, anche del responsabile del servizio di prevenzione e protezione, del rappresentante dei lavoratori per la sicurezza e/o del rappresentante dei lavoratori per la sicurezza territoriale, e dal medico competente,(ove nominato). I contenuti obbligatori[6] per la stesura di un DVR completo sono: la relazione sulla valutazione del rischio nell'ambiente di lavoro, facendo riferimento ai criteri utilizzati; un elenco delle misure di prevenzione e protezione utilizzati (misure collettive, misure organizzative, dispositivi di protezione individuale, ecc) un programma di azioni di miglioramento sui sistemi di prevenzione e protezione per innalzare via via il livello di sicurezza; la descrizione delle procedure di attuazione dei sistemi di prevenzione e protezione, e l'indicazione dei soggetti coinvolti in tali procedure, con definizione di incarichi specifici; indicazione del Responsabile del Servizio di Prevenzione e Protezione, del Responsabile dei Lavoratori e del Medico Competente; l'indicazione e la descrizione delle mansioni a rischio che richiedono un'idoneità professionale. La valutazione dei rischi, e quindi la stesura del relativo documento, devono essere fatti entro novanta giorni dall'apertura di una nuova attività lavorativa; inoltre, ogni qualvolta intervenga una qualsiasi modifica all'interno del ciclo produttivo e/o nell'organizzazione aziendale, il DVR deve essere aggiornato e adeguato alla nuova realtà lavorativa.

E' importante sottolineare l'importanza delle quattro figure principali che compongono l'azienda e che devono avere un ruolo attivo e consapevole nell'attuazione della valutazione del rischio.

- *Il Datore di Lavoro*: colui che esegue la valutazione del rischio, che sceglie e acquista le misure di prevenzione e protezione e che decide le misure organizzative e procedurali da adottare.

- *Il Dirigente*: colui che attua le direttive del datore di lavoro organizza le mansioni del personale e con esso sceglie e acquista le attrezzature per la prevenzione e protezione.

- *Il Preposto*: colui che è incaricato di sorvegliare i lavoratori affinché questi lavorino al meglio seguendo le norme di sicurezza stabilite.

- *Il Lavoratore*: colui che opera all'interno dell'azienda e che ha il compito di rispettare le norme di sicurezza, utilizzare e/o indossare gli appositi dispositivi di protezione e operare nel modo corretto.

1.2 Tematiche relative alla sicurezza affrontate nel lavoro di tesi

Da una analisi efficace della valutazione del rischio tutti gli strumenti necessari a garantire il livello di sicurezza richiesto. Vengono individuati le procedure da attuare, le restrizioni, ma anche gli equipments da fornire ai lavoratori e gli strumenti tecnologici avanzati da utilizzare per scongiurare la possibilità di infortuni. In particolare in questo lavoro di tesi ci si è occupati di:

- controllo dispositivi individuali di protezione (DPI);
- controllo accessi alle aree a rischio;
- ottimizzazione tecnologie avanzate già esistenti.

1.2.1 Dispositivi di Protezione Individuale

Si intende per Dispositivi di Protezione Individuale, definizione spesso surrogata dall'acronimo D.P.I., qualsiasi attrezzatura destinata ad essere indossata e tenuta dal lavoratore allo scopo di proteggerlo contro uno o più rischi suscettibili di minacciarne la sicurezza o la salute durante il lavoro, nonché ogni complemento o accessorio destinato a tale scopo **(art. 74, comma 1 del D.Lgs. 9 aprile 2008, n.81)**

Per quanto riguarda i dispositivi individuali di protezione il *D.Lgs. 81/2008* stabilisce gli obblighi del datore di lavoro, dei dirigenti e dei preposti e dell'operaio e le caratteristiche dei DPI. Rispetto alle norme precedentemente in vigore vi le responsabilità sono state spostate maggiormente sul datore di lavoro il quale ha l'obbligo di:

- fornire ai lavoratori D.P.I. idonei rispetto ai rischi specifici cui sono esposti;
- informare i lavoratori sui rischi dai quali i D.P.I. lo proteggono e attivare adeguate iniziative di formazione sulle caratteristiche dei DPI prescelti;
- sottoporre i lavoratori che devono utilizzare i D.P.I. a specifico addestramento.

Dal canto loro i lavoratori hanno l'obbligo per i lavoratori di indossare e conservare correttamente i D.P.I. loro forniti, non apporre modifiche di propria iniziativa e di segnalare al datore di lavoro la loro eventuale inadeguatezza o mancanza. Ai dirigenti o ai loro preposti spetta invece il compito di vigilare affinche i D.P.I. vengano utilizzati correttamente.

Proprio il mancato utilizzo dei dPI in condizione nelle quali ve ne sia bisogno è spesso causa di infortuni nei luoghi di lavoro, ciò è causato spesso

sia dalla negligenza degli operai che dalla mancanza di controlli. A questo scopo l'utilizzo di strumenti tecnologici avanzati può rappresentare un valido strumento per garantire l'osservanza delle norme.

I DPI devono:

- essere adeguati alle condizioni presenti sul luogo di lavoro.
- essere adeguati ai rischi da prevenire, senza comportare un rischio maggiore per il lavoratore
- tener conto delle esigenze ergonomiche e della salute del lavoratore

Per quanto riguarda le caratteristiche dei D.P.I. secondo il *D.Lgs. 81/2008* essi sono divisi in tre categorie, in funzione del tipo di rischio:

- I categoria - rischio lieve - autocertificato dal produttore
- II categoria - rischio significativo come ad esempio occhi, mani, braccia, viso - prototipo certificato da un organismo di controllo autorizzato e notificato
- III categoria - comprende tutti i D.P.I. per le vie respiratorie e protezione dagli agenti chimici aggressivi - prototipo certificato da un organismo di controllo autorizzato e notificato, e controllo della produzione

I DPI devono, per legge, riportare il marchio CE il quale indica la conformità ai requisiti essenziali di salute e sicurezza. Inoltre il dispositivo di sicurezza deve contenere un manuale di istruzioni per l'uso, conservazione, pulizia, manutenzione, data di scadenza, categoria e limiti d'uso possibilmente scritto nelle lingue ufficiali. Uno dei problemi maggiori è stabilire quando un dispositivo di protezione individuale è da sostituire. Alcuni dispositivi riportano una data di scadenza, altri richiedono da parte del lavoratore un controllo dello stato di usura al fine di sostituirlo nel caso non sia più idoneo. Ad esempio: un dispositivo delle vie respiratorie dovrà essere sostituito quando l'operatore nota una particolare difficoltà nella respirazione; un occhiale invece deve essere sostituito quando l'operatore rileva una non più perfetta nitidezza delle immagini. Al fine di evitare l'insorgere di problemi per il lavoratore, il datore di lavoro dovrà provvedere a sostituire con una certa frequenza i DPI.

I DPI sono classificati in base alle parti del corpo che devono proteggere **(allegato VIII del D.Lgs. n. 81/08)**, ad esempio dispositivi di protezione della testa, dell'udito, degli occhi e del viso ecc.

1.2.2 Controllo accessi nelle aree a rischio

Con il dlgs 81/2008 il datore di lavoro è responsabile dell'eventuale infortunio che sia conseguenza dell'attività lavorativa, anche se il lavoratore non

aveva attribuite quelle determinate mansioni; il datore di lavoro è responsabile per il semplice fatto di non aver preso le misure appropriate per impedire al lavoratore lo svolgimento di quella determinata attività L'art. 18 tuttavia offre un'opportunità al Datore di lavoro stabilendo che la responsabilità sussiste qualora non abbia adottato misure appropriate affinché solo i lavoratori formati e informati svolgano l'attività a rischio . Prendere le misure appropriate per limitare l'accesso ad aree a rischio vuol dire quindi predisporre tutte le procedure che impediscono l'accesso alle aree a rischio ai non autorizzati e quindi rendere evidenti gli accessi effettuati in modo da poter assumere misure di sicurezza correttive in caso di violazione. Il Datore di lavoro ha quindi sempre più bisogno della tecnologia sia per limitare determinate attività che per monitorare la quotidianità del lavoro e delle operazioni eseguite dai lavoratori. Ad esempio in molti luoghi di lavoro vi è la necessità di diversificare le aree di lavoro in funzione del coefficiente di rischio e di discriminare gli accessi a tali aree; nel presente lavoro di tesi verrà fornito un strumeto tecnologico volto ad automatizzare questo processo.

1.2.3 Ottimizzazione di tecnologie già esistenti

I recenti progressi tecnologici nei sistemi micro-elettro-meccanici, nelle comunicazioni wireless e nell'elettronica digitale hanno permesso lo sviluppo di apparecchi di dimensioni ridotte, a bassa potenza, dai costi contenuti, multifunzionali e capaci di comunicare tra loro tramite tecnologia wireless. Tali

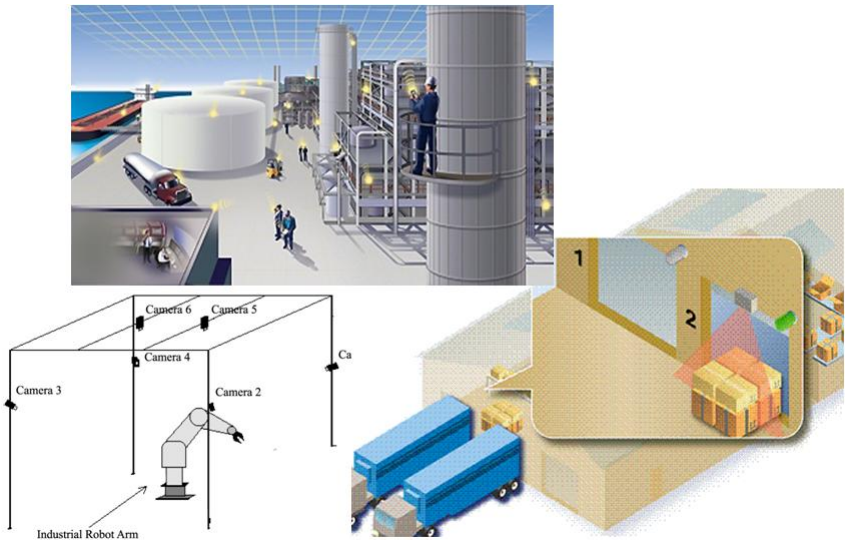

Figura 1.1: Smart factories.

funzionalità rendono sempre più vantaggiosa l'implementazione all'interno di ambienti industriali di sistemi tecnologici di ultima generazione il cui scopo principale non è solo quello di ottimizzare il processo produttivo ma anche

quello di garantire una gestione intelligente dell'ambiente industriale. Nascono in questo modo le cosiddette "Smart factories" (Figura 2.1) all'interno delle quali vengono applicati i principi delle "internet of things" per una gestione sempre più intelligente e automatizzata degli ambienti produttivi.

In quest'ottica tecnologie come le wireless sensor networks e gli RFID sono sempre più diffuse in applicazioni quali la localizzazione di cose e persone, il tracciamento delle merci e l'automazione delle linee produttive. L'applicazione oggetto di questo lavoro di tesi va a configurarsi in tale scenario fornendo strumenti avanzati di gestione del rischio e integrandosi per sua stessa natura con sitemi intelligenti già implementati o da sviluppare.

Capitolo 2

Wireless Sensor Networks

Una Wireless Sensor Network (WSN) è una rete wireless costituita da un insieme di dispositivi autonomi dotati di sensori distribuiti nell'ambiente, che permettono di rilevare fenomeni fisici o monitorare condizioni ambientali. I nodi di una WSN sono sistemi embedded dotati di radio transceiver, o altro tipo di apparato wireless, un piccolo microcontrollore e di strumenti di misurazione. Le reti di questi dispositivi costituiscono l'infrastruttura per la raccolta, fusione e aggregazione dei dati immagazzinati nei nodi sensori.

Un aspetto molto interessante di questa tecnologia è la possibilità di implementare un meccanismo multi-hop per il routing su reti di vasta area. Uno dei principali vantaggi delle WSN è la versatilità, ovvero sono in grado di supportare applicazioni molto differenti tra loro. Per questo sono particolarmente adatte a diversi campi applicativi, fra cui il controllo della salute del cittadino, la sicurezza, il controllo ambientale e del traffico, oltre a quelle in ambito militare, le quali hanno un elevato contenuto di innovazione. Altri vantaggi delle WSN rispetto ai tradizionali sistemi di misurazione sono la pervasività del monitoraggio e un'intrinseca tolleranza ai guasti.

2.1 Requisiti tipici

Dato l'ampio numero di possibili applicazioni delle WSN, non è possibile definire dei requisiti globalmente validi, per cui si è soliti riferirsi ad alcuni requisiti tipici:

1. *Scalabilità.* Il numero dei nodi di una WSN può aumentare nel tempo e diventare molto elevato, pertanto l'architettura ed i protocolli utilizzati devono supportare tale crescita senza un eccessivo degrado delle prestazioni della rete, sin dalle prime fasi di progetto.

2. *Tempo di vita.* In molte applicazioni i nodi saranno alimentati a batteria, quindi potranno disporre di una quantità limitata di energia.

Poiché la sostituzione di tali batterie è spesso impraticabile, l'obiettivo dei ricercatori è divenuto lo studio di algoritmi e protocolli, detti *power-aware protocols*, che consentano di aumentare il tempo di vita della rete. L'attività di comunicazione è la più onerosa in termini di consumi: ogni qualvolta un nodo deve trasmettere l'informazione, dovrà mantenere attivo l'ascolto sul canale radio per verificare la corretta ricezione del destinatario. Per stimare il dispendio energetico dovuto ai moduli di comunicazione, si dovrà valutare i consumi e le tempistiche relative non soltanto alla trasmissione ed alla ricezione, ma anche alle fasi di accensione e spegnimento del transceiver.

3. *Densità.* In una WSN il numero di nodi per unità di area può variare considerabilmente. Diverse applicazioni avranno differenti necessità di densità, inoltre i nodi possono guastarsi, spostarsi oppure disporsi con densità non omogenea. A questo si aggiunge il fatto che il mezzo trasmissivo è condiviso tra i vari nodi, pertanto possono presentarsi problemi di interferenze dovute al riuso di frequenza e per conflitti dovuti a trasmissioni concorrenti da parte dei nodi. La rete dovrà adattarsi alle possibili variazioni ed essere in grado di gestire protocolli di accesso al mezzo che permettano la trasmissione di dati senza il verificarsi di collisioni.

4. *Tolleranza ai guasti.* I nodi possono terminare la propria fonte di alimentazione, essere danneggiati, oppure la comunicazione tra due nodi può essere interrotta. Di conseguenza con *fault tolerance* viene indicata la capacità di una rete di sensori di mantenere le proprie funzionalità in caso di malfunzionamento dei suoi nodi, senza pregiudicarne il corretto funzionamento.

5. *Programmabilità.* I nodi dovrebbero essere programmabili per poter cambiare modalità di funzionamento nel caso sia necessario svolgere nuovi compiti all'interno della rete. Un processamento statico dell'informazione infatti non è spesso sufficiente a gestire i dati in modo esauriente ed esaustivo.

2.2 Dispositivi utilizzati nelle WSN

I dispositivi normalmente utilizzati all'interno delle Wireless Sensor Networks sono normalmente dei *sistemi embedded* ottimizzati per eseguire le operazioni richieste dalla particolare applicazione. Con il termine *embedded* si identificano tutti quei sistemi di elaborazione che non rientrano nella categoria *general-purpose*. In sostanza, data la vastità delle applicazioni in cui sono coinvolti, si tratta di sistemi progettati per svolgere una particolare attività e vengono quindi definiti, in contrapposizione ai classici calcolatori, dispositivi *special-purpose*.

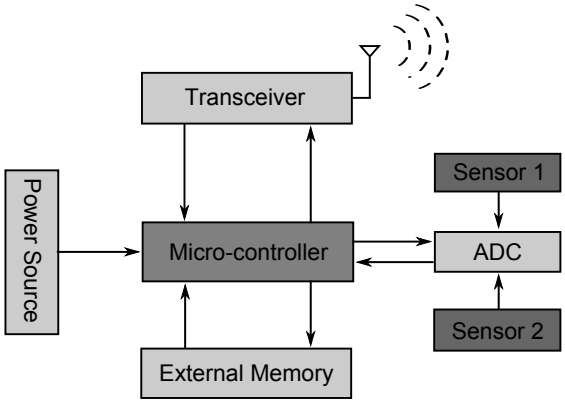

Figura 2.1: Schema a blocchi di uno smart sensor.

I sistemi embedded sono dei circuiti integrati di elementi elettronici ed unità di calcolo; possono essere realizzati con pure tecniche elettroniche *(Application-Specific Integrated Circuit)* oppure mappando via software alcune particolari funzionalità richieste. L'hardware può essere ridotto ai minimi termini per ridurre lo spazio occupato, i consumi ed i costi di fabbricazione. L'incremento delle prestazioni dei dispositivi, nonché la loro miniaturizzazione, ha permesso ai sistemi basati su microcontrollori di essere una delle principali soluzioni in molti settori nell'industria (automotive, controllo industriale, elettromedicali) e del mercato consumer (telefonia mobile, lettori portatili, elettrodomestici, smartcard).

La piattaforma sulla quale può venire sviluppato un sistema embedded varia drasticamente a seconda della sua complessità, dei consumi, del costo e del campo di utilizzo. Si passa dai PLC e microcontrollori più semplici ad architetture più complesse basate su circuiti integrati sofisticati (*System-on-a-chip*). Alcune piattaforme di sviluppo molto usate sono basate su architetture ARM, MIPS, Coldfire/68k, V850, IBM compatibili, PICmicro, Intel 8051, Atmel AVR. I SoC racchiudono, in un singolo circuito integrato di tipo ASIC, microcontroller/CPU e/o DSP, memoria, oscillatori e clock, regolatore di tensione, eventuali interfacce ADC/DAC, USB, ethernet. Un altro comune metodo di progetto prevede l'utilizzo di FPGA (*Field-Programmable Gate Array*), con la programmazione di tutta la logica interna, inclusa la CPU. Tipicamente si fa uso di FPGA affiancandoli ad altri circuiti integrati per l'interfacciamento.

2.3 Lo standard IEEE 802.15.4

Lo standard IEEE 802.15.4 viene approvato nel 2003 (ed aggiornato nel 2006) e definisce il protocollo di trasmissione a basso livello tramite comunicazione radio tra diversi dispositivi di una WPAN (Wireless Personal Area

Network) a basso rate trasmissivo (LR-Low Rate). Lo standard definisce in particolare le specifiche del livello fisico ed del livello MAC (Medium Access Control) con lo scopo di garantire una modalità di connessione wireless a basso data-rate tra i vari dispositivi che necessitano di un basso consumo di potenza così da garantire una lunga durata delle batterie a bordo e che operano in uno spazio operativo dell'ordine di qualche decina di metri. Una Lr-Low Rate è una rete di comunicazione semplice e a basso costo, selettivamente orientata verso applicazioni a basso consumo ed a throughput non elevato, principalmente caratterizzata da:

- Data-rate di 250 kbit/s, 100 kbit/s, 40 kbit/s e 20 kbit/s;
- Tipologia di rete a stella o "peer-to-peer";
- Indirizzo a 64 bit o a 16 bit;
- Allocazione opzionale di guaranteed time slots (GTSs);
- Accesso al canale in modalità CSMA-CA;
- Completa definizione del protocollo per il trasferimento dei dati;
- Basso consumo di energia;
- 16 canali nella banda attorno a 2.4GHz, 10 canali nella banda attorno a 915MHz, un canale ad 868MHz.

2.3.1 Tipi di dispositivi e ruoli

Lo standard 802.15.4 definisce due ruoli per i nodi di una WPAN: device e coordinator o PAN coordinator. I primi sono semplici nodi sensori con lo scopo di rilevare le misurazioni nell'ambiente in cui è collocato e di inviare il risultato di tali misurazioni ai secondi. I coordinator o PAN coordinator sono responsabili della gestione della rete/sottorete WPAN, e sono più complessi rispetto ai device. Nello standard vengono definiti due tipi di nodi: Reduced Function Device (RFD) e Full Function Device (FFD). Gli RFD che possono operare nella rete solo come Device, dispongono solo di un sottoinsieme delle funzionalità MAC e quindi possono comunicare solo con altri RFD. Gli FFD possono operare nella rete come PAN Coordinator, come semplice Coordinator oppure come Device e dispongono di tutte le primitive MAC e quindi possono comunicare sia con altri FFD sia con dispositivi RFD.

2.3.2 Tipologia di rete

Una WPAN è costituita da un minimo di due dispositivi. In ciascuna rete uno solo di essi può configurarsi con PAN Coordinator ed ha il compito di iniziare, gestire e terminare la comunicazione tra i diversi nodi. Una

LR-WPAN può configurarsi in due topologie: topologia a stella e topologia peer-to-peer. Nella topologia a stella ciascun dispositivo può comunicare solo con il coordinatore, il quale controlla e gestice la rete. Un device ha associato una applicazione ed è o il punto iniziale o il punto terminale di una comunicazione. Il coordinator è tipicamente collegato ad alimentazione fissa a differenza degli altri dispositivi che sono dotati di batteria. Una topologia peer-to-peer differisce dalla precedente in quanto ciascun dispositivo può comunicare direttamente con un altro interno alla rete, a patto che questo rientri nella sua area operativa di copertura, senza ricorrere alla mediazione del coordinatore. Questa topologia si presta bene alla formazione di reti di comunicazione decisamente complesse che coinvolgono potenzialmente un elevato numero di dispositivi. Esempi di applicazioni sono il controllo ed il monitoraggio industriale ed ambientale, applicazioni domotiche. Un esempio di rete complessa fondata su una topologia peer-to-peer è il cluster-tree. Ogni dispositivo della rete può funzionare da coordinatore, fornendo servizi ad altri dispositivi coordinatori, anche al di fuori della propria sfera di influenza. Però, soltanto uno di questi coordinatori locali può configurarsi come coordinatore PAN, il quale è tipicamente dotato di maggiori risorse anche a livello computazionale. Una rete peer-to-peer può permettere la comunicazione multi-hop tra qualunque device della rete. Queste funzioni, tuttavia, non sono parte dello standard, ma devono essere fornite dai livelli superiori dello stack. In Figura 2.2 sono illustrate le varie tipologie di rete.

2.3.3 Architettura del protocollo

I livelli definiti dallo standard 802.15.4 sono basati sulla pila ISO/OSI e sono solamente i primi due: il livello fisico (PHY) ed il livello datalink (MAC). L'interazione con i livelli superiori è prevista tramite un sub-layer LLC (Logical Link Control) ed un SSCS (Service Specific Convergence Sublayer). Il primo definisce i tipi di servizio ottenibili per un trasferimento dati mentre il secondo esegue le azioni necessarie per richiedere la trasmissione dati in base al servizio richiesto. In Figura 2.3 è illustrata L'Architettura del protocollo.

2.3.4 Il livello fisico

Il livello fisico fornisce due differenti servizi: quello dati e quello gestione. Le caratteristiche principali implementate a questo livello sono l'attivazione e la disattivazione del trasmettitore radio, il rilevamento dell'energia (ED), l'indicazione della qualità del collegamento (LQI), la selezione del canale, la stima della disponibilità del canale (CCA) e la trasmissione e ricezione di pacchetti sul mezzo fisico (canale radio). I bitrate offerti dal PHY sono mostrati in tabella 2.1.

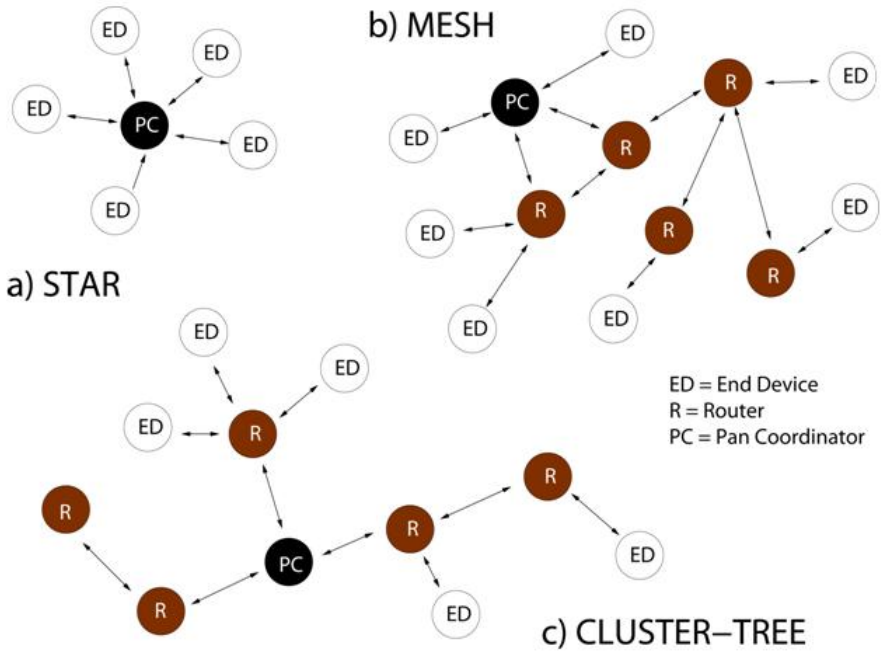

Figura 2.2: Tipologie a stella e peer to peer (mesh e cluster-tree).

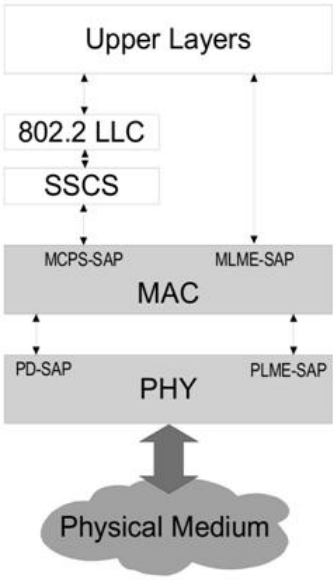

Figura 2.3: Lo stack IEEE 802.15.4.

Data rate	Symbol rate	Banda	Canali	Modulazione
20 Kbps	20 Ksyms	868 - 868.6 Mhz	1	BPSK
40 Kbps	40 Ksyms	905 - 928 Mhz	10	BPSK
250 Kbps	62.5 Ksyms	2.4 - 2.283 Ghz ISM	16	O-QPSK

Tabella 2.1: PHY Bitrates

2.3.5 Il livello MAC

I principali compiti del livello MAC sono il coordinamento dell'accesso al mezzo da parte del tranciver, creazione ed instradamento dei pacchetti, generazione e riconoscimento dell'indirizzo, verifica del numero di sequenza dei pacchetti. Le principali funzioni del livello MAC sono implementate in software a differenza di quanto avviene per il livello fisico e scritte generalmente in linguaggio C. Il livello MAC definisce 4 possibili tipi di frame:

Frame di dati costituito al massimo da 128 bytes è utilizzato per tutti i trasferimenti di dati;

Frame ACK fornisce la conferma che il pacchetto inviato è stato ricevuto correttamente;

Frame di comando MAC fornisce un meccanismo per il controllo e la configurazione da remoto dei nodi client;

Frame di beacon utilizzato dal coordinator per definire la struttura del superframe (se utilizzato) e per permettere ai device di localizzare la rete.

2.3.6 Politica di gestione degli indirizzi

L'IEEE 802.15.4 definisce due tipologie di indirizzi: indirizzi estesi IEEE EUI-64 bit, univoci globalmente, o indirizzi short a 16 bit, univoci soltanto all'interno della PAN. Entrambi i tipi sono supportati dal 6LoWPAN. Per utilizzare indirizzi short a livello di rete, dato che essi sono disponibili soltanto dopo l'assegnamento da parte del coordinatore della PAN e la loro validità è limitata unicamente alla validità dell'associazione, il 6LoWPAN impone che l'utilizzo venga effettuato sotto precise condizioni ed entro certi limiti. In 6LoWPAN si assume che una PAN sia mappata all'interno di uno specifico link Ipv6: questo comporta l'utilizzo di un prefisso comune per i dispositivi di rete. Questa assunzione ha importanti conseguenze nella gestione degli indirizzi per tre aspetti:

- gestione della comunicazione multicast;
- autoconfigurazione della rete;

- creazione e manipolazione indirizzi link local.

2.4 Applicazioni

Il settore del sensor networking è in continua evoluzione, spinto soprattutto dalle tematiche del *pervasive computing*, all'origine di moltissimi spunti di ricerca nei campi più disparati. Le reti di sensori possono essere implementate utilizzando una vasta tipologia di sensori come sensori sismici, magnetici, termici, infrarossi, acustici, radar, di temperatura, di movimento, di pressione e altri ancora. Ciascun nodo sensore può inoltre essere utilizzato in diverse modalità, dall'interrogazione periodica per ottenere un flusso costante di informazione, alla verifica del raggiungimento di una particolare condizione operativa. Gli ambiti di interesse del sensor networking possono essere essenzialmente riuniti in sei macro-aree principali:

- *Domotica.* Disciplina che si occupa dell'integrazione dei dispositivi di controllo e trasduzione, e dei rispettivi sistemi di interconnessione che si trovano nelle abitazioni. Ogni apparato elettronico all'interno della casa come televisori, forni a microonde, frigoriferi, impianto di illuminazione, impianto di riscaldamento e climatizzazione, sistema di sicurezza e altro ancora, è in grado di interagire con gli altri e verso il mondo esterno attraverso altre infrastrutture di rete (solitamente comunicazioni via satellite o internet). Anche in questo caso, costi e dimensioni sono fattori chiave, ma diventano fondamentali i concetti di *autoalimentazione* dei dispositivi e la quasi totale mancanza di manutenzione. Come in precedenza il transfer rate richiesto non è particolarmente elevato, ma il numero di nodi interconnessi può essere ragguardevole anche se generalmente fisso, trattandosi tipicamente di reti statiche.

- *Monitoraggio ambientale.* Esempi di applicazioni in questo ambito possono essere i sistemi di prevenzione degli incendi, le statistiche relative alla fauna protetta, l'agricoltura di precisione, il controllo stradale, la ricerca meteorologica e geofisica ed il controllo dell'inquinamento. In questo caso si ha a che fare con distanze più elevate, anche chilometri, e con reti dalla natura fortemente dinamica e mobile. L'architettura di rete deve garantire un'elevata flessibilità *(ad-hoc network)*, poiché il numero e la dislocazione dei nodi può cambiare in relazione alle effettive necessità di monitoraggio o a particolari situazioni di emergenza. L'alimentazione può essere integrata con sistemi di autogenerazione, come ad esempio le celle fotovoltaiche.

- *Automazione industriale.* L'utilizzo delle WSN nel settore industriale si inserisce nella continua ricerca della diminuzione dei costi per implementare sistemi di controllo per i processi produttivi. E' possibile di-

stinguere tra le problematiche proprie del controllo di processo e quelle proprie della cosiddetta *factory automation*. Le esigenze del primo sono molto simili a quelle del mondo consumer: tempi di aggiornamento e transfer rate relativamente bassi, rete statica con numero di nodi non particolarmente elevato, ma con meno attenzione a costi, dimensioni e alimentazione. Viceversa, per il secondo i requisiti temporali sono molto stringenti: in questo caso la maggior parte delle soluzioni wireless sono da intendersi *ridondanti*, di supporto a tutte quelle situazioni in cui la connessione via cavo è di difficile attuazione. I throughput richiesti divengono elevati e si traducono in consumi elevati, che obbligano al cablaggio dell'alimentazione, rendendo tali soluzioni appetibili solo in casi particolari, come gli organi in movimento.

- *Settore sanitario.* Alcuni esempi in questo campo possono essere la trasmissione dei parametri fisiologici dei pazienti all'interno degli ospedali, le attività diagnostiche, la somministrazione di medicinali, il *personal healthcare*. Il monitoraggio dei parametri fisiologici dei pazienti quali temperatura, pressione sanguigna, pulsazioni cardiache in modo non invasivo consentono l'intervento tempestivo del personale medico in caso di bisogno. Altra applicazione interessante è il controllo remoto dello stato delle persone anziane per prevenire situazioni di pericolo quali per esempio una caduta o un malore. In questo campo l'affidabilità della connessione è di vitale importanza, così come la compattezza delle dimensioni; distanze e performance vanno valutate in funzione dell'applicazione in esame.

- *Applicazioni militari.* Come per il precedente campo, si tratta di un insieme di applicazioni caratterizzate da bassi volumi produttivi, quasi a livello di prototipazione. Le reti di sensori wireless possono diventare parte integrante delle più comuni attività militari come il comando, il controllo dei campi di battaglia, la rilevazione degli spostamenti delle truppe nemiche, la sorveglianza, le operazioni di localizzazione dei bersagli, ma anche il riconoscimento di agenti chimico-fisici nel caso di uso di armi non convenzionali. L'impiego di sensori in grandi quantità anche in ambienti inospitali fa sì che, in caso di distruzione di uno o più nodi, non venga necessariamente influenzata l'efficienza della rete, cosa che invece potrebbe accadere utilizzando reti cablate.

Capitolo 3

Radio Frequency Identification

Il termine RFID significa identificazione a radiofrequenza (in inglese, Radio Frequency Identification) e si riferisce a un insieme di sistemi che permettono di identificare gli oggetti in modo automatico. Il modo più intuitivo per immaginare l'Rfid è pensare a un codice a barre capace di scambiare informazioni via radio, e anche di aggiornarsi nel corso del tempo. Molti pensano che l'Rfid sia, insieme alle sensor networks, la tecnologia che permetterà di realizzare "l'Internet delle Cose", ovvero una grande rete in cui non soltanto le persone, ma anche gli oggetti saranno collegati fra di loro. Le potenzialità di questo concetto sono state per il momento soltanto sfiorate e ci si aspetta che nei prossimi anni un gran numero di applicazioni innovative legate alla tecnologia RFID dal momento che essa può essere considerata *general purpose* e presenta un elevato livello di pervasività, ovvero una volta trovata una applicazione in un punto della filiera, l'applicazione ed i suoi benefici si propagano velocemente a monte e a valle della stessa.

3.1 Componenti del sistema RFID

Un sistema RFID è composto da tre elementi fondamentali: Sistema di Gestione, Reader, TAG.

- Sistema di Gestione: è il sistema centrale che gestisce le informazioni ottenute dai TAG tramite le interrogazioni dei Reader. Il suo compito è quello di rendere disponibili tali informazioni (normalmente codici identificativi univoci dei TAG) all'applicazione che fa uso del sistema RFID.

- Reader: è un ricetrasmettitore al cui interno è presente un microprocessore; è dotato di memoria ed è collegato ad un?antenna attraverso

Figura 3.1: Alcuni esempi di tag RFID passivi.

la quale invia il segnale di interrogazione ai vari TAG interessati e riceve il segnale di risposta dagli stessi. Il Reader è la parte più delicata dell'intero sistema in quanto ha il compito di realizzare le interrogazioni, in base alle necessità dell'applicazione, e di risolvere le eventuali collisioni in seguito a risposte contemporanee di più TAG. Il segnale a radio frequenza inviato dal reader deve essere di tipo particolare e a seconda del tipo di TAG con il quale deve avvenire la comunicazione sarà scelta una opportuna modulazione.

- TAG: detti anche Trasponder a radiofrequenza, sono costituiti da un circuito integrato di piccole dimensioni (microchip), capace di svolgere semplici funzioni logiche di controllo, il quale è connesso ad un'antenna ed è dotato di memoria limitata a causa della modesta quantità di dati contenuti in un sistema RFID che di solito è dell'ordine di qualche centinaia di byte o al massimo qualche kbyte.

In base alla gestione delle fonti energetiche i tag si distinguono in :

- *Passivi* (Figura 3.1): non possiedono una fonte propria di energia, ma ricavano quest'ultima dal segnale incidente proveniente dal Reader e lo re-irradiano modulandolo opportunamente. Questa energia però è minima a causa del fatto che la potenza emessa dal Reader stesso, regolamentata da norme restrittive degli organismi internazionali, risulta essere molto bassa ed inoltre diminuisce anche con l'aumentare della

Figura 3.2: Alcuni esempi di tag RFID semi-passivi.

distanza. Tale problema è il motivo per il quale la potenza computazionale di tali TAG è minima e si limita ad una logica di base capace di eseguire semplici istruzioni tra le quali leggere e scrivere dati nella memoria non volatile EEPROM (di capacità limitata). Inoltre, per lo stesso motivo anche la temporizzazione è ricavata direttamente dal segnale proveniente dal Reader. I TAG in questione sono realizzati in geometrie sempre più ridotte, questo grazie alla miniaturizzazione dei componenti in essi presenti, caratteristica che ne consente una riduzione dei costi. Questi tipi di TAG vengono utilizzati nelle Bande LF, HF, UHF.

- *Semi-Passivi*(Figura **??**): sono dotati di una piccola fonte di energia costituita da una batteria che ha il compito di alimentare soltanto il microchip, mentre per l'alimentazione del trasmettitore funzionano nello stesso modo dei TAG Passivi. La presenza della batteria consente a questi TAG di effettuare operazioni più complesse e di funzionare anche quando non ricevono segnale dal Reader. La capacità di eseguire funzioni più complesse consente a questi TAG di utilizzare memorie più grandi e, se previsto dall'applicazione, sensori opportuni. Anche per questi TAG la distanza operativa è limitata e le distanze massime arrivano intorno ai 10m, caratteristica che ovviamente incide sul costo del singolo TAG. Questi tipi di TAG vengono utilizzati nella Banda UHF.

- *Attivi* (Figura 3.3): Tali TAG inglobano al loro interno una batteria che ha il compito di alimentare sia il ricevitore che il trasmettitore, cosi come nei Reader. La presenza della batteria al loro interno consente a

Figura 3.3: Alcuni esempi di tag RFID attivi.

questi TAG di montare memorie di capacità molto superiori rispetto a quelle presenti nei TAG passivi, ed inoltre, consente di raggiungere distanze operativa di gran lunga superiori, dell'ordine di qualche Km, e di montare eventuali sensori come i TAG semi passivi. Il costo di tali TAG risulta di molto superiore rispetto ai precedenti, caratteristica che ne ha determinato il riutilizzo più volte all?interno delle stesse applicazioni. Questi tipi di TAG vengono utilizzati nella Banda UHF.

Il costo di un TAG, come si può immaginare, dipende quindi dal tipo e solitamente i TAG passivi risultano i dispositivi di minor costo. Per quanto riguarda le dimensioni queste dipendono ovviamente dall'applicazione, infatti ne troviamo di diversi tipi e dimensioni ad esempio inseriti in piccoli oggetti di plastica, all'interno di carte di credito, in etichette adesive, fogli di carta, ecc. I TAG inoltre possono essere di tipo read-only o read-writable, questi ultimi consentono la lettura la modifica o la riscrittura dell'informazione in essi memorizzata. Per quanto riguarda i TAG attivi o semi passivi, oltre alla maggior quantità di memoria ed alla funzione di riscrivibilità della stessa, hanno la capacità di svolgere particolari funzioni come radiolocalizzazione e misura di parametri ambientali tramite sensori. Quindi la differenza tra TAG Attivi e Semi-Passivi non è tanto nelle funzioni di memoria o negli eventuali sensori, quanto nel fatto che i TAG attivi sono dei veri e propri apparati ricetrasmittenti mentre i TAG semi passivi sfruttano la tecnologia di trasmissione dei TAG passivi e pertanto necessitano di risorse di alimentazione modeste.

3.2 Classificazione per frequenze e applicazioni

Le frequenze utilizzate nei sistemi RFID dipendono fondamentalmente dal tipo di applicazione e dal tipo di TAG, ma anche dalla regione geografica in quanto le regolamentazioni sono differenti per regione dove le imposizioni più restrittive delle organizzazioni nazionali prevalgono sugli standard imposti dagli organismi internazionali, a causa ad esempio del differente utilizzo dello

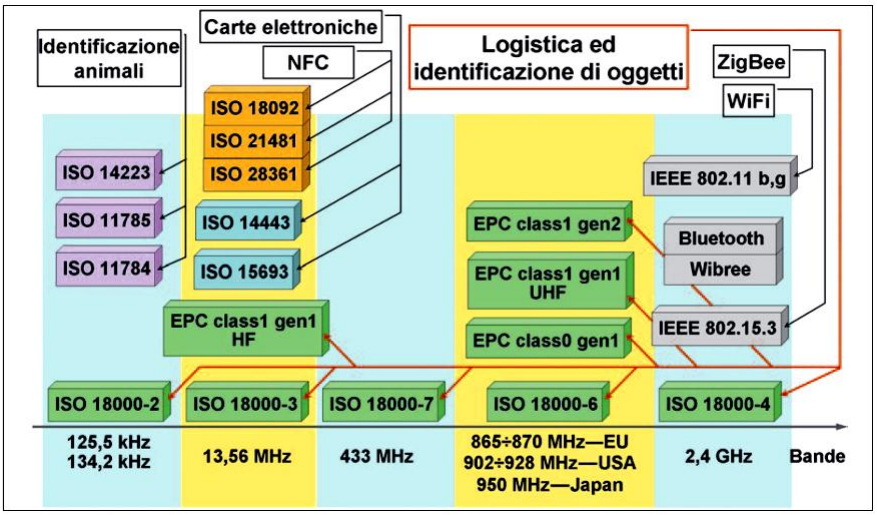

Figura 3.4: Frequenze di riferimento per la tecnologia RFID.

spettro di frequenze a livello nazionale. Le bande maggiormente utilizzate in questa tecnologia sono:

- Banda LF (Low Frequencies) e, principalmente, la sottobanda 120-145 kHz.
- Banda HF (High Frequencies) e, principalmente, la sottobanda 13,56 MHz.
- Banda UHF (Ultra High Frequencies) BASSA e, principalmente, la sottobanda 433-435 MHz.
- Banda UHF (Ultra High Frequencies) MEDIA e, principalmente, le sottobande 865-870 MHz in Europa, 902-928 MHz in USA, 950 MHz in Asia.
- Banda UHF (Ultra High Frequencies) ALTA centrata su 2,4 GHz.
- Banda SHF (Super High Frequencies) centrata su 5,8 GHz.

L'Utilizzo delle banda 13,56 MHz è accettato in tutto il mondo, diversamente dalle bande UHF media e alta dove bisogna convivere con applicazioni già esistenti quali rispettivamente la telefonia cellulare per la prima, e il WiFi Bluetooth ZigBee per la seconda. La scelta della frequenza di lavoro influisce sulla distanza (range) di operatività del sistema, sulle interferenze con altri sistemi radio, sulla velocità di trasferimento dei dati e sulle dimensioni dell'antenna. I sistemi che lavorano a frequenze più basse sono spesso basati su TAG passivi e sono in grado di trasmettere dati a distanze massime dell'ordine del metro e mezzo. Nei sistemi a frequenze più elevate, invece, possono

essere utilizzati sia TAG passivi che Attivi. I primi hanno l'inconveniente di essere limitati a pochi metri nelle distanze operative, mentre i secondi possiedono range operativi maggiori anche se limitati dai valori massimi di potenza irradiata stabiliti dagli organismi internazionali che regolano l'uso dello spettro radio. Per sistemi che utilizzano una frequenza più alta la velocità di trasferimento dati è generalmente maggiore, mentre la dimensione delle antenne si riduce, il che consente di costruire TAG più piccoli.

3.2.1 TAG induttivi LF 120-145kHz

La comunicazione Reader-TAG avviene tramite accoppiamento induttivo, stesso principio sfruttato nei trasformatori elettrici. Le frequenze utilizzate all'interno di tale banda sono due: 125,5 kHz e 134,2 kHz a seconda del tipo di applicazione. La prima è utilizzata nel settore auto motive con velocità massime nella trasmissione dei dati non superiori a un migliaio di bit al secondo a causa della bassa frequenza della portante, mentre la seconda in applicazioni per la tracciabilità animale grazie alla scarsa influenza dellacqua e dei tessuti nella trasmissione. I TAG utilizzati sono di tipo passivo e consentono di ottenere distanze operative che variano dai 30cm a 1m, distanza che diminuisce anche del 30-50% in fase di scrittura della memoria del TAG. Oltre tali distanze il campo si riduce molto rapidamente. Gli standard di riferimento per questo tipo di TAG sono *ISO 11784*, *ISO 11785* ed *ISO 14223*, *ISO 18000-2*.

3.2.2 TAG indutivi HF 13,56 MHz

La comunicazione Reader-TAG avviene tramite accoppiamento induttivo, ugualmente al caso dei TAG induttivi LF. I tipi e le forme disponibili di questi TAG sono diverse e variegate ma in comune hanno sempre un'antenna formata da un certo numero di avvolgimenti (il numero dipende da parametri come la distanza operativa e la potenza emessa dal Reader), realizzati in rame o anche alluminio, stampata su un substrato. Esempi di applicazione importante di questo tipo di TAG sono le "Smart Card contactless" , dette anche carte intelligenti, che sono di tipo passivo e i cui standard di riferimento sono l'ISO/IEC 14443 per distanze dai 10 ai 30 cm e l'ISO/IEC 15693 per distanze dai 30 ai 90cm. Queste carte trovano applicazione in carte di credito, carte per il controllo dell'accesso, ticketing.

NFC Un altro esempio sono gli NFC (Near Field Communication) che sono costituiti da apparati RFID e da carte senza contatto. Sostanzialmente un sistema NFC ingloba in un unico circuito sia il Reader che il TAG che può essere attivo o passivo. I dispositivi NFC si prestano, quindi, ad applicazioni di comunicazioni peer-to-peer che vedono lo scambio di transazioni tra due utenti con velocita' massima di trasmissione dati di 424 kBps, tuttavia sono

in grado di funzionare in modalità peer to peer, da reader o in emulazione di tag . Tale sistema consente una distanza operativa corta inferiore ai 10cm. Gli standard di riferimento per gli NFC sono **ISO 18092**, *ISO 21481*, *ISO 28361*, *ISO 18000-3*.

3.2.3 TAG elettromagnetici UHF Media 860-950 MHz

La comunicazione Reader-TAG avviene attraverso la propagazione di un campo elettromagnetico. Le distanze operative raggiungibili con tali TAG risultano essere dell'ordine dei 5m, e in alcuni casi si spingono anche oltre gli 8m, caratteristica questa che porterà tali apparati a ricoprire il compito della gestione degli oggetti. Tali TAG hanno capacità di trasmissione dati superiori a quelle dei TAG a frequenze più basse e rendono possibili anche selezioni multiple. L'utilizzo di questa banda comporta il dover risolvere alcuni problemi. Il primo è costituito dal fatto che tale banda risulta già occupata dalla telefonia cellulare e le bande disponibili agli RFID risultano diverse a seconda se la regione geografica è l'Europa, l'ASIA o l'USA. Per ovviare a tale problema si utilizzano TAG passivi capaci di ricevere a "larga banda" cosa che ne consente l'uso su bande differenti, oppure si utilizzano TAG attivi con ricevitori che possono essere tarati su più frequenze, ma tale soluzione aumenta il costo del singolo TAG. Un secondo problema sull'utilizzo di tale banda è anche il fatto che le potenze massime ammissibili per gli RFID differiscono anche queste a seconda della regione geografica. Terzo e ultimo problema è la presenza di riflessioni multiple dovute alla presenza di ostacoli, e la presenza di liquidi nell'ambiente di utilizzo, cosa che fa disperdere parte del campo elettromagnetico che viene assorbito appunto dai liquidi. Lo standard di riferimento è ISO 18000-6.

3.2.4 TAG elettromagnetici UHF Alta e SHF 2,4GHz

Tali TAG essendo a frequenza molto alta consentono una miniaturizzazione del componente, specifica importante che può anche risultare dannosa per la riduzione nella capacità di captare il campo elettromagnetico incidente, questo, quindi, comporta l'utilizzo di campi più direzionali. Il problema delutilizzo di tale banda è il fatto che anche qui sono già presenti altre tecnologie quali il WiFi, Bluetooth e ZigBee con le quali bisogna convivere. Lo standard di riferimento è ISO 18000-4.

3.3 Funzionamento dei tag passivi e semi-passivi

Per quanto riguarda il funzionamento della parte a radio frequenza è lo stesso per sia per i tag attivi che per quelli passivi, questi ultimi come detto in precedenza implementano in più altre funzionalità, normalmente di sensing.

Caratteristica fondamentale dei TAG passivi e semi-passivi, come già detto precedentemente, è l'alimentazione che avviene prelevando energia direttamente dal segnale a radio frequenza che incide su di essi. Questi TAG non generano da soli la frequenza portante del segnale a radio frequenza che utilizzano in fase di trasmissione, ma essi re-irradiano, modulandolo opportunamente, il segnale a radio frequenza precedentemente ricevuto dal Reader, e in questo modo si elimina la necessità di oscillatori locali per generare la portante di trasmissione. Tale modulazione è effettuata dal TAG facendo variare l'impedenza della propria antenna in modo da farla diventare assorbente o riflettente. I principi su cui si basa laccoppiamento tra Reader e TAG passivi sono due:

- Accoppiamento induttivo (magnetico) in condizioni di *campo vicino*;
- Accoppiamento elettromagnetico in condizioni di *campo lontano*.

3.3.1 Accoppiamento induttivo

I sistemi di questo tipo di questo tipo si basano sul fatto che, per distanze relativamente brevi rispetto alla lunghezza dell'onda emessa dallantenna del Reader, nell'antenna del TAG prevalgono gli effetti della corrente indotta dal campo magnetico che varia periodicamente nel tempo. Poiché il TAG viene a trovarsi immerso in questo campo magnetico, il flusso magnetico variabile nel tempo si concatena con le spire dell'antenna del TAG dando così origine, secondo la ben nota legge fisica di Lenz, ad una corrente indotta nelle spire. L'accoppiamento induttivo tra le antenne del TAG e del Reader avviene qavviene quindi in maniera simile ad un trasformatore. L'energia ricavata da un siffatto trasformatore viene usata per attivare il TAG. Il massimo accoppiamento tra i due avvolgimenti si ha quando lantenna ricevente è sintonizzata alla portante f_0 irradiata dal reader. Viene inserito un condensatore di tuning C_t in parallelo al coil in modo che la frequenza di risonanza del circuito in ricezione sia:

$$f_0 = \frac{1}{2\pi\sqrt{L_t C_t}}$$

Per ottenere le condizioni di "campo vicino" alle distanze operative impiegate vengono generalmente sfruttate le bande con maggiore lunghezza d'onda, ovvero LF ed HF.

3.3.2 Accoppiamento elettromagnetico

Sistemi di questo tipo si basano sul fatto che, per distanze relativamente lunghe rispetto alla lunghezza dell'onda emessa dall'antenna del Reader, nell'antenna del TAG prevalgono gli effetti del campo elettromagnetico, che varia periodicamente nel tempo. L'antenna del TAG riflette parte della potenza elettromagnetica ricevuta, fenomeno detto back-scattering, che a sua

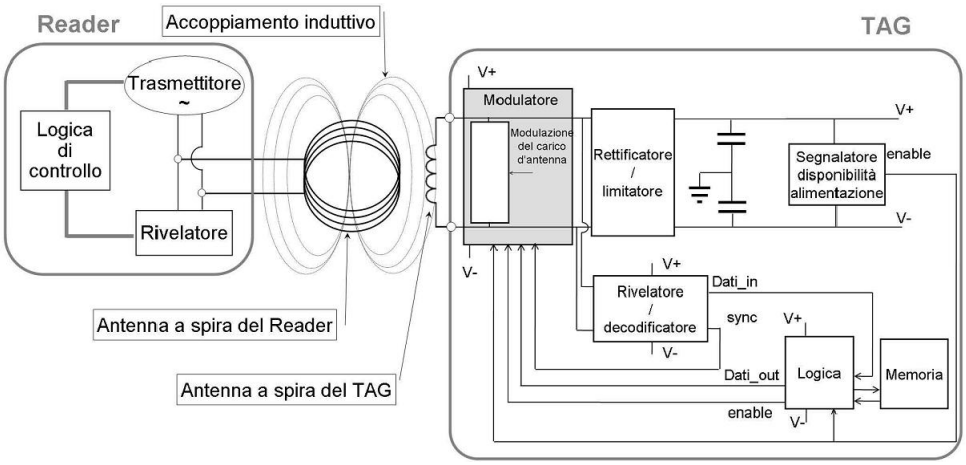

Figura 3.5: Accoppiamento induttivo (campo vicino).

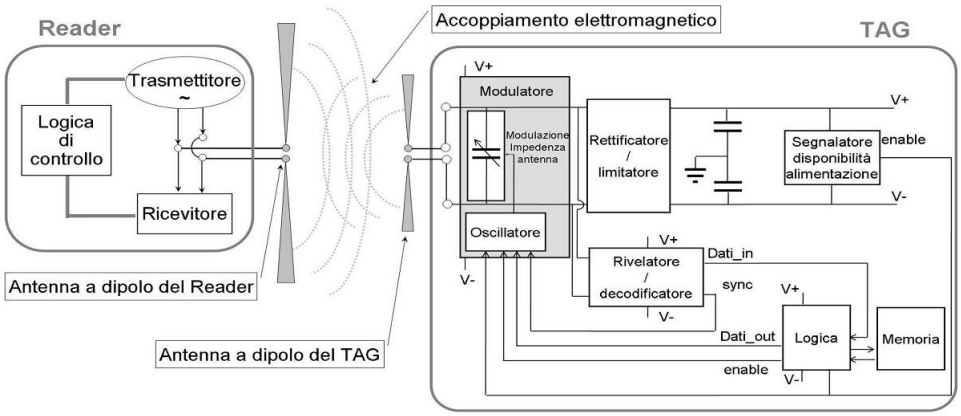

Figura 3.6: Accoppiamento elettromagnetico (campo lontano).

volta può essere rilevata dall'antenna del Reader. Per ottenere le condizioni di "campo lontano" alle distanze operative vengono utilizzate le bande con minore lunghezza d'onda, quali UHF e SHF.

Un sistema RFID backscatter è fondamentalmente un sistema radar in cui il lettore (transceiver radar) fornisce il segnale di frequenza radio per le comunicazioni in entrambe le direzioni. Il tag non ha una fonte di generazione di potenza del trasmissione, ma utilizza il segnale ricevuto dal lettore sul quale modula la sua risposta. Nei sistemi passivi la potenza del segnale ricevuto dal lettore viene utilizzata anche per fornire le tensioni di funzionamento necessarie per i circuiti all'interno del tag. Il tag può cambiare sia l'ampiezza o la fase del segnale re-irradiato a seconda che la parte reale o reattiva dell'impedenza sia cambiata.

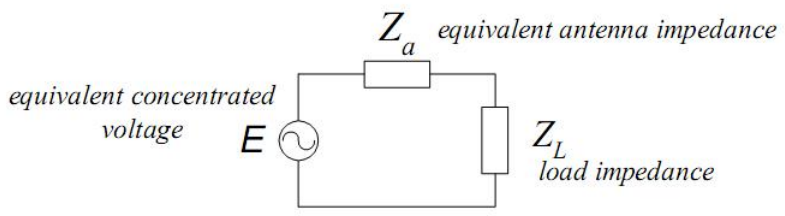

Figura 3.7: Circuito equivalente del tag.

Calcolo del coefficiente di riflessione L'antenna del tag ha un'impedenza legata alla radiazione resistenza R_r. Il chip può avere una componente reattiva. Per avere il massimo trasferimento di energia al chip, l'impedenza del chip deve essere un coniugato dell'impedenza dell'antenna. Tuttavia è importante ricordare che i circuiti logici di un chip utilizzato in un tag hanno un assorbimento di potenza molto piccolo rispetto alla quantità di energia consumata dal circuito di input RF del chip. Lo scopo è spesso quello di massimizzare la tensione recuperata.

L'antenna può essere considerata come un generatore di tensione E che ha una impedenza interna Z_a. Il rapporto tra la tensione emessa E e la potenza delle onde radio incidenti è detta altezza efficace dell'antenna (es.: $E = \epsilon h$ dove h è l'altezza efficace dell'antenna), mentre l'impedenza Z_a equivalente è il circuito equivalente circuito che il carico vede quando guarda verso l'antenna. La corrente che scorre in questo circuito è identica a quella che fluisce nell'antenna nel punto in cui viene inserito il carico.

Quando la parte reattiva dell'antenna e impedenza di carico sono desintonizzati e ignorando la perdita di resistenza antenna, la potenza totale in watt estratta dalle onde radio è data da:

$$P_r = \frac{(\epsilon h)^2}{R_L + R_r}$$

dove:

- ϵ= Potenza intensità del campo del segnale radio in Volt per metro
- h= Altezza effettiva dell'antenna in metri
- R_r= Resistenza di radiazione dell'antenna
- R_L= Resistenza di carico

La parte $R_L/R_r + R_L$ dell'energia totale viene passata al carico mentre la parte $R_r/R_r + R_L$ dell'energia totale viene re-irradiata. Da questo si può vedere che quando il carico è adattato perfettamente con l'antenna il 50% dell'energia viene assorbito e il 50% dell'energia viene re-irradiato.

Poichè il modulatore commuta tra due stati, l'impedenza di carico Z_L passa tra due stati. La riflessione a causa del una mancato adattamento tra l'antenna e il carico in un tag backscatter è analoga alla riflessione trovata nelle linee di trasmissione e può essere espressa in termini di coefficiente di riflessione. Il coefficiente di riflessione ρ varia quindi appena il modulatore cambia lo stato. Il coefficiente di riflessione è dato da:

$$\rho = \frac{\frac{Z_L}{Z_0} - 1}{\frac{Z_L}{Z_0} + 1}$$

Quando l'impedenza dell'antenna è puramente resistiva:

$$\rho = \sqrt{\frac{(R_L - R_r)^2 + X_L^2}{(R_L + R_r)^2 + X_L^2}}$$

dove:

- R_L è la parte resistiva del carico;
- X_L è la parte reattiva del carico;
- R_r è la resistenza di radiazione dell'antenna.

Pertanto il coefficiente di riflessione varia tra quando il modulatore è acceso e quando è spento.

In sintesi, quando $R_L = R_r$ il tag in grado di assorbire la metà della potenza incidente e re-irradiare l'altra metà. Quando $R_L < R_r$ l'antenna re-irradia più del 50% della potenza e quando l'antenna è in corto circuito viene re-irradiato il 100%. Quando l'antenna è a circuito aperto invece, è trasparente per le onde radio incidenti e non ne assorbe nè ne riflette la potenza.

Calcolo del link budget per modulazione Amplitude Shift Keying
Il reader trasmette un'energia P_t che investe il tag ed è modulata usando la modulazione di impedenza che cambia la riflessività del tag come descritto precedentemente. Un target di *cross section* σ in metri quadrati intercetta la potenza proveniente dall'antenna in accordo con la seguente formula:

$$P_i = \frac{P_t G_t}{4\pi R^2}\sigma$$

dove P_i è la potenza intercettata espressa in Watt. La potenza intercettata è re-irradiata in modo che al ricevitore radar, dopo il viaggio di ritorno a distanza R, la densità di potenza P_d al radar è

$$P_d = \frac{P_t G_t}{4\pi R^2}\frac{\sigma}{4\pi R^2}$$

Un'antenna di guadagno G ha un'apertura effettiva pari a:

$$A_e = \frac{G\lambda^2}{4\pi}$$

La Radar Cross Section (RCS) dell'antenna del tag è equivalente all'apertura effettiva dell'antenna A_e quando l'antenna è adattata, mentre dalle precedenti equazioni la Radar Cross Section quando l'antenna è disadattata risulta essere:

$$\sigma = \frac{\lambda^2}{4\pi} G^2 \rho^2$$

dove:

- G è il guadagno dell'antenna del tag (è al quadrato perchè il segnale è ricevuto e re-irradiato);
- λ è la lunghezza d'onda;
- ρ è il coefficiente di riflessione del modulatore del tag.

Combinando le precedenti equazioni la potenza ricevuta all'antenna del tag può essere espressa da:

$$S = \frac{P_t G_t^2 \sigma \lambda^2}{(4\pi)^3 R^4}$$

3.3.3 Funzionamento dei tag attivi

Da un punto di vista hardware i tag attivi hanno la stessa struttura dei dispositivi utilizzati come nodi di una wireless senor network, essi sono infatti costituiti da un circuito dalle dimensioni il più contenute possibili basato su un classico microcontrollore e un transciever a radiofrequenza (normalmente questo tipo di tag opera alle frequenze intorno a 2.4 Ghz). L'alimentazione a batteria ha in questo caso lo scopo di alimentare il transciever che a differenza delle altre tipologie di tag è in grado di emettere segnale. Grazie a questa caratteristica i tag attivi permettono distanze di lettura notevolmente superiori (nell'ordine di diverse decine di metri). Dal punto di vista software tali tag non implementano protocolli di rete ma rispettano in tutto e per tutto i principali standard della tecnologia RFID per le frequenze alle quali si riferiscono. Tali dispositivi sono inoltre dotati normalmente di alcune periferiche di sensing in funzione dell'applicazione a cui sono destinati.

3.4 Applicazioni

La tecnologia RFID è costituita da un grande insieme di micro-apparati che si differenziano principalmente per gli standard e le frequenze di riferimento:

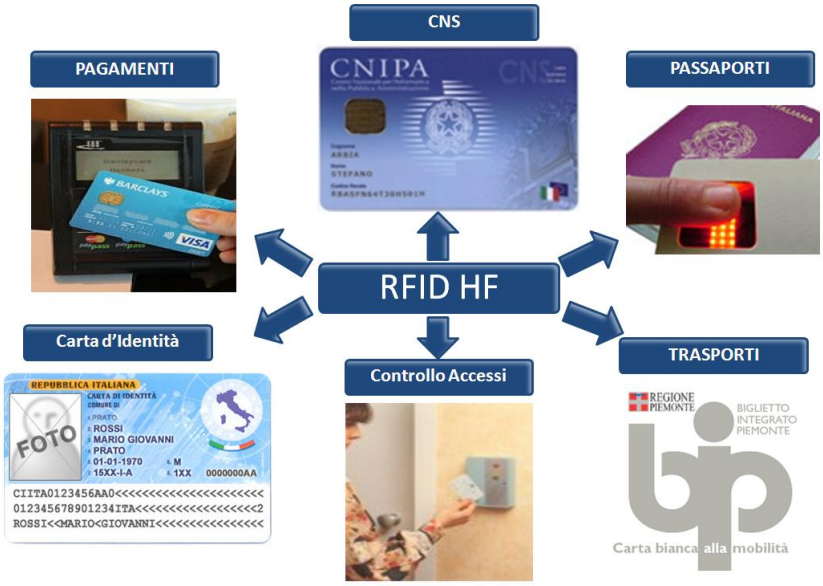

Figura 3.8: Applicazioni della tecnologia RFID HF.

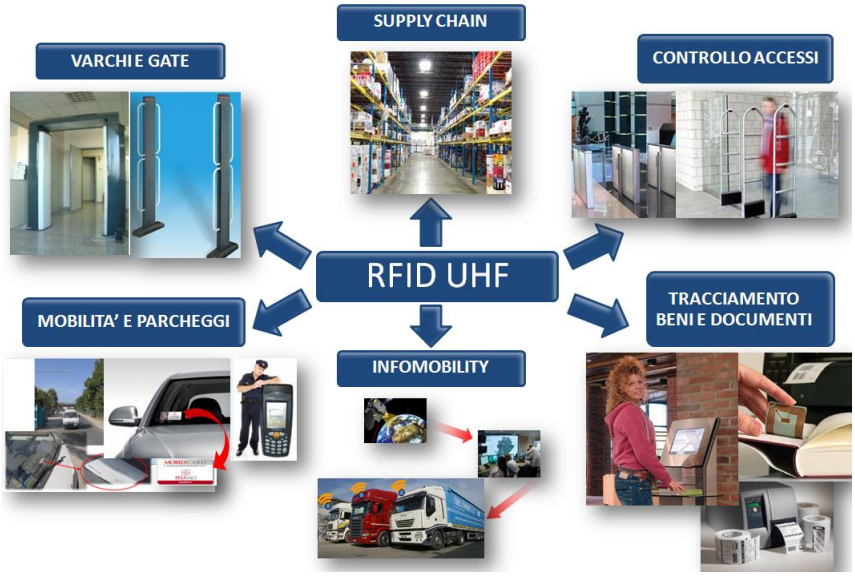

Figura 3.9: Applicazioni della tecnologia RFID UHF.

Applicazione	Informazione contenuta nel TAG	Distanza operativa	Tecnologia
Gestione merci	Identificativo (EPC)	Fino a qualche metro	TAG RFID UHF
Biglietti e pedaggi	Valore monetario e regole di pagamento	Fino a 10cm	TAG RFID HF e NFC
Carte di pagamento	Chiavi di sicurezza	Fino a 10 cm	Smart card RFID HF e NFC
Documenti elettronici	Dati personali	Fino a 10 cm	Smart card RFID HF

Tabella 3.1: Applicazioni e tecnologie utilizzate

- Le soluzioni su tag 125/134 kHz trovano diversi campi applicativi: tracciabilità animali domestici e di allevamento. Immobilizer per auto (sono ormai contenuti in tutte le chiavi di apertura e avviamento delle auto, moto, camion, ecc.), apertura serrature (settore alberghiero e controllo accessi).
- standard ISO 15693 fino ad 1m, per la tracciabilità (alimentare, prodotti, etc), borsellini elettronici non bancari (villaggi vacanze, discoteche, logistica in generale, etc);
- standard ISO 14443 fino a 10cm, (ad alta sicurezza) per carte bancarie, tessere documenti di identità elettronici, titoli di viaggio elettronici, sistemi di bigliettazione elettronica per metropolitane, treni, autobus, moneta elettronica per Macchine Distributrici prodotti alimentari, ecc.
- Standard NFC Near Field Communication, fino a 10cm, e' il nuovo standard ISO 18092 NFCIP-1 e ISO 21841 NFCIP-2, ma che con i recenti chips permette di leggere/scrivere anche tag ISO14443 e ISO15693, Felica e quindi divenendo un lettore/scrittore universale di tutti i protocolli HF, ed addirittura evolvendo il concetto di tag e Lettore/Scrittore potendo scambiare dati ad alta velocita' max 424kBps anche tra 2 lettori/Scrittori come possono essere 2 cellulari NFC e che sostituira' man mano i contanti e le carte di debito e credito, usando direttamente il proprio cellulare, per pagamenti Bigliettazione, pagamenti vari, ma potrà essere anche usato per scambio dati tra cellulari come foto,

Header	EPC Manager	Object Class	Serial Number
36	28	24	36

Figura 3.10: Codice EPC.

video, biglietti da visita, ecc. Ne sentiremo sempre più parlare nel prossimo futuro.

- le soluzioni con tag UHF ISO 18000 in teoria fino a 10m, sono dedicate alla logistica sia interna che esterna d'azienda, ma anche per la protezione del marchio o prodotto da clonazione (brand Protection) ed in generale dove si necessita di tantissimi tag e pochi lettori in proporzione ai tag.
- le soluzioni con tag 2,4 GHz e oltre per la mobilità (Telepass e similari) e gli interporti.

3.4.1 Lo standard Electronic Product Code

Lo standard EPC (Electronic Product Code) definito da EPC-Global Inc. è lo standard che a livello mondiale definisce le linee guida per L'Applicazione della tecnologia RFID. L'EPC è nato come soluzione per l'identificazione e il tracciamento dei prodotti attraverso l'utilizzo di Tag RFId e prevede di identificare un oggetto secondo un codice di 96 bit suddivisi come in figura **??**in cui:

- Header: identifica la versione dell'EPC e permette diverse versioni o tipi:
- EPC Manager: identifica il produttore dell'oggetto al quale il tag è associato;
- Object Class: identifica il tipo di prodotto;
- Serial Number: rappresenta l'identificativo unico del prodotto.

Lo standard EPC non definisce solo il tipo di veicolo di identificazione ma anche le tecnologie di rete necessarie per garantire la tracciabilità dei prodotti lungo la supply chain secondo lo stack mostrato in Figura **??**. Lo standard infatti introduce, oltre all'Elettronic Product Code, l'Object Name Service (ONS) e l'EPC Information Service (EPC-IS).

L'ONS è una rete globale aperta per la rintracciabilità delle merci e fornisce l'infrastruttura complementare alla memorizzazione dei dati, dal momento che il trasponder memorizza esclusivamente l'EPC. L'Object Name Server presenta funzionalità di networking analoghe al DNS (Domain Name Server): sulla base del codice EPC ricevuto, fornisce l'indirizzo del

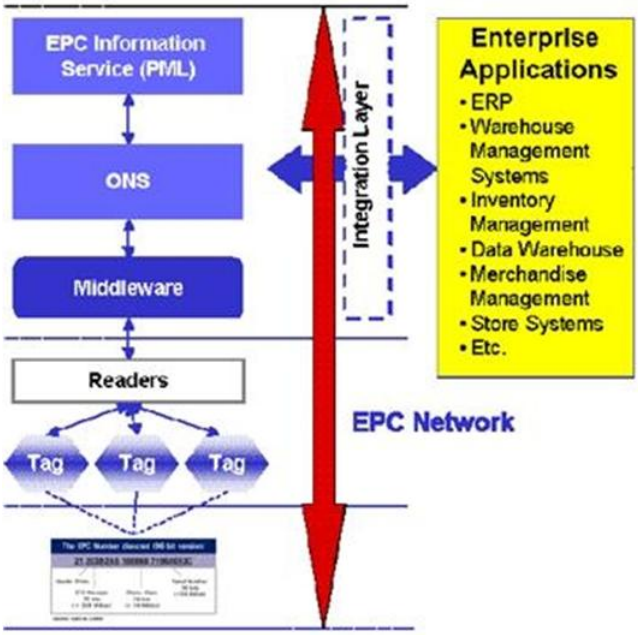

Figura 3.11: Stack EPC global.

sistema sul quale risiedono le informazioni relative al prodotto. EPC-IS ha lo scopo di definire interfacce comuni, per il software RFId, in grado di porre le aziende nella condizione di scambiare e utilizzare i dati RFId indipendentemente dall'applicazione con la quale sono stati creati o archiviati. Per le aziende operanti lungo tutte le fasi della supply chain, ciò significa avere la possibilità di acquisire in maniera conveniente grandi volumi di dati dettagliati a ogni livello della supply chain e di condividerli facilmente con i propri partner. E' facile dunque capire l'importanza dell'adozione dello standard Electronic Product Code Information Service (EPCIS) per lo scambio e la consultazione dei dati memorizzati nei tag RFID. Per quanto riguarda i Tag RFId l'identificazione univoca è garantita dal General Identifier (GID-96) al cui interno oltre al General Managment Number (numero identificativo dell'azienda) e l'Object Class (tipologia di oggetto prodotto), c'è il Serial Number che consente di identificare il singolo item di produzione

Capitolo 4

Integrazione tra WSNs e RFID

La Radio-Frequency Identification e le Wireless Sensor Network sono due importanti tecnologie wireless adottate in una vasta gamma di applicazioni. Gli RFID facilitano la rilevazione e l'identificazione di oggetti che non sono facilmente individuabili e riconoscibili utilizzando le tecnologie dei sensori convenzionali. Tuttavia, non forniscono informazioni sulle condizioni degli oggetti rilevati (oppure nel caso dei tag semi-passivi forniscono una sensoristica limitata a basso costo), inoltre ogni device lavora in isolamento e non in maniera collaborativa. Le WSN dal canto loro permettono invece di implementare applicazioni avanzate di sensoristica e sistemi cooperanti a fronte di un costo ovviamente maggiore. Quindi, l'integrazione di queste tecnologie si espande le loro funzionalità e le loro capacità complessive.

Come si può vedere dalla Tabella 4.1, le reti RFID e WSN rappresentano due tecnologie complementari e ci sono una serie di vantaggi a unirle.

Come detto le etichette RFID sono molto più economiche rispetto ai nodi sensore. In molti casi risulta economico utilizzare i tag RFID per sostituire alcuni dei nodi nella rete WSN. Inoltre, poichè un oggetto che è provvisto di un tag RFID è tracciabile, la tecnologia RFID fornisce una interessante aggiunta per le reti di sensori wireless nel tracciamento di oggetti che altrimenti sono difficili da rilevare. D'altra parte, le WSN offrono una serie di vantaggi rispetto ai tradizionali sistemi a tecnologia RFID da un punto di vista sia comunicativo che di logica computazionale. Da tutto ciò risulta che queste due tecnologie sono per molti aspetti complementari e la loro integrazione può quindi estendere maniera significativa le applicazioni e le potenzialità di entrambe.

Di seguito viene riportata una overview di stato sull'arte delle applicazioni che integrano gli RFID con le reti di sensori. Nell'overview presentata i sistemi integrati sono stati divisi in quattro classi:

- Integrazione di TAG RFID con sensori;

	WSN	RFID
Obiettivo	Rilevare parametri ambientali o fornire informazioni sulle condizioni dell'oggetto	Rilevare la presenza dell'oggetto
Protocolli	IEEE 802.15.4, Bluetooth	RFID standards
Comunicazione	Multihop	Single-hop
Mobilità	I nodi sono normalmente statici	I TAG si muovono con l'oggetto
Alimentazione	A batteria	I TAG sono alimentati a batteria o passivi
Costo	Sensore - medio Sink - alto	Reader - alto TAG - molto basso
Disposizione organi di rilevamento	Random o fissa	Fissa, richiede un attento posizionamento
Obiettivo del design	General Purpose	I TAG sono ottimizzati per un singolo scopo

Tabella 4.1: Confronto WSN-RFID

- Integrazione di TAG RFID con nodi WSN e dispositivi wireless;
- Integrazione di reader RFID con nodi WSN e dispositivi wireless;
- Sistemi ibridi di dispositivi RFID e WSN cooperanti.

4.1 Integrazione di TAG RFID con sensori

Questo tipo di integrazione consente l'aggiunta di funzionalità di sensing nei sistemi RFID (Figura 4.1), non si tratta di una vera e propria integrazione tra WSN e RFID in quanto tali dispositivi non hanno la possibilità di cooperare, vengono comunque normalmente catalogati come sistemi integrati dal momento che prevedono funzionalità aumentate rispetto ai tradizionali tag passivi. I tag RFID con sensori (sensor tags) usano gli stessi protocolli e meccanismi degli RFID per la lettura degli ID dei tag, così come per la

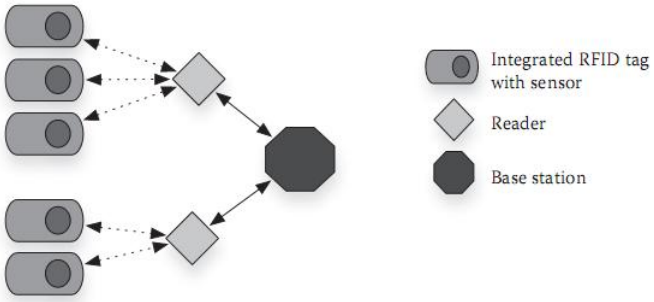

Figura 4.1: Integrazione di TAG RFID con sensori.

raccolta dei dati rilevati. Dal momento che i sensori integrati all'interno dei tag RFID sono utilizzati solo per scopi di sensing, attualmente i protocolli di questi tag RFID si basano sulla comunicazione singlehop. Si possono distinguere tre categorie di *sensor tags*:

Passivi : non sono alimentati, quindi la rilevazione dei dati da parte del sensore viene effettuata solo quando si trovano nel raggio d'azione dell'antenna del reader.

Semi-passivi : hanno una piccola fonte di alimentazione a batteria e sono in grado di rilevare i dati dai i sensori in qualsiasi momento e di creare un logging dei dati.

Attivi : sono alimentati a batteria, sono in grado di rilevare i dati dai sensori in qualsiasi momento e sono provvisti anche di un microcontrollore che permette di eseguire operazioni più complesse.

I sensor-tags si possono suddividere in tre categorie in funzione dell'alimentazione e cioè:

- Passivi - non sono alimentati e le funzioni di sensing sono attuate solo quando il tag è nel range di lettura del reader;
- Semi-passivi - utilizzano l'alimentazione a batteria per il logging dei parametri rilevati ma non per la comunicazione;
- Attivi - utilizzano l'alimentazione a batteria sia per il sensing che per la comunicazione.

4.2 Integrazione tra tag RFID e nodo WSN

Questo tipo di integrazione permette ai tag di comunicare non solo con il reader mediante l'interfaccia a radio-frequenza, ma anche con il dispositivo wireless della rete (Figura 4.2). Si può distinguere tra tre casi:

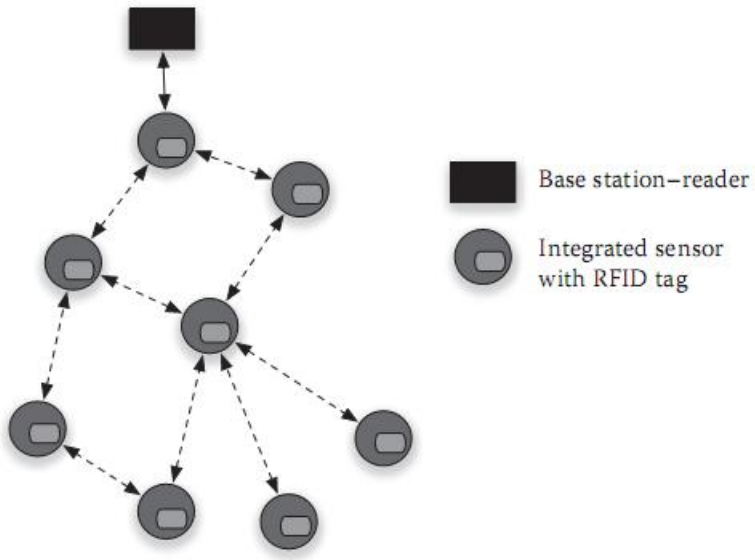

Figura 4.2: Integrazione tra tag RFID e nodo WSN.

- Nodi WSN con funzionalità RFID;
- Nodi WSN con interfaccia a radio-frequenza multipla (RFID e altri protocolli);
- Nodi WSN nei quali è integrato un tag RFID che ha lo scopo di ottimizzare il consumo energetico.

Di particolare interesse è la terza tipologia della quale è possibile trovare un esempio in [17], in quanto la durata delle batterie dei nodi è un problema molto sentito nelle WSN. Nello specifico il tag ha lo scopo di risvegliare il nodo nel momento in cui viene letto dal reader RFID. Il nodo può quindi rimanere in sleep mode quando non vi sia la necessità di comunicare con la rete e viene risvegliato quando deve ricevere o inviare un pacchetto. In questo modo il sistema è in grado di ottimizzare il consumo energetico delle batterie dei nodi.

4.3 Integrazione tra reader RFID e nodo WSN

Un'altra tipologia di integrazione è la combinazione di lettori RFID con i nodi della Wireless Sensor Network (*smart nodes*). I dispositivi risultanti sono in grado di rilevare le condizioni ambientali, comunicare tra loro in modalità wireless, leggere i numeri di identificazione degli oggetti taggati, e trasmettere in modo efficace queste informazioni nella rete WSN. Esempi di questo tipo di integrazione possono essere trovati in [18] e in [19]

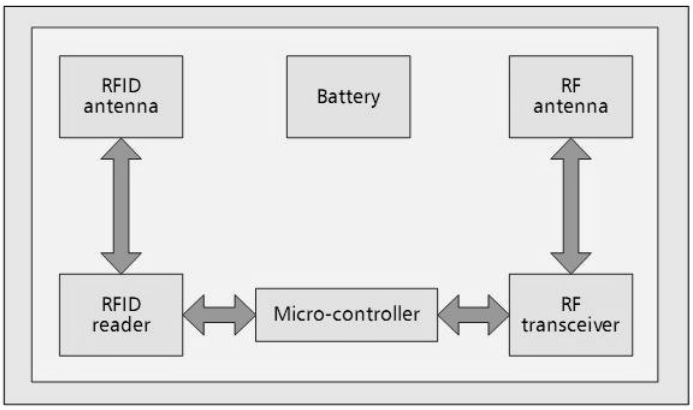

Figura 4.3: Schema di un dispositivo integrato reader RFID e nodo WSN.

Gli smart nodes integrati possono essere considerati come nodi sensori che possono essere utilizzati come lettori RFID con estese capacità di rilevamento. Gli smart nodes sono in grado di trasmettere informazioni e di essere configurati come nodi relay di una WSN. Sono in grado di comunicare tra di loro creando una rete di comunicazione ad hoc. Tali dispositivi sono in grado di funzionare come router e di trasmettere messaggi alla destinazione desiderata. Gli smart nodes si occupano di raccogliere i dati da semplici tag RFID nel loro range di funzionamento e di comunicare tra loro per trasmettere i dati alla base station in cui vengono raccolti ed elaborati tutti i dati . L'architettura di questa rete integrata, illustrata nella Figura 4.3, è simile a quella di clustering gerarchico basato su due livelli di WSN.

Questo tipo di strategia di integrazione offre nuove prospettive in molte applicazioni. I tradizionali lettori RFID UHF hanno infatti alcune limitazioni in qualche applicazione soprattutto per quanto riguarda la mobilità a causa del loro grande volume, e il posizionamento delle antenne che deve essere abbastanza accurato, mentre il nodo integrato intelligente è più piccolo, meno costoso e più facile da posizionare. Tuttavia, questa strategia di integrazione presenta anche alcuni importanti svantaggi, soprattutto per quanto riguarda sia l'elevato consumo energetico causato dal lettore RFID e sia lo squilibrio di consumo tra i vari nodi.

Un interessante campo da esplorare riguarda l'integrazione tra nodi WSN e reader HF/NFC dal momento che tali lettori hanno diversi vantaggi in termini dimensioni dell'antenna e potenza dissipata. Il sistema integrato potrebbe inoltre rendere disponibili all'interno della rete WSN i dati proveniente da carte RFID ISO1443 o da dispositivi NFC quali i cellulari di ultima generazione. Non vi è attualmente tuttavia molto in letteratura su tale tipologia di integrazione se non pochissimi esempi quali [20].

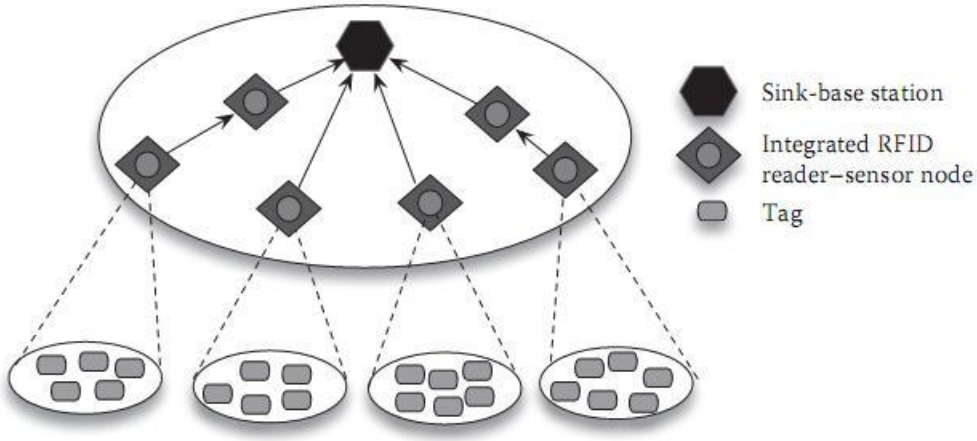

Figura 4.4: Integrazione tra reader RFID e nodi WSN.

4.4 Sistemi ibridi RFID-WSN

Nei sistemi misti i tag RFID e nodi sensore sono dispositivi fisicamente distinti ma coesistono in una una rete integrata e funzionano in modo indipendente. Il vantaggio principale di una tale architettura è il fatto che non c'è bisogno di progettare un dispositivo hardware integrato. Tuttavia, vi è la possibilità di interferenze nella comunicazione tra o tag i lettori RFID e i nodi sensore perché in questo caso tutti i dispositivi sono fisicamente distinti. Le procedure da seguire per evitare tali interferenza possono causare un ulteriore overhead del sistema.

Un sistema integrato RFID-WSN che segue l'architettura mista è composto da tre tipi di dispositivi:

- la smart station;
- i normali tag RFID
- i normali nodi WSN

Una smart station è un particolare dispositivo composto da un lettore RFID, un microprocessore e una interfaccia di rete. La smart station è alimentata da rete elettrica quindi non ha limiti sul consumo energetico ed è in grado di ricevere e processare le informazioni dei tag RFID e dei nodi sensore e di trasmetterle ad un host locale o ad una LAN remota.

Le informazioni dii tag RFID e dei nodi sensori possono essere trasmessi alla base station, poichè la smart station non ha vincoli di potenza, possono essere implementati anche le tradizionali architetture basate protocollo Internet. In questo modo, le smart station sono in grado di eseguire non solo

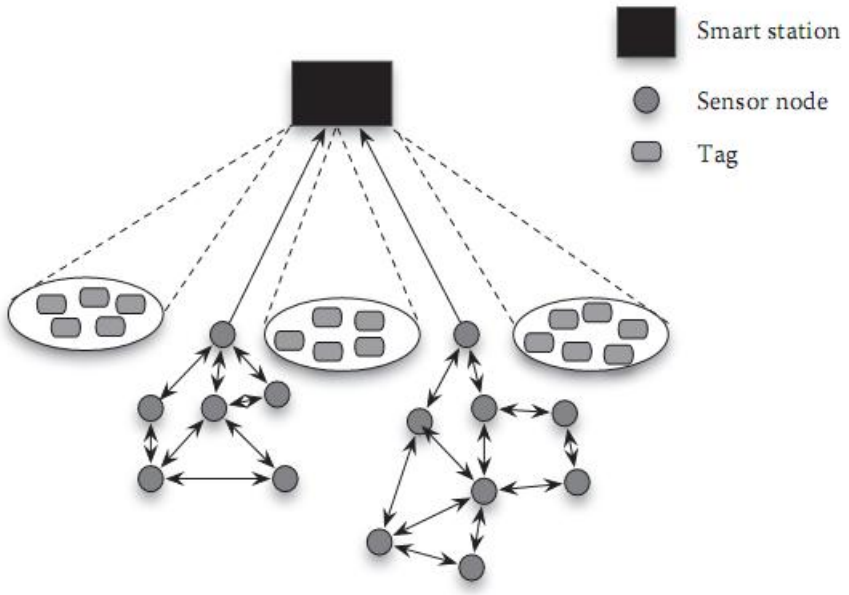

Figura 4.5: Architettura mista RFID-WSN.

l'elaborazione dei dati, ma anche protocolli di routing e protocolli di trasporto come TCP. Un protocollo di comunicazione che può essere utilizzato in un ambiente eterogeneo è la tecnologia 802.11/WiFi.

Capitolo 5

Scenario di riferimento

Come introdotto in precedenza l'obiettivo dell'attività di tesi è di:

- Fornire uno strumento di controllo degli accessi alle varie aree dello stabilimento;
- Aumentare il livello di sicurezza degli operai all'interno fornendo un meccanismo di controllo dei dispositivi individuali di protezione;
- Aumentare le funzionalità e le prestazioni dei sistemi tecnologici avanzati già esistenti.

Lo scenario di riferimento considerato per lo sviluppo del sistema è rappresentato da un tipico impianto industriale suddiviso in base al layout in diverse aree di lavoro come come si può vedere in figura 5.1.

In tale scenario è possibile dividere le varie aree dello stabilimento in funzione del livello di sicurezza richiesto, in modo da poter realizzare una classificazione delle persone autorizzate all'ingresso con i relativi vincoli dettati dagli equipments necessari alla sicurezza dell'operaio.

5.1 Requisiti del sistema

Requisiti del sistema sono:

- Controllo accessi nelle aree a rischio;
- Autorizzazione all'ingresso solo se in possesso dei dispositivi individuali di protezione richiesti e tutto il necessario per operare all'interno dell'area;
- Attivazione della specifica risorsa all'interno dell'area (ad es. un macchinario) solo se in possesso di autorizzazione;
- Integrazione con sistemi già esistenti di controllo dei parametri vitali dell'operaio e segnalazione di pericolo se necessario;

Figura 5.1: Esempio di layout di un impianto industriale.

- Integrazione con sistemi già esistenti di localizzazione della posizione degli operai all'interno dell'area;
- integrazione con sistemi di controllo e di attuazione già esistenti.

Il sistema deve inoltre garantire il minimo consumo energetico e quindi la rete deve disattivata in assenza di operai all'interno dell'area e attivata appena un operaio effettua l'accesso, stesso discorso per quanto riguarda il dispositivo in dotazione all'operaio per il quale deve essere garantito il funzionamento senza ricarica della batteria per lo meno fino alla fine del turno di lavoro dell'operaio.

5.2 Overview del sistema e building blocks

Il sistema proposto è costituito dai seguenti dispositivi:

tag RFID UHF passivi : possono essere attaccati ad oggetti per identificarli esattamente, vengono rilevati dal reader RFID che è in grado di leggerne i vari banchi di memoria;

nodo WSN semplice : è un dispositivo in grado di comunicare con la rete WSN, di rilevare i parametri ambientali dell'area (ad esempio temperatura, presenza di gas ecc.) e di smistare le informazioni;

nodo tag (nodo WSN integrato con tag RFID UHF passivo : si tratta di un dispositivo in grado di comunicare con la rete WSN, di essere

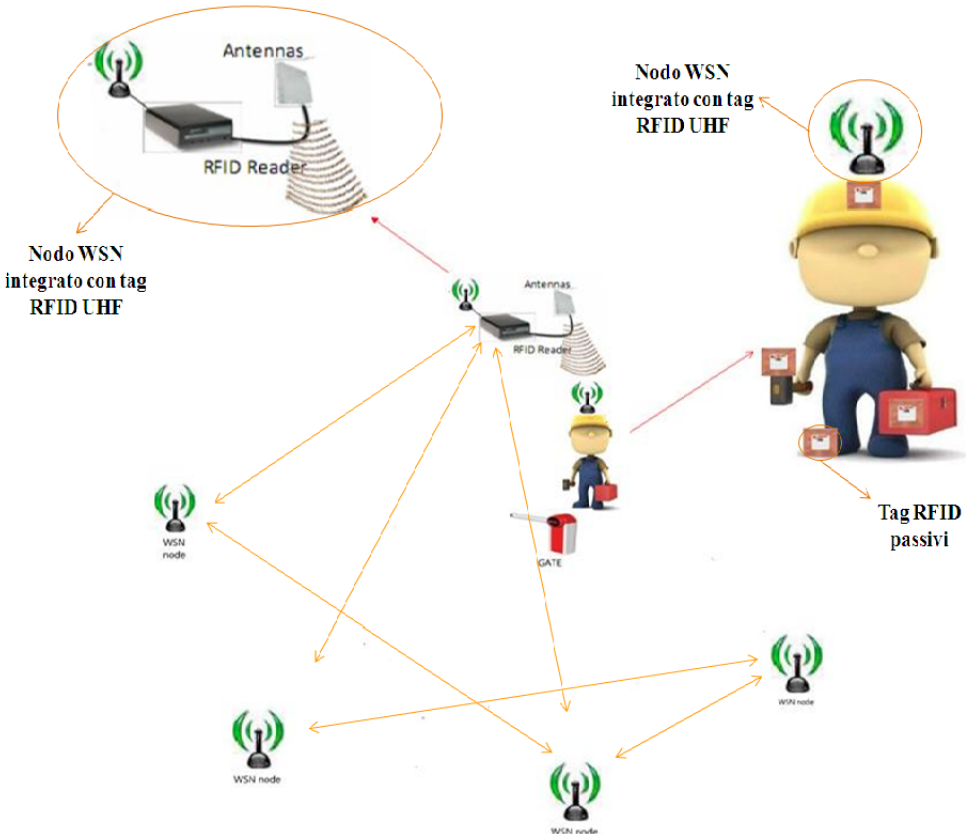

Figura 5.2: Schema di funzionamento del sistema.

letto dal reader RFID e di rilevare alcuni parametri vitali della persona che ne è in possesso attraverso vari sensori.

nodo reader (nodo WSN integrato con reader RFID UHF : è un dispositivo che permette di rilevare i tag RFID in possesso dell'operaio e di comunicare con la rete WSN e con un database centrale.

Al fine di ottenere gli obiettivi preposti e di soddisfare le specifiche il sistema è stato progettato come in figura 5.2.

I tag RFID sono fisicamente attaccati ad ogni dispositivo di protezione individuale in dotazione all'operaio, quindi devono essere piccoli, facili da attaccare e molto resistenti. Nella memoria dei tag è contenuto un codice che identifica la classe degli oggetti di quel tipo e l'oggetto in particolare. Ad ogni operaio viene fornito inoltre un "nodo tag" all'interno della cui memoria è contenuto il codice identificativo dell'operaio e l'informazione relativa agli oggetti che deve avere con sè per potere accedere all'area. Inoltre il "nodo tag" rileva i parametri vitali dell'operaio, comunica alla rete eventuali allarmi e scambia con la rete i dati necessari per permettere la localizzazione. Il "nodo tag" è normalmente spento e si attiva solo quando l'operaio viene rilevato da un "nodo reader".

All'ingresso di ogni area è invece installato un "nodo reader" all'interno del quale è contenuta la logica che permette l'autorizzazione all'accesso. Tale dispositivo rileva la presenza dell'operaio e confronta il suo codice identificativo con quelli contenuti in un database remoto, quindi rileva se sono presenti tutti i dispositivi di protezione individuale e se l'operaio è autorizzato e possiede tutti i DPI invia un segnale di apertura porte e risveglia la rete WSN dell'area.

Dal canto suo la rete WSN nel momento in cui un operaio entra nell'area inizia a rilevare i parametri ambientali e a comunicare con il "nodo tag" per localizzarlo e per ricevere eventuali allarmi.

All'ingresso di ogni area è inoltre installato un display che comunica all'operaio se ha o meno l'autorizzazione all'accesso e eventualmente i dispositivi di protezione di cui necessita.

5.3 Esempio: cabina di trasformazione

Ad esempio se consideriamo la cabina di trasformazione di uno stabilimento, ci sono normalmente poche persone autorizzate all'accesso per operazioni di manutenzione, e devono essere ad esempio dotate di scarpe e guanti isolanti adeguati secondo il documento di valutazione del rischio come spiegato nel Capitolo 1. In questo caso il sistema identifica l'operaio e confronta il suo codice identificativo con quelli contenuti nel database centrale. Se la persona è autorizzata all'ingresso il sistema controlla gli oggetti in suo possesso e se rileva le scarpe adatte lo autorizza. Nel momento in cui l'operaio entra

Figura 5.3: Esempio cabina di trasformazione.

la rete WSN accende le luci all'interno del locale e sblocca la chiusura dei quadri elettrici sui quali l'operaio può andare ad operare. In questo modo si può pensare di diversificare ulteriormente le autorizzazioni cioè ad esempio un lavoratore può agire solo su un particolare quadro elettrico piuttosto che su un altro e così via.

Capitolo 6

Componenti del sistema

Nel seguito del presente capitolo vengono presentati i componenti software e hardware del sistema sviluppato. Per quel che concerne le tecnologie di riferimento per sistemi RFID e WSN si è scelto:

- RFID UHF in banda 868 Mhz;
- WSN su standard IEEE 802.15.4.

I vari dispositivi sono stati quindi integrati per andare a costituire i building blocks descritti nel precedente capitolo.

6.1 Descrizione del software

L'ambiente di sviluppo software è diviso in software per sistemi embedded (sistema operativo real-time *ERIKA Enterprise*e tools Microchip per il supporto alla programmazione) e software per sistemi high-end (framework Qt).

6.1.1 ERIKA Enterprise

ERIKA (*Embedded Real tIme Kernel Architecture*) è un sistema operativo real-time (*RTOS*) ideale per soluzioni embedded su piccoli microcontrollori e con una API simile a quelle proposte dal consorzio OSEK/VDX. Viene distribuito da Evidence s.r.l. in doppia licenza (commerciale e GPL). Il kernel è accompagnato da un ambiente di sviluppo, *RT-Druid*, che fornisce strumenti per la scrittura, la compilazione e l'analisi dell'applicazione. Il sistema operativo offre:

- quattro classi di conformità OSEK;
- scheduling *Fixed Priority* (FP) e *Earliest Deadline First* (EDF);
- multitasking di tipo preemptive e non-preemptive;

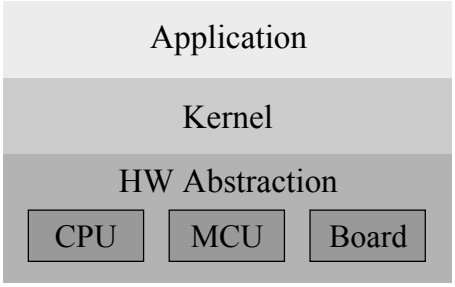

Figura 6.1: Architettura di *ERIKA RTOS*.

- gestione della condivisione dello stack e delle risorse condivise (*Resource*) mediante *Immediate Priority Ceiling*;
- semafori generalizzati (*Semaphore*);
- attivazione periodica dei task mediante allarmi (*Alarm*).

L'architettura di Erika, mostrata in Figura 6.1, è costituita da due livelli: *Kernel Abstraction Layer* (KAL) e *Hardware Abstraction Layer* (HAL). Il livello del Kernel implementa la gestione dei task e le varie strategie di schedulazione adottate, ed esporta al livello applicativo una serie di RTOS API per *Tasks*, *Alarms*, *Resources* e *Semaphores*. Il livello HAL si occupa invece della parte del sistema operativo dipendente dall'hardware, tra cui la gestione delle interruzioni e delle commutazioni di contesto. Tra le architetture attualmente supportate si ricordano Microchip dsPIC, Microchip PIC32, Altera NIOS II, Motorola PPC e Hitachi H8, Texas MSP430.

Lo standard OSEK (*Offene Systeme und deren schnittstellen für die Elektronik im Kraftfahrzeug* - open system and corresponding interfaces for automotive electronics) è un progetto congiunto di alcune industrie automobilistiche per lo sviluppo di sistemi di controllo distribuiti nei veicoli. Il consorzio prende il nome di OSEK/VDX poiché si occupa della definizione di un insieme di API per sistemi operativi real-time (OSEK) e di un sistema per la gestione di rete (*Vehicle Distributed eXecutive*, VDX). Esempi di applicazioni tipiche dello standard riguardano sistemi di controllo con vincoli temporali stringenti ed orientati a produzione di massa; pertanto le caratteristiche principali risultano: *scalabilità* (supporto per molteplici architetture hardware), *portabilità* (interfaccia ISO/ANSI-C), *configurabilità* (estensione *OSEK Implementation Language* - OIL) e *allocazione statica* (elementi del kernel allocati a tempo di compilazione).

L'ambiente di sviluppo per Erika Enterprise, RT-Druid , è composto da un set di plugin per il framework Eclipse responsabili della generazione del codice e dell'analisi di schedulabilità. Il *code generator* include un compilatore dello standard OSEK per OIL e, in base alle specifiche definite nella configurazione del sistema, costruisce la struttura necessaria alla compilazio-

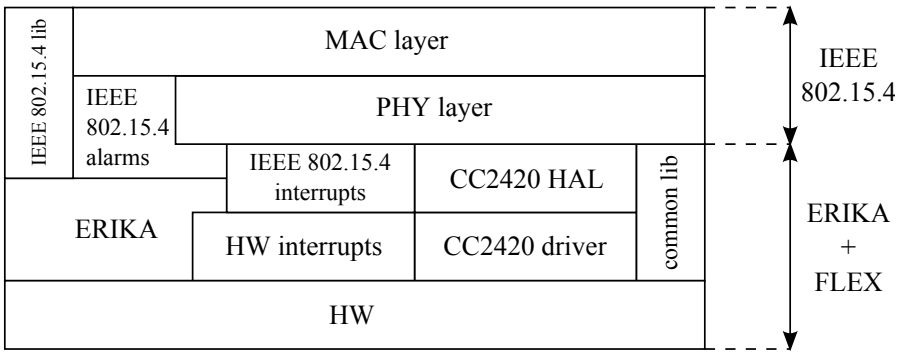

Figura 6.2: Architettura del sistema di comunicazione μWireless.

ne di ERIKA (makefiles, inizializzazione delle strutture dati, ecc). Il plugin per l'analisi della schedulabilità, disponibile solo nelle versioni commerciali del sistema operativo, permette di modellizzare, analizzare e simulare il comportamento temporale del sistema embedded.

La libreria *μWireless* è un'implementazione parziale del MAC layer di 802.15.4 per Erika Enterprise sviluppato dal laboratorio Retis della Scuola Superiore Sant'Anna di Pisa. Non vi è implementato un meccanismo per l'associazione dinamica dei dispositivi, che tuttavia include il supporto della modalità *beacon-enabled* con GTS. Come descritto in Figura 6.2, μWireless fa uso delle primitive software messe a disposizione da Erika. Le funzionalità dello stack sono infatti implementate all'interno di appositi task che vengono attivati per mezzo di allarmi periodici. Tra le numerose funzionalità esportate al livello applicativo da μWireless si riportano le seguenti:

- possibilità di configurare la durata del superframe e dei timeslot tramite i parametri *macBeaconOrder* (BO) e *macSuperframeOrder* (SO);
- trasmissione in modalità CSMA/CA o GTS;
- possibilità di sincronizzare l'applicazione con gli eventi della rete (ricezione di un frame, inizio e fine del superframe o del singolo time slot, ecc.) tramite la tecnica delle *callback*;
- possibilità di impostare il payload del beacon frame.

6.1.2 Tools Microchip per il supporto alla programmazione

Microchip fornisce un ambiente di sviluppo (*MPLAB IDE*), i driver per le periferiche ed un compilatore, integrando diverse funzionalità: organizzazione di file e progetti, editor per i sorgenti, programmazione e debugging. *MPLAB C30* è un compilatore ANSI-C compliant che include alcune estensioni di linguaggio per i microcontrollori della famiglia dsPIC®DSC mediante il porting del compilatore GCC della Free Software Foundation e viene

distribuito insieme ad un assemblatore e ad un linker. Le peculiarità di questo tipo di compilatore riguardano sostanzialmente la specifica di attributi per le variabili e per le funzioni, la definizione di funzioni inline e l'associazione di variabili a specifici registri. Il compilatore è distribuito anche in una versione *student* liberamente scaricabile dal sito di Microchip.

6.1.3 Framework Qt

Qt è una libreria multipiattaforma per lo sviluppo di programmi con interfaccia grafica tramite l'uso di widget (congegni o elementi grafici). *Qt*, ampiamente utilizzato nell'ambiente desktop KDE, viene sviluppato dall'azienda *Qt Software* (meglio conosciuta come *Trolltech* o *Quasar Technologies*) di proprietà di *Nokia*.

Qt usa il linguaggio C++ standard con un estensivo uso del preprocessore C per arricchire il linguaggio, ma esistono interfacce per Java, Python, C, Perl e PHP. Gira sulle piattaforme principali ed integra funzioni per l'accesso ai database SQL, parsing di documenti XML ed API multipiattaforma per l'uso dei file. Le librerie Qt sono disponibili per queste piattaforme:

- Qt/X11: Qt per X Window System;
- Qt/Mac: Qt per Mac OS X di Apple;
- Qt/Windows: Qt per Microsoft Windows;
- Qt/Embedded: Qt per piattaforme embedded (palmari e simili).

Attualmente vengono offerte in quattro edizioni, disponibili per ciascuna delle piattaforme:

- Qt Console: Versione embedded per lo sviluppo senza interfaccia grafica;
- Qt Desktop Light: Versione base per applicazioni con interfaccia grafica, senza supporto rete e SQL;
- Qt Desktop: La versione completa;
- Qt Open Source Edition : La versione completa, per sviluppo di software libero.

Le prime tre versioni sono rilasciate sotto con una licenza commerciale; la versione libera, invece, viene rilasciata sotto licenza GPL. A partire da Qt 4 inoltre, a differenza di Qt 3.3, Qt viene rilasciato sotto doppia licenza (GPL e commerciale) anche su Windows. L'edizione commerciale per Windows supporta Visual Studio; tutte supportano comunque il compilatore C++ GCC. La squadra di KDE ha anche rilasciato un'edizione di Qt, chiamata Qt/Windows Free Edition basata sul codice sorgente della versione Qt/X11

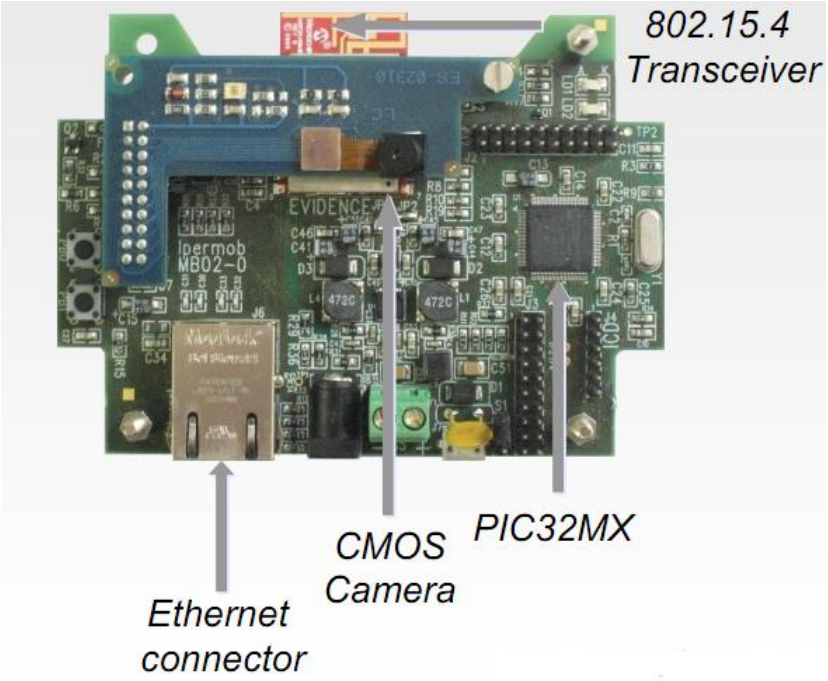

Figura 6.3: Seed-eye board.

e licenziata sotto GPL, viene rilasciata per il progetto KDE via Cygwin. Questo era stato ispirato dal fatto che le versioni di Qt precedenti alla 4.0 non erano disponibili come software libero sotto Windows.

6.2 Descrizione dell'Hardware

Di seguito sono elencati i vari dispositivi hardware utilizzati e le rispettive caratteristiche tecniche.

6.2.1 Seed-eye board

La *Seed-eye board* (Figura 6.3) è un dispositivo sviluppato sviluppato dal laboratorio Retis della Scuola Superiore Sant'Anna di Pisa e distribuito da da Evidence s.r.l. . Si tratta di una board embedded basata su microcontrollore PIC32 e interfaccia radio IEEE 802.15.4. Di seguito vengono riportate le caratteristiche tecniche:

- piccole dimensioni, peso ridotto;
- CPU: Microchip PIC32, 80 MHz;
- interfaccia Ethernet 100 Mb/s;

- interfaccia USB;
- CMOS Camera (fino a 30 fps, 640x480 pixels)
- transceiver IEEE 802.15.4;
- Connettori di espansione per unità di coprocessing esterna (CPLD + RAM).

6.2.2 Tag RFID basato su chip IDS sl900a

Il tag RFID usato per l'integrazione con la board *Seed-eye* è basato sul chip *IDS sl900a* (figura 6.4) Il tag funzione alle frequenze 860-960 MHz ed è conforme a tutti gli standard RFID di riferimento, può funzionare in modalità *passiva* senza alimentazione ma è anche in grado di funzionare in modalità *semi-passiva* con alimentazione a batteria da 1.5V oppure 3V nel caso via sia la necessità di rilevare ed eseguire il logging di parametri ambientali attraverso il sensore di temperatura oppure due sensori esterni. Di seguito vengono riportate le caratteristiche tecniche:

- Alto Range di temperatura: -40°C to +125°C;
- Frequenza: 860 - 960 MHz;
- Alimentazione a batteria: 1.5V or 3V;
- Data logging mediante:
  - Sensore di temperatura on-chip;
  - 2 sensori esterni;
- Conforme agli standard EPC Class1 e Class3;
- Compatibile con lo standard EPC Gen.2;
- Real-time clock per il data logging;
- Possibilità di ricevere interrupt dai sensori esterni;
- Serial Peripheral Interface (in modalità slave);
- Memoria EEPROM da 9k bit on-chip;
- Dynamic shelf life calculation integrato;
- Sistema avanzato di logging con 4 limiti settabili;

In figura 6.5 è mostrato lo schema di funzionamento del chip IDS sl900a.

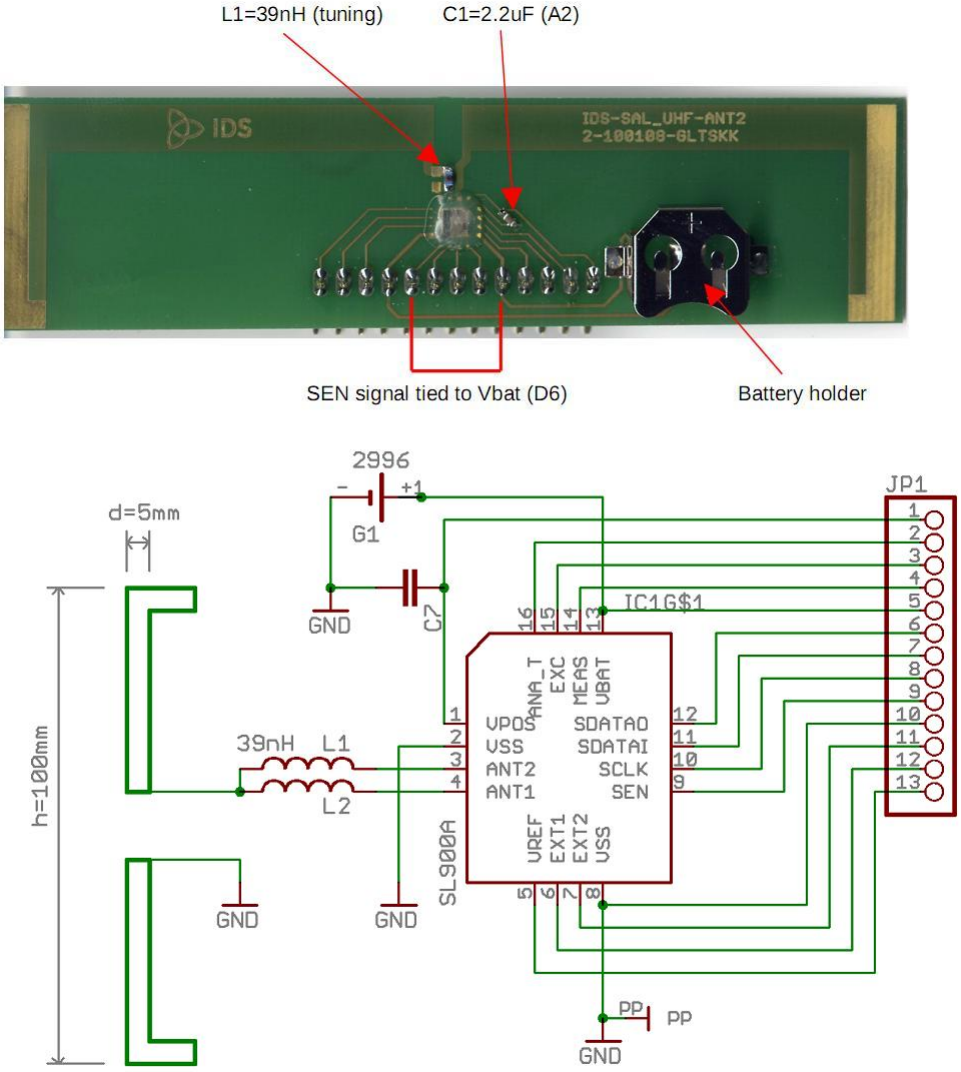

Figura 6.4: Tag RFID basato su chip IDS sl900a.

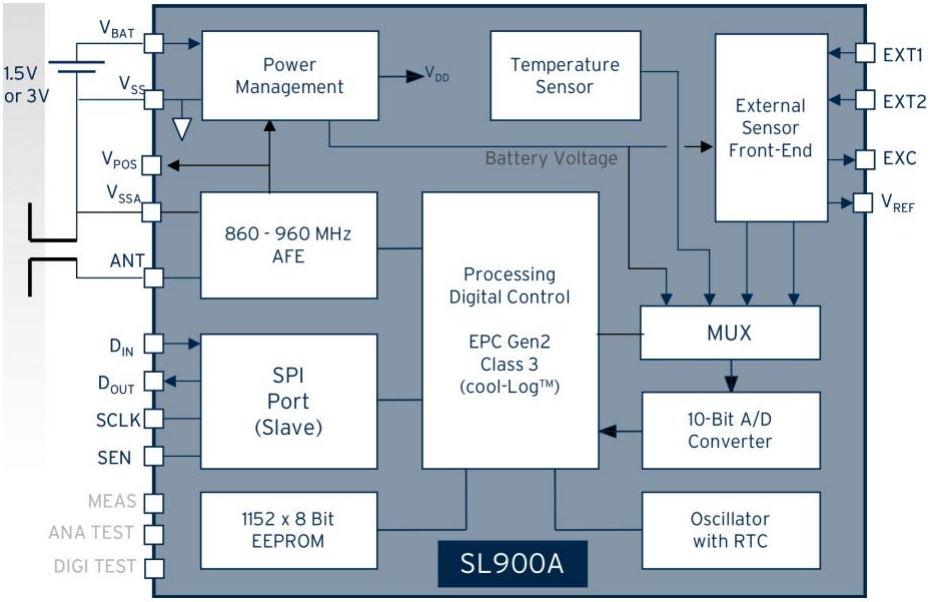

Figura 6.5: Schema del chip IDS sl900a.

Memoria La parte di memoria del dispositivo IDS sl900a alla quale si può accedere sia da interfaccia a radio frequenza che da SPI è organizzata nei seguenti blocchi:

- *TID memory (10 byte)*: è una memoria read only che contiene l'identificativo del tag che rispetta lo standard ISO 15693;
- *EPC memory (12 byte)*: è la memoria nella quale è contenuto l'Elctronic Product Code, il cui valore viene restituito al momento della ricerca dei tag da parte del reader;
- *RESERVED memory (8 byte)*: è la memoria che contiene le password kill e access (4 byte per ognuna);
- *USER memory (1052 byte)*: è la memoria utente che può essere usata per qualsiasi applicazione.

6.2.3 Tag RFID UHF passivi

I tags Passivi utilizzati sono il modello ALN-9654 G Inlay prodotto da Alien. Detti tags rispettano tutti gli standard di riferimento della tecnologia RFID UHF e garantiscono prestazioni ottime in termini di velocità di attivazione e distanze di lettura a fronte di un costo estremamente ridotto e in costante diminuzione grazie all'utilizzo sempre più massiccio di tag di questo tipo.

I tags passivi garantiscono la massima flessibilità di utilizzo: grazie alle piccole dimensioni soprattutto in termini di spessore (0.25mm massimo) si

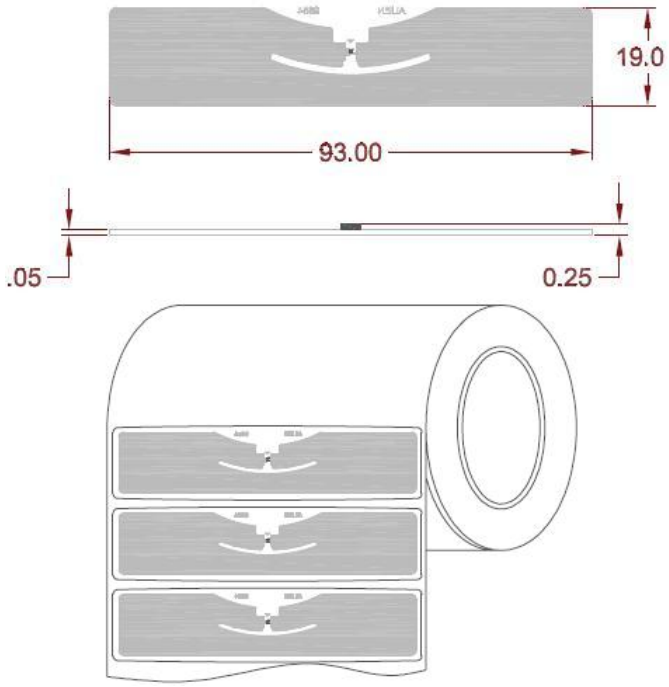

Figura 6.6: Tag passivi.

può pensare ad una agevole integrazione in qualsiasi oggetto da rilevare; possono essere inoltre forniti dal produttore in rotoli adesivi in modo da garantire un rapido utilizzo. In figura 6.6 sono riportate le dimensioni in mm dei tag passivi utilizzati.

Non essendo necessaria la batteria per il funzionamento, le temperature di utilizzo dei tags passivi sono estremamente flessibili (il funzionamento è garantito da -40°C fino a +70°C) inoltre grazie alle caratteristiche intrinseche di resistenza all'usura i tags passivi garantiscono una durata teoricamente infinita (non essendo legata alla vita della batteria) e basso deterioramento. Di seguito sono riportate le caratteristiche tecniche dei tag passivi utilizzati:

- Conforme allo standard EPC Global Class 1 Gen 2 (v1.2.0);
- Conforme allo standard ISO/IEC 18000-6C;
- Frequenze supportate: 840-960MHz;
- Memoria TID modificabile: 32 bit;
- Pre-programmato con un identificativo di 64 bit unico e non modificabile;
- Memoria EPC: 96 bit, esensibilile fino a 480 bit;

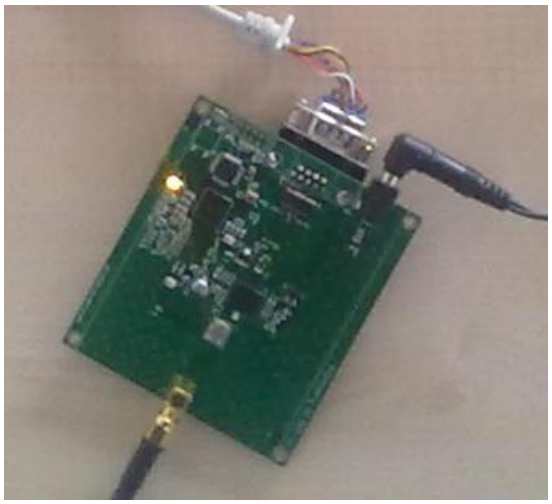

Figura 6.7: Reader RFID UHF.

- Memoria USER: 512 bit;
- Access Password: 32 bit;
- Kill password: 32 bit;
- La USER Memory può essere bloccata in lettura e scrittura in modo definitivo;
- Temperatura operativa da -40°C a +70°C.

6.2.4 Reader RFID UHF

Il reader RFID utilizzato è il modulo *Sensor ID Discovery UHF OEM* (figura 6.7)nella versione con potenza di uscita 500mW (27dBm). La frequenza di trasmissione del segnale a radio frequenza è settabile da software nel range 865-920 MHz (le frequenze utilizzate per l'europa sono 865.7-867.5 MHz) ed è in grado di rilevare tag passivi fino alla distanza di 5 metri.

Di seguito sono riportate le caratteristiche tecniche:

- Frequenza settabile via software (Europe 865.7 - 867.5);
- Alimentazione: 5V - 300mA inreading mode, 70mA in idle mode e 10mA in stand-by mode;
- Distanza di lettura: fino a 5 m (dipende dall'antenna);
- Potenza RF: 500mW (27dBm) (Attenuazione configurabile via software);
- GPIO: 2;

Figura 6.8: Antenna RFID UHF.

- Standard Supportati:
  - ISO 18000 6 A/B/C;
  - EPC Class 1 Gen 2;
  - Philips UCODE;
- Antenna esterna alimentata da lettore

6.2.5 Antenna RFID UHF

L'antenna utilizzata (Figura 6.8) è pensata per ottenere buone performance in termini di distanza di lettura ed è inoltre adatta a qualsiasi tipo di installazione sia in ambienti chiusi che all'aperto. Di seguito sono riportate le caratteristiche tecniche:

Caratteristiche elettriche:

- Frequenza operativa: 860-960 MHz;
- Gain (max): 7 dBi (+-1 dB);
- Gain (min over the band): 6.5 dBi (+-0.5dB);
- Axial ratio: <1dB;
- Feed power handling: 10W;
- Horizontal 3dB beamwidt: 67° (+− 5°);

- Vertical 3 dB beamwidth: 69° (+−5°);
- Front to back (F/B ratio): 20 dB (+−3 dB);
- Nominal input impedance: 50 Ohm;
- Polarizzazione: circolare.

Caratteristiche ambientali:

- Range di temperatura: da -20°C a +70°C;
- Resistenza all'acqua: IP66;
- Umidità: >95% (Non condensing) MIL STD810E;

Caratteristiche meccaniche:

- Dimensioni: 245 mm x 235 mm x 40 mm;
- Peso: 470g;

6.3 Integrazione dell'hardware

6.3.1 Nodo WSN integrato con reader RFID UHF

Il reader RFID è stato collegato al nodo WSN mediante interfaccia seriale RS232 con i seguenti settaggi:

- BaudRate = 15200;
- DataBits = 8;
- Parity: NO;
- FlowControl: NO;

La porta utilizzata è mappata all'interno dei driver per il sistema operativo Erika Enterprise come EE_UART_PORT_1A.

6.3.2 Nodo WSN integrato con tag RFID UHF

Comunicazione Serial Peripheral Interface Il tag RFID UHF basato su chip IDS sl900a è stato integrato mediante interfaccia SPI (*Serial Peripheral Interface*) con il mocrocontrollore PIC32 della board Seed-eye secondo lo schema mostrato in figura 6.10.

Come si può vedere i pin utilizzati per il collegamento SPI sono quelli che vanno dal 9 al 12 e cioè:

- SCLK: serial clock signal, generato dal microcontrollore PIC32 della board Seed-eye;

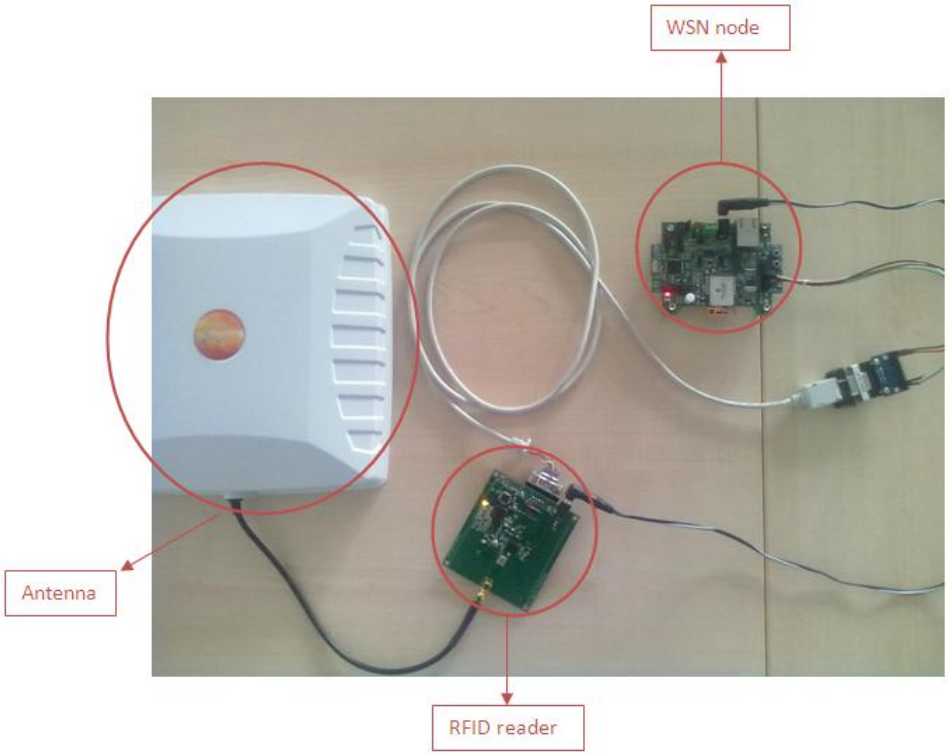

Figura 6.9: Nodo WSN integrato con reader RFID UHF.

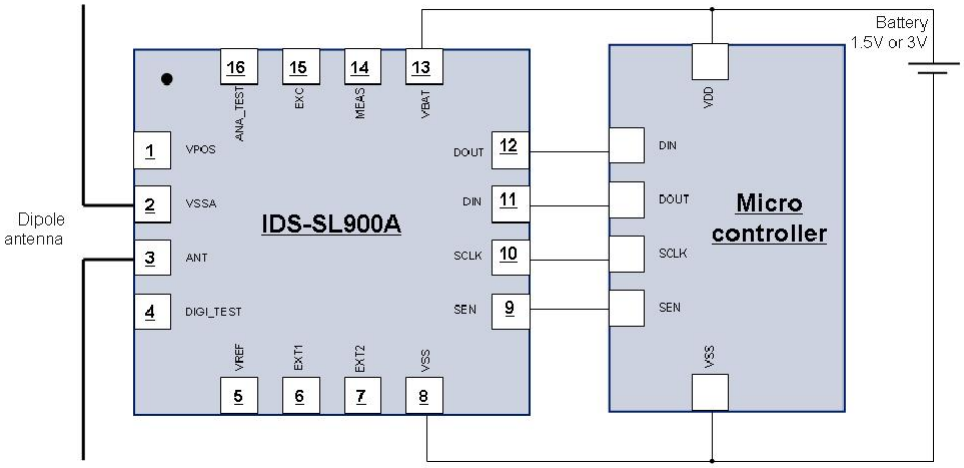

Figura 6.10: Integrazione del chip sl900a con il micro PIC32.

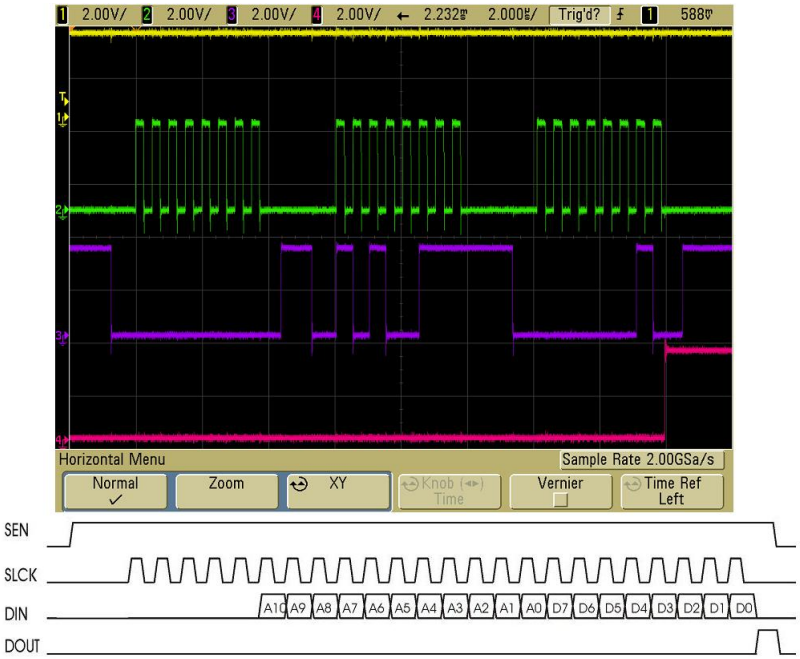

Figura 6.11: Livelli di tensione in caso di scrittura sul chip sl900a.

- SEN: slave enable signal, generato dal microcontrollore;
- DIN: data input signal, generato dal microcontrollore;
- DOUT: data output signal, generato dal chip sl900a.

La massima frequenza del clock è 5MHz se alimentato a 3V e 1MHz se alimentato a 1V. La comunicazione è attiva quando il segnale SEN è alto. I dati sono resi disponibili sul fronte di salita del segnale SCLK e campionati sul fronte di discesa del segnale SCLK.

In figura 6.11 sono mostrati i livelli di tensione dei vari pin per un'operazione di scrittura sul chip.

6.3.3 Gestione dell'interrupt mediante segnale a radiofrequenza

Come evidenziato nel capitolo 3 uno dei requisiti del sistema è quello di avere una gestione intelligente del consumo energetico delle batterie dei nodi. A questo proposito è stato implementato un meccanismo che permette ad ogni nodo in dotazione agli operai di entrare in uno *stand-by mode* quando non si trova all'interno dell'area a rischio e di risvegliarsi nel momento in cui l'operaio accede mediante un interrupt.

Il tag RFID infatti funziona in modalità passiva, quindi non è alimentato ma riceve l'energia necessaria dal reader e comunica secondo il principio

dell'effetto backscatter decritto nel Capitolo 3 per il quale viene modulata l'impedenza del circuito dell'antenna in modo da riflettere o assorbire parte del segnale ricevuto. In questo tipo di funzionamento la potenza disponibile sul circuito del tag è espressa dalla formula di Friis:

$$P_{tag} = G_{tag}\tau\rho_t\frac{\lambda_{RF}^2}{8\pi\eta}|\vec{E}_{inc}|^2 = P_{eirp}G_{tag}\frac{\lambda_{RF}^2}{8\pi d}\tau\rho_t$$

dove:

- P_{eirp} è la potenza irradiata dal reader;
- G_{tag} è il guadagno dell'antenna del tag;
- d è la distanza tra reader e tag;
- λ_{RF} è la lunghezza d'onda operativa;
- $\tau \in [0:1]$ è il fattore di polarizzazione che tiene conto del disadattamento di polarizzazione tra l'antenna ricevente e il campo incidente;
- $|\vec{E}_{inc}|$ è il campo elettrico incidente sull'antenna del tag;
- η è l'impedenza caratteristica del mezzo di propagazione;
- ρ_t è il coefficiente di assorbimento di potenza del tag espresso da:

$$\rho_t = \frac{4R_{A_tag}R_l}{|Z_{A_tag} + Z_l|^2} \in [0:1]$$

il quale risulta dipendente dall'impedenza dell'antenna del tag, $Z_{A_tag} = R_{A_tag} + jX_{A_tag}$ e dall'impedenza di carico $Z_l = R_l + jX_l$ dello stesso tag.Quest'ultima può variare tra due valori Z_{l1} e Z_{l2} per modulare il backscattering.

Dalla formula di Friis è possibile ricavare la distanza operativa di un tag passivo elettromagnetico che è data da:

$$d = \frac{\lambda_{RF}}{4\pi}\sqrt{\frac{P_{eirp}G_{tag}\tau\rho_t}{P_{int_min}}}$$

Essendo il limite per le emissioni previsto dalla normativa italiana [31] pari a 2W E.I.R.P., la lunghezza d'onda alla frequenza di 868 MHz pari a 0,3456 m, il modulo dell'antenna del tag pari a circa 2,15 dB e la corrente necessaria per attivare l'interrupt pari a 50 μA si ha che la distanza massima ottenibile per L'Alimentazione del tag in condizioni ideali è pari a circa 7 metri contro i 9 metri per la normale lettura del tag.

Per l'implementazione reale del sistema si è utilizzato pin Vref del tag sl900a è disponibile la potenza calcolata quando il tag si trova nel raggio

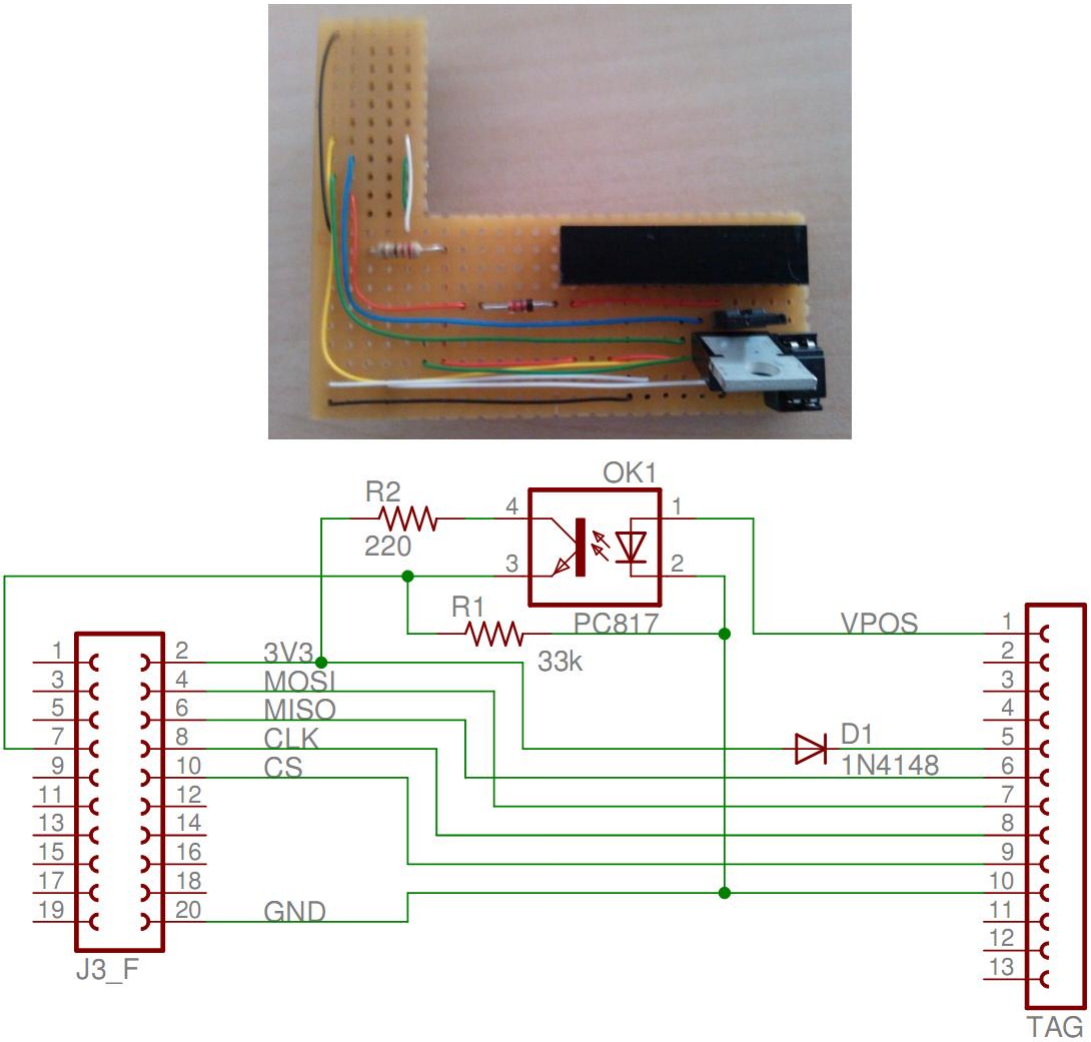

Figura 6.12: Board di connessione.

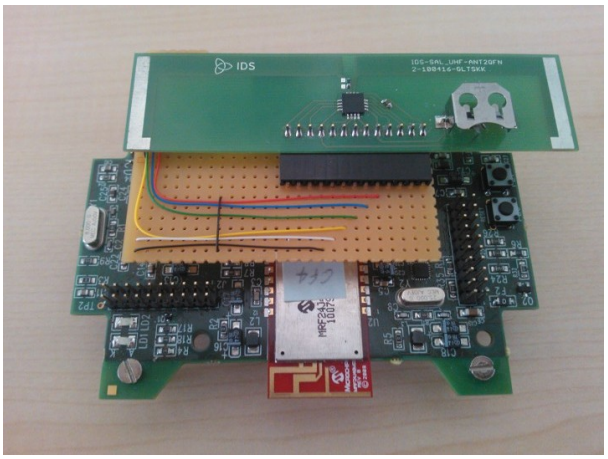

Figura 6.13: Nodo WSN integrato con tag RFID UHF.

d'azione dell'antenna del reader e quindi tale pin è stato collegato al pin CN5 del microcontrollore PIC32 che corrisponde al pin RB3 della porta B.

In Figura 6.12 è riportata la board di connessione tra il tag IDS sl900a e la board Seed-eye, mentre in figura 6.13 è riportato il dispositivo "nodo tag" risultante dall'integrazione.

La gestione dell'interrupt per il sistema operativo ERIKA Enterprise per il risveglio del microcontrollore PIC32 dallo *stand-by mode* è stata implementata nei file **sleep.h** e **sleep.c** (Appendice A) facendo uso delle funzioni che gestiscono il *change notify* all'interno del sistema operativo.

6.4 Implementazione dei driver per ERIKA Enterprise

Al fine di garantire un utilizzo rapido e intuitivo del reader *Discovery UHF* e del tag *IDS sl900a* all'interno del sistema operativo *ERIKA Enterprise RTOS* sono state implementate alcune funzioni nelle librerie **reader_uhf.h** e **ids_sl900a.h** in modo che il microcontrollore possa gestire il reader come una semplice periferica.

Sono inoltre stati creati i file **cfg.mk** e **libcfg.mk** che permettono il linkaggio e l'utilizzo semplice della libreria mediante il file **conf.oil** previsto dal sistema operativo.

Driver per il reader RFID UHF Le funzioni implementate all'interno della libreria **reader_uhf.h** permettono la comunicazione tra la board sulla quale è installato il sistema operativo (nello specifico la *Seed-eye*) e tutte le operazioni necessarie sulla memoria dei tag rilevati dal reader. In particolare tali funzioni permettono di:

- Inizializzare il reader e settare la potenza del segnale in uscita;
- Ottimizzare la frequenza di uscita per migliorare il rilevamento del tag IDS sl900a;
- Ricercare i tag all'interno del range di lettura dell'antenna;
- Scrivere e leggere allinterno dei vari campi di memoria dei tag come da standard EPC global class1 gen2;
- Settare password di accesso per proteggere i vari campi di memoria.

In Tabella 6.1 è riportata una descrizione delle funzioni più importanti:

Funzione;	Descrizione
rfid_init	Setta la comunicazione seriale
set_max_output_power	Setta la potenza in uscita e la sensitività in ricezione
set_freq_ids	Setta la frequenza di ricezione per ottimizzare le prestazioni del tag
inventory	Rileva i tag e slva i loro EPC
select_tag	seleziona il tag tra quelli trovati
read_tag	legge nei vari banchi di memoria del tag
read_tag	scrive nei vari banchi di memoria del tag
lock_memory	blocca con password i banchi di memoria
unlock_memory	sblocca con password i banchi di memoria

Tabella 6.1: Funzioni della libreria rfid.h

La descrizione completa e l'implementazione delle funzioni è riportata in Appendice A.

Driver per il tag IDS sl900a Le funzioni implementate all'interno della libreria **ids_sl900a.h** permettono la comunicazione tra la board sulla quale è installato il sistema operativo (nello specifico la *Seed-eye*) e di gestire le funzionalità del chip *IDS sl900a*. In particolare tali funzioni permettono di:

- Inizializzare il tag e settare la velocità di trasmissione dei dati;
- Leggere i registri del chip IDS sl900a;
- Scrivere nei registri del chip IDS sl900a;
- Acquisire il dato del sensore di temperatura integrato;
- Acquisire i dati di eventuali sensori aggiuntivi;

- Acquisire il dato del voltaggio della batteria (nel caso di tag alimentato indipendentemente);
- Acquisire il tempo di vita residuo della batteria (nel caso di tag alimentato indipendentemente);

In Tabella 6.2 è riportata una breve descrizione delle funzioni fondamentali della libreria: La descrizione completa e l'implementazione delle funzioni

Funzione;	Descrizione
tag_init	Inizializza la comunicazione SPI
write_register	Scrive in uno specifico registro
read_register	Legge in uno specifico registro
get_temperature	Restituisce la temperatura dal sensore integrato
get_battery	Restituisce la tensione in V della batteria
get_sensor1	Restituisce il valore del sensore 1
get_sensor2	Restituisce il valore del sensore 2

Tabella 6.2: Funzioni della libreria sl900a.h

è riporata in Appendice A.

Capitolo 7

Progettazione del sistema

In questo capitolo vengono analizzati i meccanismi di *data management* e i protocolli di comunicazione implementati per il funzionamento dell'applicazione e lo scambio delle informazioni tra i diversi dispositivi del sistema precedentemente descritti. Viene inoltre descritto il meccanismo di autenticazione che abilita o meno l'ingresso degli operai all'interno delle varie aree dello stabilimento.

7.1 Data Management

Per quanto riguarda la gestione dei dati e della memoria si è scelto di tenere la quantità maggiore possibile di informazione nella memoria del *nodo tag* in dotazione all'operaio, questo nell'ottica di creare come sviluppo futuro una rete *ad-hoc* tra gli oggetti qualora la tecnologia lo rendesse tecnicamente ed economicamente possibile. Creando una rete di questo tipo, infatti, il dispositivo in dotazione agli operai potrebbe essere in grado di rilevare in ogni momento la presenza o meno dei dispositivi individuali di protezione andando cosi a segnalare se un dispositivo è stato dimenticato ecc.

A questo scopo si è scelto di inserire l'informazione relativa agli oggetti in dotazione e al loro utilizzo oltre che il codice identificativo dell'operaio all'interno della memoria del "nodo Tag". In particolare i dati vengono salvati all'interno dei vari banchi di memoria del tag *IDS sl900a* integrato con la board *Seed-eye* essendo in questo modo accessibili sia da interfaccia a radiofrequenza che dal microcontrollore del nodo e quindi resi disponibile alla rete WSN per eventuali aggiornamenti o funzionalità aggiuntive. Nello specifico sono stati utilizzati il banco EPC e il banco USER come di seguito descritto. In figura 7.1 è riportato uno schema dei registri della memoria utilizzata.

Loc. #	Physical address	Bank	Bank name	Logical address	Content	Group
65	0x040	0	RESERVED	0x00	Kill Password [31:24]	Kill Password
66	0x041				Kill Password [23:16]	
67	0x042			0x01	Kill Password [15:8]	
68	0x043				Kill Password [7:0]	
69	0x044			0x02	Access Password [31:24]	Access Password
70	0x045				Access Password [23:16]	
71	0x046			0x03	Access Password [15:8]	
72	0x047				Access Password [7:0]	
RAM - 1	RAM - 0x00	1	EPC	0x00	CRC-16 [15:8]	CRC-16 is stored in the RAM portion and is mapped to the EPC memory block
RAM - 2	RAM - 0x01				CRC-16 [7:0]	
73	0x048			0x01	PC [15:8]	PC
74	0x049				PC [7:0]	
75	0x04A			0x02	EPC [127:120]	EPC
76	0x04B				EPC [119:112]	
77	0x04C			0x03	EPC [111:104]	
78	0x04D				EPC [103:96]	
79	0x04E			0x04	EPC [95:88]	
80	0x04F				EPC [87:80]	
81	0x050			0x05	EPC [79:72]	
82	0x051				EPC [71:64]	
83	0x052			0x06	EPC [63:56]	
84	0x053				EPC [55:48]	
85	0x054			0x07	EPC [47:40]	
86	0x055				EPC [39:32]	
87	0x056			0x08	EPC [31:24]	
88	0x057				EPC [23:16]	
89	0x058			0x09	EPC [15:8]	
90	0x059				EPC [7:0]	
91	0x05A	2	TID	0x00	TID [7:0] – 0xE0	TID (same format as UID in ISO 15693), READ ONLY
92	0x05B				TID [15:8] – 0x36	
93	0x05C			0x01	TID [23:16]	
94	0x05D				TID [31:24]	
95	0x05E			0x02	TID [39:32]	
96	0x05F				TID [47:40]	
97	0x060			0x03	TID [55:48]	
98	0x061				TID [63:56]	
99	0x062			0x04	Chip version [7:0]	Version, etc... READ ONLY
100	0x063				RFU [7:0]	
101	0x064	3	USER	0x000	USER memory start - UMI	USER / MEASUREMENT memory – 1052 bytes
102	0x065					
–	–			–		
–	–			–		
1151	0x47E			0x211		
1152	0x47F				USER memory end	

Figura 7.1: Indirizzi della memoria del tag IDS sl900a.

7.1.1 Organizzazione della memoria EPC

LA memoria EPC come detto è composta da 12 byte ed è organizzata come mostrato in tabella 7.1.

Byte 0	Byte 1 - Byte 2	Byte 3	Byte 4...Byte 12
ID del tipo di tag	ID della persona	ID oggetto	RFU

Tabella 7.1: Memoria EPC

IL byte 0 viene utilizzato per discriminare se il tag sia associato ad una persona oppure ad un oggetto e quindi sia il tag *sl900a* oppure un normale tag passivo associato ad un dispositivo individuale di protezione. Nei byte 1 e 2 è contenuto il codice identificativo dell'operaio e nel byte 3 è contenuto il codice identificativo dell'oggetto, mentre i byte da 4 a 12 sono lasciati per eventuali altri utilizzi.

Quindi nel caso il tag identifichi un operaio si avrà la configurazione di tabella 7.2:

Byte 0	Byte 1 - Byte 2	Byte 3	Byte 4...Byte 12
0x01	ID operaio	0xFF	RFU

Tabella 7.2: Memoria EPC per tag operaio

Nel caso il tag identifichi un dispositivo individuale di protezione si avrà invece la configurazione di tabella 7.3:

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4...Byte 12
0x02	0xFF	0xFF	ID oggetto	RFU

Tabella 7.3: Memoria EPC per tag oggetto

7.1.2 Organizzazione della memoria USER

Nella memoria user del tag *IDS sl900a* è contenuta l'informazione relativa ai dispositivi individuali di protezione. In particolare nella prima parte di memoria (indirizzi da 0x064 a 0x080) sono contenuti gli identificativi dei diversi oggetti in dotazione all'operaio (max 16 oggetti). Ogni oggetto è identificato con 1 byte (che corrisponde al byte 3 della memoria EPC del tag associato all'oggetto). In totale quindi l'area di memoria occupata è 16 byte. In tabella 7.4 è mostrato un esempio di memorizzazione degli identificativi nei vari registri.

ID Oggetto	Registro
Caschetto	0x064
Mascherina	0x065
Occhiali	0x066
...	0x067

Tabella 7.4: Esempio di memorizzazione dei DPI

Nella seconda parte della memoria User (indirizzi da 0x081 a 0x112)è contenuta una maschera di 16 bit che permette di associare ad ogni area dello stabilimento gli oggetti in dotazione che il singolo operaio deve avere con sé per essere autorizzato all'ingresso. Il bit più significativo della maschera corrisponde all'oggetto salvato nel primo indirizzo della prima parte della memoria USER cioè 0x064 e così via. In tabella 7.5 è mostrato un esempio di memorizzazione delle maschere.

Area	Registro	Maschera
Produzione	0x081	0001000000100010
Magazzino	0x082	1000001000100010
Locale caldaia	0x083	0100001000010000
...	...	...

Tabella 7.5: Esempio di memorizzazione delle maschere

Sono previste 8 aree soggette a controllo e quindi la memoria occupata è pari a 16 byte. Complessivamente la memoria USER viene quindi occupata per 32 byte.

Al fine di prevedere un'inizializzazione semplice e modulare della memoria sono state esportate alcune semplici funzioni per il sistema operativo *ERIKA Enterprise* all'interno del file **set_memory.h**. In questo modo è possibile, implementando tecniche di riprogrammazione in remoto, effettuare in qualsiasi momento aggiornamenti sia delle aree da controllare che dei dispositivi di protezione necessari per ogni area Tali funzioni sfruttano i driver di comunicazione SPI tra la board *Seed-eye* e il tag *IDS sl900a* precedentemente descritti in modo da permettere la modifica dei dati anche dalla rete WSN. In Tabella 7.6 è riportata la descrizione delle funzioni implementate:

La descrizione completa e l'implementazione delle funzioni è riportata in Appendice B.

Funzione;	Descrizione
insert_obj_id	Inserisce il codice identificativo di un oggetto nella memoria del tag
add_obj_req	Inserisce il requisito di un oggetto per una determinata erea dello stabilimento
del_obj_req	Cancella il requisito di un oggetto per una determinata erea dello stabilimento

Tabella 7.6: Funzioni di inizializzazione della memoria

7.2 Meccanismo di autenticazione

Come descritto precedentemente la memoria utilizzata per la gestione dei dati relativa ai dispositivi individuali di protezione è quella del "nodo tag" mentre i dati relativi ai codici identificativi degli operai e alle loro autorizzazioni sono contenuti in un database centrale. Per quanto riguarda il meccanismo di autenticazione quindi, il sistema deve essere in grado di gestire i dati dei vari tag rilevati comunicando con il "nodo tag" e con il database centrale.

In figura 7.2 è riportato lo schema logico del meccanismo di autenticazione.

La logica di autenticazione è implementata sul dispositivo "nodo reader" attraverso la funzione **autenticate** che gestisce le letture necessarie sui tag come descritto nello schema e ritorna una maschera di 16 bit che, in caso di autorizzazione non concessa, consente di conoscere i dispositivi di protezione mancanti. L'implementazione della logica di autenticazione è riportata in Appendice C.

7.3 Protocollo di comunicazione della rete WSN e localizzazione

La rete WSN è configurata per funzionare secondo l'architettura 6LoWPAN quindi con indirizzamento dei device secondo lo standard IPv6. Il *nodo reader* è il coordinator della rete, quindi una volta eseguito il meccanismo di autenticazione invia un messaggio multicast che attiva la rete WSN.

La struttura dei messaggi scambiati tra i vari dispositivi della rete è riportata in tabella 7.7

Nello specifico, una volta autenticato l'operaio, viene attivato un servizio di localizzazione basato sul rilevamento del valore RSSI del segnale emesso dal *nodo mobile*. Tale valore viene rilevato di volta in volta dai vari nodi della rete (ancore) e quindi l'operaio viene localizzato con esattezza.

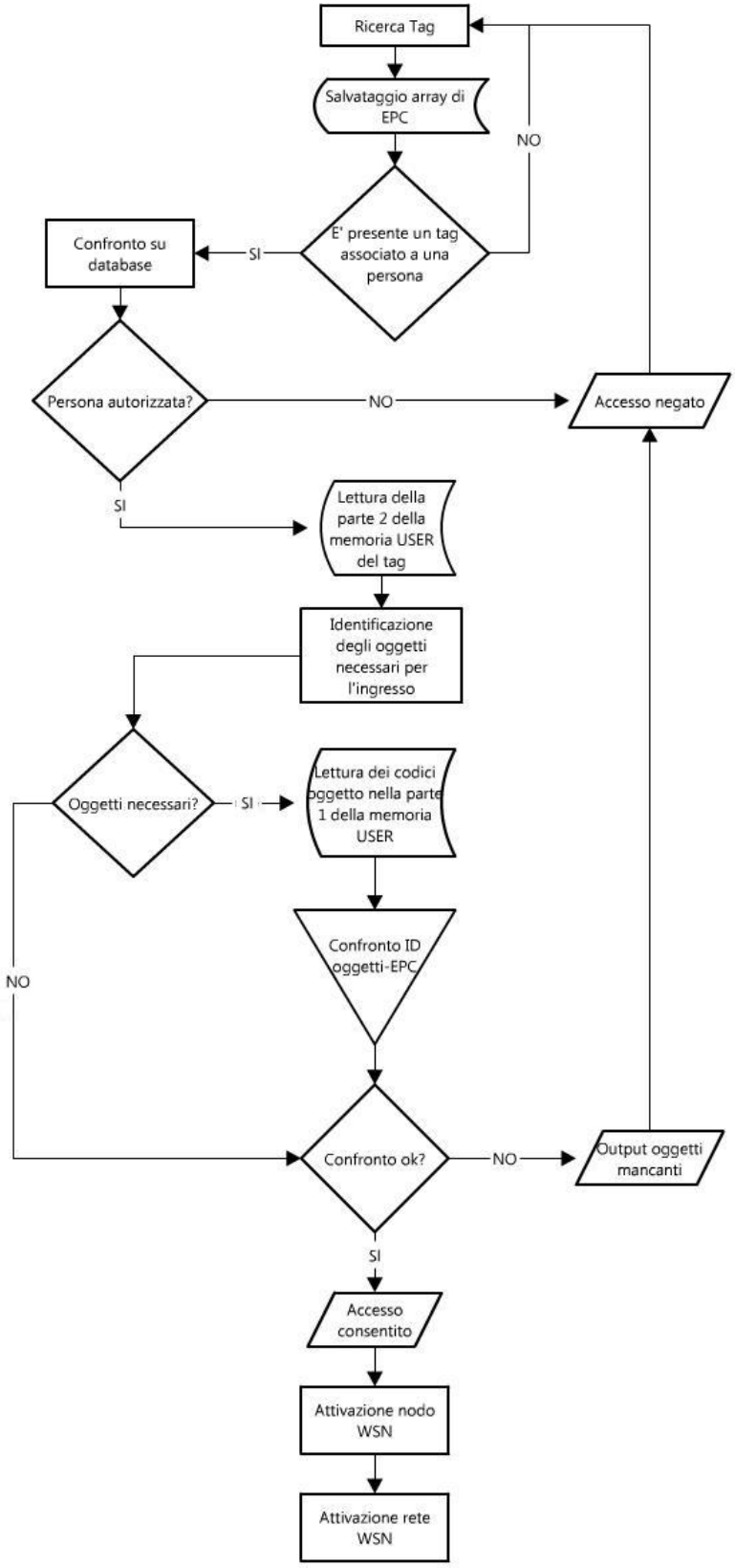

Figura 7.2: Schema del meccanismo di autenticazione.

Byte 0	Byte 1	Byte 2	Byte 3 - Byte 19	Byte 20 -
Tipo di messaggio	Azione	Servizio	ID nodo (IPv6)	Parametri

Tabella 7.7: Struttura dei messaggi scambiati

Dopo la procedura di autenticazione quindi il nodo coordinatore invia il messaggio multicast di attivazione del servizio di localizzazione. Il *nodo mobile* che nel frattempo si è stato attivato dal reader RFID secondo lo schema descritto nella sezione 6.3.2 riceve il messaggio e comincia ad inviare i pacchetti che permettono di localizzarlo; le ancore ricevono i pacchetti e una volta calcolata la posizione il coordinatore stampa su interfaccia grafica le coordinate della posizione dell'operaio all'interno dell'area. Il codici del coordinator e del nodo mobile che gestiscono il meccanismo descritto sono riportati in appendice E.

7.4 Database e GUI su PC

Come detto precedentemente i dati relativi ai codici identificativi degli operai e alle autorizzazioni previste per l'accesso a ciascuna area da parte degli stessi sono contenuti in un database centrale.

Il datatbase è stato implementato su pc ed è stato creato, a titolo esemplificativo, sulla base di una serie di file di testo ciascuno dei quali si riferisce ad una determinata area dello stabilimento. E' stata inoltre creata una GUI (graphical user interface) basata su framework qt che permette di avere disponibile sullo schermo del pc un semaforo di autenticazione per l'accesso alle varie aree dello stabilimento. Di seguito è riportato il funzionamento del database e della GUI.

Il nodo WSN collegato al pc tramite seriale RS232 trasmette l'informazione relativa all'dentità dell'operaio e ai dispositivi di protezione di cui è in possesso mediante un messaggio di 5 byte secondo lo schema di seguito riportato:

Byte 0	Byte 1 - Byte 2	byte 3 - byte 4	EOF
ID area	ID operaio	Maschera DPI	EOF

Tabella 7.8: Messaggio nodo-database

In base al valore del byte 0 l'applicazione decide quale file di testo aprire e va a verificare se all'interno dello stesso vi sia il codice identificativo dell'operaio e in caso affermativo ne legge nome e cognome. L'implementazione del codice relativo alla gestione delle aree e dei dati ricevuti è riportata in appendice D.

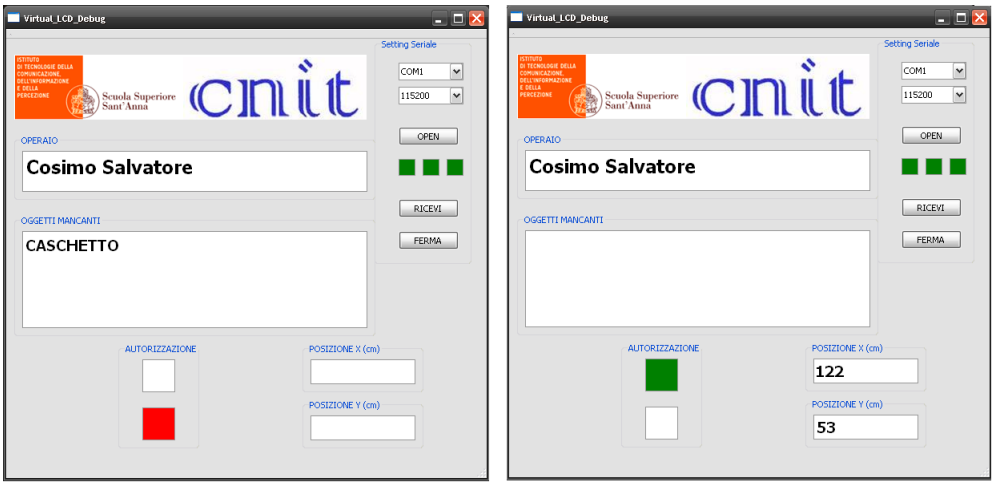

Figura 7.3: Interfaccia grafica su pc.

Tali dati vengono stampati nell'apposito riquadro della GUI su sfondo verde; in caso l'operaio non sia autorizzato viene stampata la scritta "NON AUTORIZZATO" su sfondo verde.

Quindi l'applicazione va ad esaminare la maschera dei DPI contenuta nei byte 3 e 4 del messaggio ricevuto su pc. Nel caso in cui l'operaio non abbia con se tutti gli oggetti vengono stampati quelli mancanti nell'apposito riquadro su sfondo rosso, in caso contrario lo sfondo è verde. Infine il terzo riquadro stabilisce l'ingresso o meno dell'operaio nell'area. Sull'interfaccia grafica viene stampata istante per istante la posizione dell'operaio all'interno dell'area ottenuta dai dati del servizio di localizzazione basato su RSSI. I dati sono raccolti dal coordinator che li passa all'interfaccia, la quale stampa negli appositi campi le coordinate della posizione.

In figura 7.3 è riportato un esempio di come si presenta l'interfaccia grafica.

Capitolo 8

Prestazioni del sistema

Al fine di analizzare sperimentalmente il corretto funzionamento e le prestazioni del sistema è stato implementato uno scenario reale di testing come mostrato in Figura 8.1. I test effettuati hanno riguardato, oltre alla verifica della conformità del sistema alle specifiche previste, l'analisi delle prestazioni del sistema per quanto riguarda:

- Affidabilità e tempi necessari per il processo di autenticazione.
- Riduzione del consumo energetico;

8.1 Affidabilità e tempi necessari per il processo di autenticazione

Il test è stato ripetuto 500 volte per avere un campione adeguato di dati con diverse configurazioni per quanto riguarda il numero di oggetti necessari.

Per garantire i necessari livelli di sicurezza il sistema è stato implementato in modo da evitare false autenticazioni, non deve essere permesso infatti ad un operaio non autorizzato o non in possesso dei dispositivi di protezioni necessari di accedere all'area in presenza di errori del sistema. Con il sistema implementato non si hanno quindi false autenticazioni, mentre autentication missing si hanno solo se i tag sono fuori dal range di lettura del reader.

Il tempo necessario per il processo di autenticazione è quell'intervallo che va dal momento in cui il *nodo mobile* viene rilevato dal *nodo reader* e quindi risvegliato dallo stato di sleep mode, all'istante in cui riceve il pacchetto di attivazione della rete WSN. In tale intervallo temporale il *nodo reader* legge più volte la memoria del tag "master" per verificare gli oggetti necessari come descritto nel capitolo 7 e legge le memorie dei tag attaccati agli oggetti per verificare il loro identificativo. Tale processo può comportare ovviamente errori di lettura dovuti a molteplici parametri quali la posizione dei tag e la potenza del reader. Nel caso ci siano problemi di questo genere il

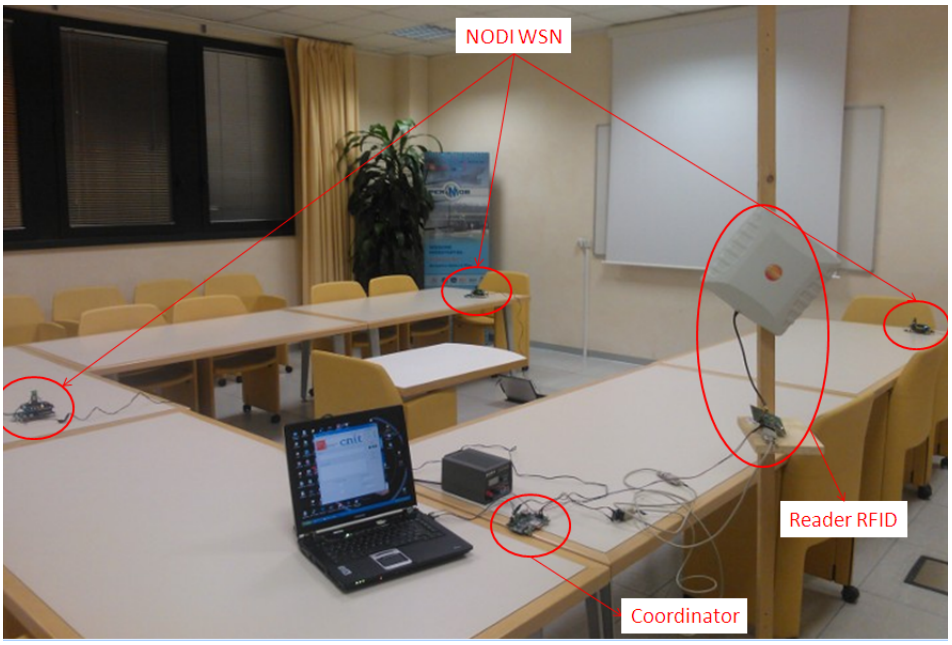

Figura 8.1: Scenario di testing.

sistema esegue l'operazione di lettura più volte con un conseguente aumento del tempo necessario per l'autenticazione.

Dai test effettuati è risultato comunque che tale inconveniente si manifesta in un numero limitato di casi, per i quali in ogni caso il tempo di autenticazione rimane più che accettabile non solo per l'applicazione proposta ma anche per eventuali altre applicazioni ad esempio veicolari. Si è rilevato inoltre che non si hanno significative variazioni dei tempi in funzione del numero di oggetti, essendo il processo di lettura degli stessi gestito in maniera molto rapida dal reader.

In figura 8.2 si può vedere come in un numero limitato di casi il sistema non riesca ad effettuare il processo di autenticazione al primo tentativo.

In figura 8.3 si può vedere come variano i tempi in funzione del numero di tentativi di autenticazione necessari e in funzione dell'errore

In figura 8.4 è riportata la distribuzione dei tempi nel caso il processo di autenticazione avvenga senza errori di lettura dei tag.

8.2 Riduzione del consumo energetico dei nodi

Come detto precedentemente una delle specifiche più importanti del sistema è l'ottimizzazione del consumo delle batterie dei nodi, quest'aspetto infatti risulta essere fondamentale per un'implementazione efficace di una rete WSN, in particolare è importante garantire che le batterie dei *nodi mobili* forniti agli operai abbiano un'autonomia che ne consenta il funzionamento

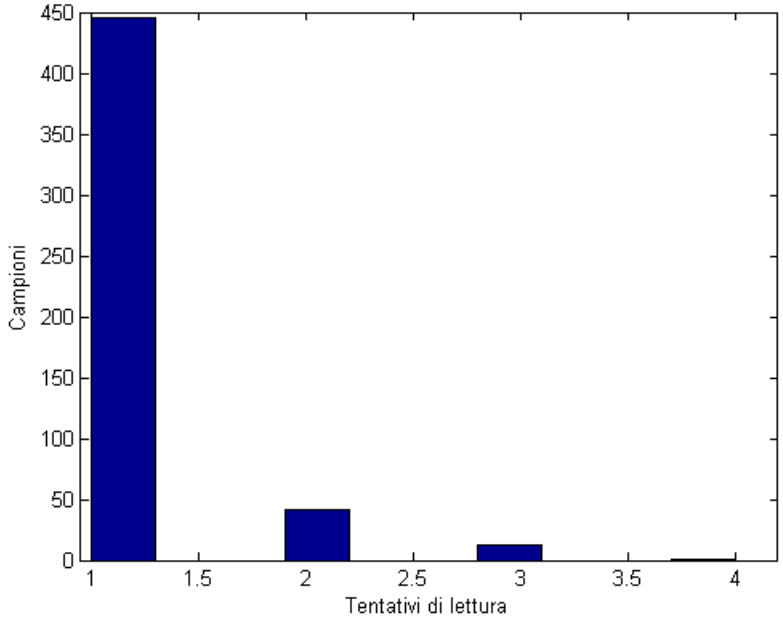

Figura 8.2: Numero di tentativi di autenticazione.

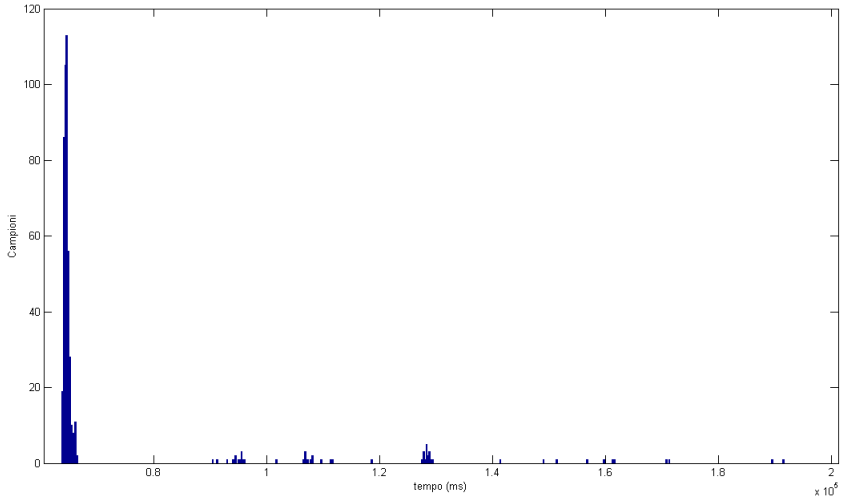

Figura 8.3: Tempi rilevati per l'autenticazione.

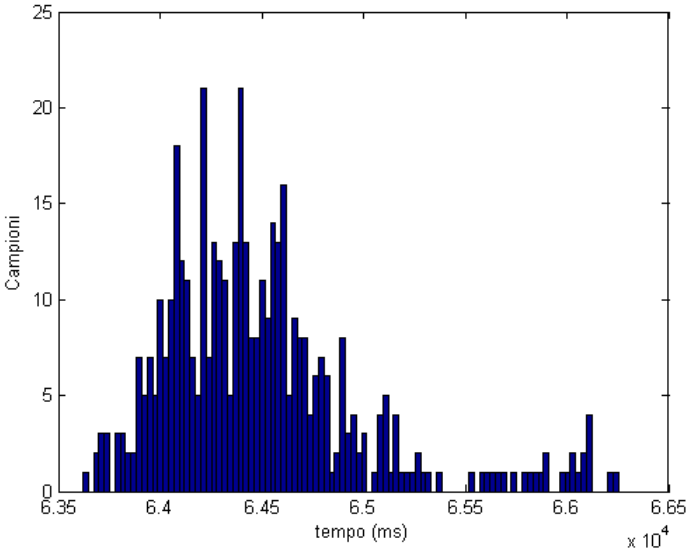

Figura 8.4: Tempi rilevati senza errori di lettura.

senza ricarica almeno per la durata di un interno turno di lavoro, utilizzando delle batterie che siano il più possibile di basso costo e di dimensioni contenute.

Nell'applicazione oggetto del presente lavoro è stato assunto che il *nodo mobile* in dotazione all'operaio possa essere messo in modalità stand-by quando si trova al di fuori dell'area a rischio e sia "risvegliato" secondo il meccanismo descritto nel paragrafo 6.3.2 all'ingresso dell'area.

Sono state analizzate diverse configurazioni di funzionamento a seconda della percentuale di tempo all'interno di un turno lavorativo per la quale il sistema è attivo, questa infatti può variare in funzione di diversi parametri come ad esempio il layout della fabbrica.

Essendo in dispositivi alimentati ad una tensione nominale di 6V, si è ipotizzato di utilizzare per ogni nodo due batterie in serie da 1400 mAh con tensione nominale 3V. Tali batterie sono infatti facilmente reperibili, di basso costo e di dimensioni piuttosto contenute.

Per quanto riguarda il consumo energetico dei nodi si è ipotizzato di avere in funzione il servizio di localizzazione e la *body area network* mediante un dispositivo bluetooth integrato. In questa configurazione il consumo energetico rilevato sperimentalmente è pari a 188,5 mA e quindi 1,13 W con l'alimentazione a 6V, mentre in stand-by mode il consumo rilevato è pari a 59,2 mA con l'alimentazione a 6V, quindi In queste condizioni nel caso di dispositivo sempre attivo la durata sarebbe pari a:

$$D = \frac{B}{C} = \frac{1400}{188,5} = 7,42h$$

dove D è la durata, B è la capacità della batteria e C è il consumo. Quindi la durata non è sufficiente per l'intero turno di lavoro.

Ipotizzando però un'inattività del 10% del tempo dovuta a frangenti della giornata nei quali l'operaio si trova in aree dello stabilimento non considerate a rischio si ottiene che il consumo del nodo è in media pari a:

$$C_m = C_a * 0.9 + C_{sb} * 0.1 = 188,5 * 0.9 + 59.2 * 0.1 = 175,5mA$$

dove C_m è il consumo medio, C_a è il consumo del nodo attivo e C_{sb} è il consumo del nodo in stand-by mode. In queste condizioni la durata delle batterie è pari a:

$$D = \frac{B}{C_m} = \frac{1400}{175,5} = 8h$$

Quindi già appena sufficiente per garantire la fine del turno. Nel caso di 20% di inattività avremo $C_m = 162,6mA$ e quindi una durata pari a:

$$D = \frac{B}{C_m} = \frac{1400}{162,6} = 8,61h$$

Mentre nel caso di inattività del 30% avremo $C_m = 149,7mA$ con una durata paria a:

$$D = \frac{B}{C_m} = \frac{1400}{149,7} = 9,35h$$

e così via.

In figura 8.5 è riportata la durata delle batterie in funzione della percentuale di inattività dei nodi.

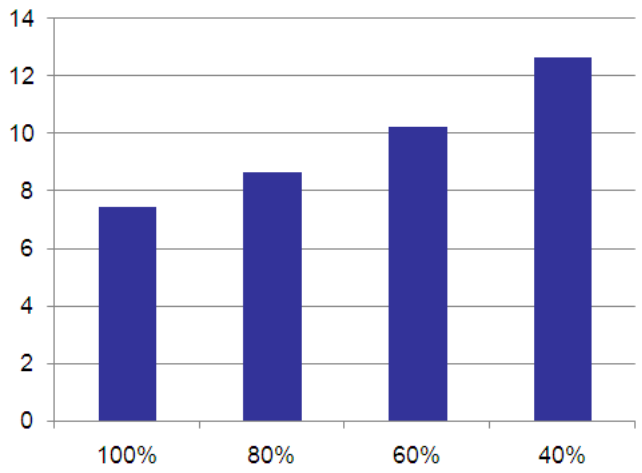

Figura 8.5: Durata delle batterie.

Capitolo 9

Conclusioni

Nel presente lavoro di tesi è stato fornito uno strumento tecnologico avanzato per la gestione e l'ottimizzazione dei livelli di sicurezza all'interno di aree industriali a rischio basato sull'integrazione tra RFID e WSN operanti secondo lo standard IEEE 802.15.4.

Sono stati analizzati nel dettaglio il funzionamento e gli standard di riferimento dei sistemi RFID e WSN, è stata inoltre effettuata un'analisi sullo stato dell'arte per quanto riguarda l'integrazione tra le due tecnologie al fine di identificare le frequenze e gli standard di riferimento più idonei per l'applicazione in oggetto.

Il sistema è stato implementato sia a livello hardware che software fornendo una soluzione innovativa di integrazione per quanto riguarda l'ottimizzazione del consumo energetico dei nodi.

E' stato quindi implementato uno scenario di testing per la validazione del sistema e per l'analisi delle prestazioni. Dai dati sperimentali è emerso che tutte le specifiche sono soddisfatte. L'applicazione proposta sarà oggetto di una sperimentazione in ambiente industriale reale per analizzare nel dettaglio costi e benefici.

Come sviluppo futuro vi è la possibilità di analizzare la fattibilità di una rete ad hoc RFID tra i dispositivi di protezione in dotazione agli operai in modo che il nodo master possa avere in ogni momento la percezione degli oggetti in possesso dell'operaio, in quest'ottica la progettazione del sistema è stata effettuata per garantire un semplice passaggio a questo tipo di scenario.

Altro sviluppo futuro è l'analisi delle performance di applicazioni di questo tipo basate sull'integrazione RFID-WSN in scenari diversi quali ad esempio quello delle *smart cities*, nelle quali si ha sempre di più l'esigenza di strumenti tecnologici intelligenti con un alto livello di pervasività a basso costo per permettere il controllo capillare ad esempio del traffico, dei parcheggi, degli accessi alle zone limitate e così via.

Appendice A

Driver per Erika Enterprise

A.1 Driver per il reader RFID

```
 /*
 * rfid.h
 *
 *  Created on: Oct 25, 2011
 *      Author: Cosimo Salvatore
 *
 * The reader is the Discovery UHF model product by SensorID
 */

#ifndef RFID_H_
#define RFID_H_
#include <stdio.h>
#include "mcu/microchip_pic32/inc/ee_uart.h"

#define UART_PORT EE_UART_PORT_1A
#define FLAGS (EE_UART_BIT8_NO|EE_UART_BIT_STOP_1)



#define MAX_EPC_SIZE 12
#define MAX_PACK_SIZE 64

#define RES_MEM 0 //reserved memory
#define EPC_MEM 1 //epc memory
#define TID_MEM 2 //Tag ID memory
#define USER_MEM 3 //User memory
#define KILL_PASS 4 //Kill password memory
#define ACC_PASS 5 //Access password memory
```

```
/**
* @brief Set serial communication
*
*/
void rfid_init(void);

 /**
* @brief Set output power and receiving sensitivity to the max
*
*/
void set_max_output_power(void);

 /**
* @brief Set frequency of received data to optimize the performance of
*        the the tag IDS sl900a
*
*/
void set_freq_ids(void);

 /**
* @brief Detects the tags and store their epc codes
*
* @param epcs: matrix in which epc codes of the tags are stored
*
* @return Returns the number of tags found
*
*/
unsigned int inventory(unsigned char epcs[][MAX_EPC_SIZE]);

 /**
* @brief Select one tag from those detected
*
* @param epc: pointer to the epc code of the tag to select
*
* @return Returns 0 if no error occurs,
*               otherwise returns a specific rfid reader error
*
*/
unsigned int select_tag(unsigned char *epc);

 /**
* @brief Reads in a specific memory bank of the tag
```

```
*
* @param epc: pointer to the epc code of the tag to read
*
* @param buf: pointer to a buffer to store the data
*
* @param memory bank: address of the specific memory bank,
*                      possible addresses are RES_MEM, EPC_MEM,
*                       TID_MEM, USER_MEM
*
* @param address: tag memory address (in words)
*
* @param data_len: lenght of data to read (in words)
*
* @return Returns the length of data read if no error occurs,
*                otherwise returns  a specific rfid reader error
*
*/
unsigned char read_tag(unsigned char *epc, unsigned char *buf,
                       unsigned int memory_bank, unsigned int address,
                       unsigned char data_len);

 /**
* @brief write data in a specific memory bank of the tag
*
* @param epc: epc code of the tag to read
*
* @param data: pointer to the data to be written
*
* @param data_len: length of the data to be written
*
* @param memory bank: address of the specific memory bank, possible
*                     addresses are EPC_MEM, TID_MEM, USER_MEM
*
* @param address: tag memory address (in words)
*
* @param psw: pointer to the access password of the specific memory bank
*
* @return Returns 0 if no error occurs, otherwise returns a
*               specific rfid reader error
*
*/
unsigned char write_tag(unsigned char *epc, unsigned char *data,
                        unsigned int data_len, unsigned int memory_bank,
                        unsigned int address, unsigned char *psw);
```

```
 /**
* @brief lock a specific memory bank of the tag
*

* @param memory bank: address of the specific memory bank, possible
*                     addresses are EPC_MEM, TID_MEM, USER_MEM
*
*
* @param psw: pointer to the access password of the specific memory bank
*
* @return Returns 0 if no error occurs, otherwise returns a
*               specific rfid reader error
*
*/
unsigned int lock_memory(unsigned char *psw, unsigned int memory_bank);

 /**
* @brief unlock a specific memory bank of the tag
*

* @param memory bank: address of the specific memory bank, possible
*                     addresses are EPC_MEM, TID_MEM, USER_MEM
*
*
* @param psw: pointer to the access password of the specific memory bank
*
* @return Returns 0 if no error occurs, otherwise returns a
*               specific rfid reader error
*
*/
unsigned int unlock_memory(unsigned char *psw, unsigned int memory_bank);


/*
 * rfid.c
 *
 *  Created on: Oct 25, 2011
 *      Author: Cosimo Salvatore
 */
#include "uhf_reader.h"
#include <stdio.h>
#include <stdarg.h>
#include <string.h>
```

```
void set_freq_ids(void){
  unsigned char freq[64]={0x59, 0x40, 0x01, 0x0C, 0x01, 0x03, 0x01,
                                      0x00, 0x01, 0x01, 0x01, 0x05};
  unsigned char ans[MAX_PACK_SIZE];
  writeStr(freq, 64);
  read_packet(ans, 64);
  EE_delay_us(100000);
}

void set_max_output_power(void){
  char msg1[6] = {0x1d, 0x06, 0x06, 0x3f, 0x20, 0x00};
  char msg2[6] = {0x1a, 0x06, 0x15, 0x00, 0x3f, 0x20};
  writeStr(msg1,6);
  EE_delay_us(10000);
  writeStr(msg2,6);
  EE_delay_us(10000);
  char msg3[3] = {0x1c, 0x03, 0x15};
  char msg4[6] = {0x1a, 0x06, 0x15, 0x00, 0x3f, 0x20};
  writeStr(msg3,6);
  EE_delay_us(10000);
  writeStr(msg4,6);
}

unsigned int inventory(unsigned char epcs[][MAX_EPC_SIZE]){
  unsigned char start_inventory[3]={0x31, 0x03, 0x01};
  unsigned int tags_num=0;
  unsigned char ans[MAX_PACK_SIZE];
  int i=0;
  int j=0;
  unsigned int epc_len[10];
  unsigned int last_off_byte=3;
  writeStr(start_inventory, 3);
  read_packet(ans, MAX_PACK_SIZE);
  tags_num=ans[2];
  for(i=0;i<tags_num;i++){
    epc_len[i]=ans[last_off_byte];
    for(j=0;j<epc_len[i]-2;j++){
      epcs[i][j]=ans[last_off_byte+3+j];
    }
    for(j=epc_len[i];j<MAX_EPC_SIZE;j++){
      epcs[i][j]=0;
    }
    last_off_byte += epc_len[i]+4;
  }
```

```
  return tags_num;
}

unsigned int select_tag(unsigned char *epc){
  unsigned int epc_len = 12;
  unsigned char msg[MAX_PACK_SIZE];
  unsigned char ans[3];
  unsigned char err=0;
  if(epc[0] == 0x01){
    epc_len=0x08;
  }
  msg[0]=0x33;
  msg[1]=0x40;
  msg[2]=epc_len;
  int i=0;
  for(i=0;i<epc_len;i++){
    msg[i+3]=epc[i];
    msg[i+4]=0x00;
    msg[i+5]=0x00;
  }
  writeStr(msg,epc_len+5);
  read_packet(ans,3);
  err=ans[2];
  return err;
}

unsigned char read_tag(unsigned char *epc, unsigned char *buf,
                       unsigned int memory_bank, unsigned int address,
                       unsigned char data_len){
  unsigned char msg[9];
  unsigned char err;
  unsigned char ans[MAX_PACK_SIZE];
  unsigned int i=0;
  unsigned int j=0;
  unsigned char length;
  msg[0]=0x37;
  msg[1]=0x09;
  switch (memory_bank) {
    case RES_MEM:
      msg[2]=0x00;
      break;
    case EPC_MEM:
      msg[2]=0x01;
      break;
```

```
    case TID_MEM:
      msg[2]=0x02;
      break;
    case USER_MEM:
      msg[2]=0x03;
      break;
  }
  msg[3]=address;
  msg[4]=data_len;
  msg[5]=0x00;msg[6]=0x00;msg[7]=0x00;msg[8]=0x00;
  for(i=0;i<3;i++){
  err=select_tag(epc);
  if(err==0)
    break;
    else
      return err;
  }
  writeStr(msg,9);
  read_packet(ans,MAX_PACK_SIZE);
  length=ans[3];
  for(j=0;j<length*2;j++)
  buf[j]=ans[j+4];
  return length*2;
}

unsigned char write_tag(unsigned char *epc, unsigned char *data,
                        unsigned int data_len, unsigned int memory_bank,
                        unsigned int address, unsigned char *psw){
  unsigned char msg[MAX_PACK_SIZE];
  unsigned char ans[4];
  unsigned char err;
  int i=0;
  int j=0;
  msg[0]=0x35;
  msg[1]=0x40;
  switch (memory_bank) {
    case EPC_MEM:
      msg[2]=0x01;
      break;
      case TID_MEM:
        msg[2]=0x02;
        break;
      case USER_MEM:
        msg[2]=0x03;
```

```
        break;
  }
  msg[3]= address;
  for(i=0;i<4;i++)
  msg[4+i]=psw[i];
  msg[8]=data_len/2;
  for(j=0;j<data_len;j++){
    msg[9+j]=data[j];
  }
  int k=0;
  for(k=0;k<MAX_PACK_SIZE-9-data_len;k++)
    msg[k+data_len+9]=0x00;   //sistemare data_len (byte-words)
    for(i=0;i<3;i++){
      err=select_tag(epc);
      if(err==0)
        break;
      else
        return err;
  }
  EE_delay_us(10000);
  writeStr(msg,MAX_PACK_SIZE);
  read_packet(ans,4);
  return ans[3];
}

unsigned int lock_memory(unsigned char *psw, unsigned int memory_bank){
  unsigned char msg[8];
  unsigned char ans[3];
  int i=0;
  msg[0]=0x3b;
  msg[1]=0x08;
  msg[2]=0x01;
  switch (memory_bank) {
    case KILL_PASS:
      msg[2]=0x00;
      break;
    case ACC_PASS:
      msg[2]=0x01;
      break;
    case EPC_MEM:
      msg[2]=0x02;
      break;
    case TID_MEM:
      msg[2]=0x03;
```

```
      break;
    case USER_MEM:
      msg[2]=0x04;
      break;
  }
  for(i=0;i<4;i++)
    msg[4+i]=psw[i];
  writeStr(msg,8);
  read_packet(ans,3);
  return ans[3];
}

unsigned int unlock_memory(unsigned char *psw, unsigned int memory_bank){
  unsigned char msg[8];
  unsigned char ans[3];
  int i=0;
  msg[0]=0x3b;
  msg[1]=0x08;
  msg[2]=0x00;
  switch (memory_bank) {
    case KILL_PASS:
      msg[2]=0x00;
      break;
    case ACC_PASS:
      msg[2]=0x01;
      break;
    case EPC_MEM:
      msg[2]=0x02;
      break;
    case TID_MEM:
      msg[2]=0x03;
      break;
    case USER_MEM:
      msg[2]=0x04;
      break;
  }
  for(i=0;i<4;i++)
    msg[4+i]=psw[i];
  writeStr(msg,8);
  read_packet(ans,3);
  return ans[3];
}

/* Serial communication functions */
```

```
int read_packet(char * buf ,unsigned int len  ){
  EE_UINT8 read=0;
  EE_UINT32 count = 350 /* *4 */ ;// Interrarrival number
                                  // of clock cycles
  EE_UINT8 b;
  EE_INT8 ret_val;
  while((read<len)&&(count  > 0)){
    ret_val = EE_uart_read_byte(UART_PORT,&b);
    if(ret_val==-EE_UART_ERR_OVERFLOW)
      break;
    if(ret_val== EE_UART_NO_ERROR){
      count = 350 /* *4 */;
      buf[read]=b;
      read++;
    }
    else
      if (read > 1)
        count--;
      }
    return read;
}
void writeStr(char * str, int len){
  int i;
  for (i = 0; i < len ; i++){
  putByte(str[i]);
  }
}


void readStr(char * buf, int len){
  int i;
  for (i = 0; i < len ; i++){
  buf[i] = getByte();
  }
}


/* A printf-like function */
void myprintf(const char *format, ...)
{
#define MAXCHARS 128
  char printf_trunc[] = "..[TRUNCATED]..\n";
  char str[MAXCHARS];
```

```
  int len;
  va_list args;
  va_start(args, format);
  len = vsnprintf(str, MAXCHARS, format, args);
  va_end(args);
  if (len > MAXCHARS - 1) {
    /* vsnptintf() returns the number of characters needed */
    writeStr(str, MAXCHARS - 1 );
    writeStr(printf_trunc, sizeof(printf_trunc) - 1);
  }
  else {
    writeStr(str, len);
  }
}


EE_INT8 getByte(void){
  EE_INT8 b;
  EE_INT8 r_;
  EE_UINT32 count_ = 1000000;
  //SetRelAlarm(AlarmKillUart, 100, 0);
  while(count_ -- > 0) {
    r_ = EE_uart_read_byte(UART_PORT, &b);
    if( r_ == -EE_UART_ERR_OVERFLOW){
      b=-1;
      break;
    }
    if( r_ == EE_UART_NO_ERROR)
      break;
    if(r_ ==-EE_UART_ERR_NO_DATA){
      b=0;
      break;
    }
  }
  return b;
}



void putByte(EE_UINT8 b){
  EE_uart_write_byte(UART_PORT, b);
}
```

```
void configUART(EE_UINT32 u32_baudRate){
  EE_INT8 retval = EE_uart_init(UART_PORT, u32_baudRate,
                            FLAGS, EE_UART_CTRL_SIMPLE);
  if (retval != EE_UART_NO_ERROR){
    while(1);
  }
}

void rfid_init(void){
  configUART(115200);
}
```

A.2 Driver per il tag IDS sl900a

```
/*
 * ids_sl900a.h
 *
 *  Created on: 11/nov/2011
 *      Author: Cosimo Salvatore
 *
 * The tag chip is the sl900a product by IDS
 */

#ifndef IDS_SL900A_H_
#define IDS_SL900A_H_

#include <stdio.h>
#include "mcu/microchip_pic32/inc/ee_spi.h"

#endif /* IDS_SL900A_H_ */

#define SPI_PORT EE_SPI_PORT_1
#define TAG_TRIS_CS TRISDbits.TRISD9
#define TAG_CS PORTDbits.RD9
/**

* @brief Initialize SPI communication
*
*/
void tag_init(void);

 /**
```

```
* @brief Write into a specific register of the tag memory
*
* @param reg: address of register to be written
*
*/
void write_register(unsigned short int reg, unsigned char data);

 /**
* @brief read the value of a specific register of the tag memory
*
* @param reg: address of register to read
*
* @return Returns the value of the register
*/
unsigned char read_register(EE_UINT16 reg);

 /**
* @brief get the temperature value of the integrated
*         sensor of the tag
* @return Returns the temperature value in °C
*/
double get_temperature(void);

 /**
* @brief get voltage value of the battery (NOT TESTED)
*
* @return Returns the voltage value in V
*/
unsigned char get_battery(void);

 /**
* @brief get voltage value of the battery (NOT TESTED)
*
* @param *data: pointer to data calculated (3 bytes)
*/
void get_shelf_life(unsigned char *data);

 /**
* @brief get the value of the external sensor 1 (NOT TESTED)
*
* @return Returns the value of the external sensor 1
*/
unsigned char get_sensor1(void);
```

```
 /**
* @brief get the value of the external sensor 2 (NOT TESTED)
*
* @return Returns the value of the external sensor 2
*/
unsigned char get_sensor2(void);

 /*
 * ids_sl900a.c
 *
 *  Created on: 11/nov/2011
 *      Author: Cosimo Salvatore
 */

#include <stdio.h>
#include <stdarg.h>
#include <string.h>
#include "mcu/microchip_pic32/inc/ee_spi.h"
#include "ids_sl900a.h"

void tag_init(void){
TAG_TRIS_CS = 0;
EE_spi_init(SPI_PORT, 1000000,
            EE_SPI_MASTER|EE_SPI_SDI_ON_CLOCK_END);
}

void write_register(EE_UINT16 reg, unsigned char data){
  unsigned char byte1=0;
  unsigned char byte2=0;
  byte1 = reg>>8;
  byte2 = reg;
  unsigned char msg[3] = {byte1, byte2, data};
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 3);
  TAG_CS = 0;
}

unsigned char read_register(unsigned short int reg){
  unsigned char byte1=0x40;
  unsigned char byte2;
  unsigned char value;
  unsigned char tmp;
  tmp=reg>>8;
```

```
  byte1 = byte1 | tmp;
  byte2 = reg;
  unsigned char msg[2] = {byte1, byte2};
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 2);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, &value, 1);
  TAG_CS = 0;
  return value;
}

double get_temperature(void){
  unsigned char msg[1]={0xC1};
  unsigned char ans[2];
  double temp;
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 1);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, ans, 2);
  TAG_CS = 0;
  temp = ans[0]<<8 | ans[1];
  temp=temp*0.18-89.3;
  return temp;
}

unsigned char get_battery(void){
  EE_delay_us(10000);
  unsigned char msg[1]={0xC2};
  unsigned char batt;
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 1);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, &batt, 1);
  TAG_CS = 0;
  return batt;
}

unsigned char get_sensor1(void){
  EE_delay_us(10000);
  unsigned char msg[1]={0xC3};
  unsigned char batt;
```

```
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 1);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, &batt, 1);
  TAG_CS = 0;
  return batt;
}

unsigned char get_sensor2(void){
  EE_delay_us(10000);
  unsigned char msg[1]={0xC3};
  unsigned char batt;
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 1);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, &batt, 1);
  TAG_CS = 0;
  return batt;
}

void get_shelf_life(unsigned char *data){
  unsigned char msg[1]={0xC8};
  TAG_CS = 1;
  EE_delay_us(1000);
  EE_spi_write(SPI_PORT, msg, 1);
  EE_delay_us(10000);
  EE_spi_read(SPI_PORT, data, 3);
  TAG_CS = 0;
}


 /*
 * sleep.h
 *
 *  Created on: 15/dic/2011
 *      Author: Cosimo Salvatore
 */
#ifndef SLEEP_H_
#define SLEEP_H_

#endif /* SLEEP_H_ */
```

```
#include "mcu/microchip_pic32/inc/ee_cn.h"

#define SL900A_PIN_WAKE_INIT() EE_cn_init(7, 3)
#define SL900A_PIN_WAKE_START() EE_cn_enable(sl900a)
#define SL900A_PIN_WAKE_STOP() EE_cn_disable(sl900a)
#define SL900A_WAKE_RESET_IF()
#define SL900A_WAKE_INTERRUPT()  EE_CN_HANDLER(sl900a)
#define SL900A_WAKE_VALUE()  PORTDbits.RB3

#define SL900A_DISABLE_INTERRUPTS() asm volatile("di")
#define SL900A_ENABLE_INTERRUPTS() asm volatile("ei")
```

A.3 Implementazione della funzione di interrupt in ERIKA

```
 /*
 * sleep.c
 *
 *  Created on: 15/dic/2011
 *      Author: Cosimo Salvatore
 */
#include "mcu/microchip_pic32/inc/ee_i2c.h"
#include "mcu/microchip_pic32/inc/ee_oc.h"
#include "mcu/microchip_pic32/inc/ee_dma.h"

#include "cpu/pic32/inc/ee_utils.h"
#include <plib.h>
#include "sleep.h"

SL900A_WAKE_INTERRUPT(){
  EE_leds_toggle();
  SL900A_WAKE_RESET_IF();
  SoftReset();
}
```

Appendice B

Inizializzazione della memoria

```
/*
 * iniizializza.h
 *
 *  Created on: 22/nov/2011
 *      Author: Cosimo Salvatore
 */

#ifndef INIIZIALIZZA_H_
#define INIIZIALIZZA_H_


#endif /* INIIZIALIZZA_H_ */
#include <stdio.h>
#include "ids_sl900a.h"

#define IN_MEM1 0x65
#define IN_MEM2 0x8D

/****************DEFINE OBJECTS******************/
#define CASCO 0
#define SCARPADX 1
#define SCARPASX 2
#define OCCHIALI 3
#define GUANTODX 4
#define GUANTOSX 5
#define GIUBBETTO 6
#define MASCHERINA 7
```

```
/*****************DEFINE AREAS*******************/
#define PRODUZIONE 1
#define CALDAIA 2
#define MAGAZZPF 3
#define MAGAZZMP 4
#define UFFICIO 5
#define LABORAT 6

/**
* @brief Insert identification code of an object
*        in the USER memory of the tag
*
* @param obj_type: Type of the object to insert
*
* @param obj_id: Identification code of the object
*
*/
void insert_obj_id( EE_UINT8 obj_type, EE_UINT16 id_obj);

/**
* @brief Adds a request of an object for a
*        specific plant area
*
* @param obj: Object type
*
* @param area: Plant area
*
*/
void add_obj_req(EE_UINT8 area, EE_UINT8 obj);

/**
* @brief Deletes a request of an object for a
*        specific plant area
*
* @param obj: Object type
*
* @param area: Plant area
*
*/
void del_obj_req(EE_UINT8 area, EE_UINT8 obj);
```

Appendice C

Meccanismo di autenticazione

```
 /*
 * autentication.h
 *
 *  Created on: 25/nov/2011
 *      Author: Cosimo Salvatore
 */

#ifndef AUTENTICATION_H_
#define AUTENTICATION_H_


#endif /* AUTENTICATION_H_ */

#include <stdio.h>
#include "uhf_reader.h"

#define MEM2 20
#define MEM1 1

/****************DEFINE OBJIECTS******************/
#define CASCO 1
#define SCARPADX 2
#define SCARPASX 3
#define OCCHIALI 4
#define GUANTODX 5
#define GUANTOSX 6
#define GIUBETTO 7
#define MASCHERINA 8
```

```
/*****************DEFINE AREAS*******************/
#define PRODUZIONE 1
#define CALDAIA 2
#define MAGAZZPF 3
#define MAGAZZMP 4
#define UFFICIO 5
#define LABORAT 6

EE_UINT16 autenticate(EE_UINT8 area,
                      EE_UINT8 epcs[][12], EE_UINT8 tags_num);
EE_UINT8 match_obj_id(EE_UINT8 *epc, EE_UINT16 obj_id);

 /*
 * autentication.c
 *
 *  Created on: 25/nov/2011
 *      Author: Cosimo Salvatore
 */

#include "autentication.h"
#include <stdio.h>
#include <stdarg.h>
#include <string.h>


EE_UINT16 autenticate(EE_UINT8 area, EE_UINT8 epcs[][12],
                                      EE_UINT8 tags_num){
 EE_UINT16 required=0;
 EE_UINT16 found=0;
 EE_UINT8 id[2]={0x00, 0x00};
 EE_UINT8 read[2]={0x00,0x00};
 EE_UINT16 object=0;
 EE_UINT16 tmp=0;
 EE_UINT16 cfr=1;
 int i=0;
 int j=0;
 int k=0;
 for(i=0;i<tags_num;i++){                //controllo se c'è il nodo
     if(epcs[i][0] == 0x01){
           read_tag(epcs[i], read, USER_MEM, MEM2+area, 1);
           required=read[0];
           required=required<<8;
           required=required | read[1];
```

```
            for(j=0;j<16;j++){           //cerco gli oggetti necessari
                tmp = required >> j;
                tmp = tmp & cfr;
                if(tmp == 1){
                    read_tag(epcs[i], id,
                             USER_MEM, MEM1+j, 2); //salvo l'id degli
                    object=id[0];                 //oggetti necessari
                    object=object<<8;
                    object=object | id[1];
                    for(k=0;k<tags_num;k++){  //controllo se tra i tag
                                              //trovati c'è l'oggetto id
                            if(match_obj_id(epcs[k], object)){

                            found=found | (cfr<<j);  //aggiorno maschera
                            EE_led_on(1);             //degli oggetti
                        }                             //trovati
                    }
                }
            }
    }
 }
 return required ^ found;           //ritorno maschera
}                                   //degli oggetti non trovati


EE_UINT8 match_obj_id(EE_UINT8 *epc, EE_UINT16 obj_id){
   EE_UINT16 tag_code=0;
   tag_code = epc[5];
   tag_code = tag_code << 8;
   tag_code = tag_code | epc[6];
   if(tag_code == obj_id)
        return 1;
        else
        return 0;
}
```

Appendice D

Implementazione del sistema

D.1 Implementazione del codice del nodo coordinator

```
 #include "uwl_ieee802154.h"
#include "lwp_6lowpan.h"
#include "kal/uwl_kal.h"
#include "ee.h"
#include "console_serial.h"
#include <stdarg.h>
#include <string.h>
#include <stdio.h>
#include "mcu/microchip_pic32/inc/ee_timer.h"
#include "uhf_reader.h"
#include "mcu/microchip_pic32/inc/ee_spi.h"
#include "autentication.h"
#include "seriale2.h"

#include <math.h>
/* ************************************************************ */
/*                          FUSE Bits                           */
/* ************************************************************ */
/*  Initial Configuration
 *
 *  Main Clock       -> SYSCLK = (Crystal_Freq / FPLLIDIV) * FPLLMUL /
                                                            FPLLODIV
 *  Peripheral Clock -> PBCLK  = SYSCLK / FPBDIV
 */
#pragma config POSCMOD = HS
#pragma config FNOSC = PRIPLL
#pragma config FPLLMUL = MUL_20
```

```
#pragma config FPLLIDIV = DIV_2
#pragma config FPLLODIV = DIV_1
#pragma config FPBDIV = DIV_1   /* SYSCLK = 80MHz, PBCLK = 80MHz */

#pragma config FWDTEN = OFF

/*****************************************************************/
/*                      Constant Definitions                     */
/*****************************************************************/

#define TEST_PANID              0x000A
#define TEST_COORD_ADDR         0x0001
//#define TEST_DEV_ADDR         0x0034
#define TEST_CHANNEL            23
#define TEST_BO                 15
#define TEST_SO                 6

#define BORDER_ROUTER           1

#define MAX_PCK_LEN             127
#define USE_GTS                 0
#define DO_NOT_USE_GTS          0

#define MAX_MSG_SIZE            80
#define PACKET_SIZE             10

#define DATA_PORT               9761
#define HOP_LIMIT               64

#define sw2                     PORTDbits.RD5 //button 2 pin
#define BUTTONPRESSED           sw2==0

uint8_t rcv_packet[MAX_MSG_SIZE];
uint8_t snd_packet[PACKET_SIZE];
uint16_t src_IP_addr[8];
uint16_t dst_IP_addr[8];

uint8_t status_localization = 0;

#define LOC_ON                  1
#define LOC_OFF                 0

#define MOBILE_ADDR             0x0030
```

```
#define TYPE_CMD_LOC    1 //from coordinator in multicast
                              (on/off localization system)
#define TYPE_CMD_FIND   2 //from mobile to anchor
#define TYPE_CMD_RSSI   3 //from anchor to coordinator


/*****************************************************************/
/*                    Function Definitions                       */
/*****************************************************************/
//console_descriptor_t *my_console_1;
#ifdef __32MX795F512L__
#define SERIAL EE_UART_PORT_1B
#else //! __32MX795F512L__
#define SERIAL EE_UART_PORT_2
#endif

EE_UINT8 cont=0;

/*****************************************************************/
/*                        Localization                           */
/*****************************************************************/
#define NUM_ANCHOR         3

#define ANC_21_X_POS     100
#define ANC_21_Y_POS     100
#define ANC_22_X_POS     100
#define ANC_22_Y_POS     100
#define ANC_23_X_POS     100
#define ANC_23_Y_POS     100

#define GT       1.0
#define GR       1.0
#define PT       1.0 //mW

#define C        299792458.0
#define FREQ     2400000000.0
#define LAMBDA   C/FREQ
#define N        3.0

#define SQRT(value, n)          pow(value, (1/n))
#define DBM_TO_MW(value)        pow(10.0, (value/10.0))

#define ATT                     pow((LAMBDA/(4*M_PI)), 2)
```

```
uint8_t anc_rssi[NUM_ANCHOR];
double anc_dist[NUM_ANCHOR];

uint8_t num_rssi = 0;

static void calculate_distance_from_anc_to_node(void){
  uint8_t i = 0;
  for(i = 0; i < NUM_ANCHOR; i++){

anc_dist[i] = (PT * GT * GR * ATT) /
DBM_TO_MW(anc_rssi[i]);

anc_dist[i] = SQRT(anc_dist[i], N);

anc_dist[i] = anc_dist[i] * 100.0; //cm
}

}

static void calculate_node_position(double d1, double d2,double d3,
double x1, double x2, double x3, double y1, double y2, double y3)
{
double x, y, A, B, CC;

uint16_t x_int = 0;
uint16_t y_int = 0;

A = pow(d1, 2) - pow(d3, 2) - pow(x1, 2) + pow(x3, 2)
                                    - pow(y1, 2)+ pow(y3, 2);
B = pow(d2, 2) - pow(d3, 2) - pow(x2, 2) + pow(x3, 2)
                                   - pow(y2, 2) + pow(y3, 2);
CC = 2 * (y3 - y2);
CC -= ((2 * (y3 - y1) * (x3 - x2)) / (x3 - x1));

y = ((B * (x3 - x1)) - (A * (x3 - x2)));
y = y / ((x3 - x1) * CC);
x = (A - (2 * y * (y3 - y1))) / (2 * (x3 - x1));

x_int = (uint16_t) x * 100;
y_int = (uint16_t) y * 100;

//TODO: inviare posizione alla GUI x_int y_int
//writeStr2(reset, 5);
```

```
//writeStr2(msg, 5);
}

void calculate_location()
{
calculate_distance_from_anc_to_node();

calculate_node_position(anc_dist[0], anc_dist[1],
anc_dist[2], ANC_21_X_POS,
ANC_22_X_POS, ANC_23_X_POS,
ANC_21_Y_POS, ANC_22_Y_POS, ANC_23_Y_POS);
}

void save_rssi(uint8_t rssi, uint16_t anchor_addr)
{
int8_t i = -1;

if(anchor_addr == 0x0021){
i = 0;
} else if(anchor_addr == 0x0022){
i = 1;
} else if(anchor_addr == 0x0023){
i = 2;
}

if(i < 0){
return;
}

if(anc_rssi[i] == 0){
anc_rssi[i] = rssi;
num_rssi++;
} else {
return;
}

if(num_rssi == NUM_ANCHOR){
calculate_location(anc_rssi[0], anc_rssi[1], anc_rssi[2]);
num_rssi = 0;
}
}


static void rx_data(int8_t status, uint8_t *data, uint16_t len,
```

```
uint16_t *addr, uint16_t dst_port, uint16_t src_port)
{

if (len <= MAX_MSG_SIZE) {
memcpy(rcv_packet, data, len);
memcpy(src_IP_addr, addr, sizeof(uint16_t) * 8);

//ActivateTask(RECEIVE);
}
}
/*****************************************************************/
/*                          MAIN                                 */
/*****************************************************************/
EE_INT8 retv;
int main(void)
{
EE_system_init();
EE_leds_init();
EE_timer_soft_init(EE_TIMER_1, 100000);
EE_timer_start(EE_TIMER_1);

/* Init the Network Stack */
lwp_udp_set_rx_callback(rx_data);

retv = netInt_init_coordinator(TEST_COORD_ADDR, TEST_PANID,
TEST_CHANNEL, TEST_BO, TEST_SO);

if (retv < 0) {
//ERRORE
for (;;) ;
}

serial2_init();
rfid_init();
//EE_led_on(1);
//set_max_output_power();
//set_freq_ids();
EE_delay_us(100000);
//ActivateTask(SENDER);
SetRelAlarm(SenderAlarm , 10, 1000);

/* Forever loop: background activities (if any) should go here */
for (;;) {
```

```
}
return 0;

/*****************************************************************/
/*                          TASKs                                */
/*****************************************************************/
/* ---------------------------------------------------------------
|  Test 6LoWPAN TX                                                |
 ---------------------------------------------------------------- */
TASK(SEND_MULTICAST)
{
uint8_t packet_size = 0;

snd_packet[0] = TYPE_CMD_LOC;

status_localization = !status_localization;

//if(status_localization){
// turn on localization
snd_packet[1] = LOC_ON;
snd_packet[2] = cont;
//} else {
//snd_packet[1] = LOC_OFF;
//}

packet_size = 3;

lwp_net_addr_set_multRouter(dst_IP_addr);

lwp_udp_send(snd_packet, packet_size, dst_IP_addr, HOP_LIMIT,
DATA_PORT, DATA_PORT);

}

/* ---------------------------------------------------------------
|  Test 6LoWPAN RX                                                |
 ---------------------------------------------------------------- */
TASK(RECEIVE)
{

if(src_IP_addr[7] != MOBILE_ADDR){
switch(rcv_packet[0]){
case TYPE_CMD_RSSI:
save_rssi(rcv_packet[0], src_IP_addr[7]);
```

```
break;
}
}
}

EE_UINT16 maschera=0xffff;
unsigned char mask[2]={0xff,0xff};
EE_UINT16 operaio;
TASK(SENDER){
EE_UINT16 errore=0;
int tags;
unsigned char code[5][12];
unsigned char mem_w[2]={0x11, 0x11};
unsigned char mem_r[64];
unsigned char password[4]={0x00, 0x00, 0x00, 0x00};
unsigned int lung;
int i=0;
int j=0;

for(j=0;j<12;j++){
code[0][j]=0;
code[1][j]=0;
}
cont++;
EE_led_toggle(0);
tags=inventory(code);

    if(tags>=1){
retv = lwp_net_init(BORDER_ROUTER);

if (retv < 0) {
//ERRORE
for (;;) ;
}

     errore = autenticate(PRODUZIONE, code, tags, &maschera, &operaio);
     if(errore==0){
     mask[1]=maschera;
     maschera=maschera>>8;
     mask[0]=maschera;
ActivateTask(SERIAL);
     }
    }
}
```

```
TASK(SERIAL){
unsigned char op[2]={0x00,0x00};
op[1]=operaio;
operaio=operaio>>8;
op[0]=operaio;
unsigned char msg[6]={PRODUZIONE, op[0], op[1], mask[0], mask[1],0xef};
unsigned char reset[6]={0xff,0xff,0xff,0xff,0xff,0xef};
writeStr2(reset, 6);
writeStr2(msg, 6);
read_packet2(acc, 2);

if(mask[1]==0){
putByte2(cont);
}
if(acc[0]==0x30){
EE_led_on(1);
ActivateTask(SEND_MULTICAST);
}

}
/*****************************************************************/
/*                            ISRs                               */
/*****************************************************************/

/*****************************************************************/
/*                            TRAPs                              */
/*****************************************************************/

/*****************************************************************/
/* General Excepiton Handler: overrides the default handler      */
/*****************************************************************/
static enum
{
EXCEP_IRQ = 0, // interrupt
EXCEP_AdEL = 4, // address error exception (load or ifetch)
EXCEP_AdES, // address error exception (store)
EXCEP_IBE, // bus error (ifetch)
EXCEP_DBE, // bus error (load/store)
EXCEP_Sys, // syscall
EXCEP_Bp, // breakpoint
EXCEP_RI, // reserved instruction
EXCEP_CpU, // coprocessor unusable
EXCEP_Overflow, // arithmetic overflow
EXCEP_Trap, // trap (possible divide by zero)
```

```
EXCEP_IS1 = 16, // implementation specfic 1
EXCEP_CEU, // CorExtend Unuseable
EXCEP_C2E
// coprocessor 2
} cp0_exception_code;

static unsigned int cp0_exception_cause; // CP0: CAUSE register
static unsigned int cp0_exception_epc; // CP0: Exception Program Counter

void _general_exception_handler(void)
{
asm volatile("mfc0 %0,$14" : "=r" (cp0_exception_epc));
asm volatile("mfc0 %0,$13" : "=r" (cp0_exception_cause));
cp0_exception_code = (cp0_exception_cause >> 2) & 0x0000001F;
while (1)
;
}
```

D.2 Implementazione del codice del nodo mobile

```
 #include "uwl_ieee802154.h"
#include "lwp_6lowpan.h"
#include "kal/uwl_kal.h"
#include "ee.h"
#include "console_serial.h"
#include <stdarg.h>
#include <string.h>
#include <stdio.h>
#include <plib.h>
#include "mcu/microchip_pic32/inc/ee_timer.h"
#include "uhf_reader.h"
#include "mcu/microchip_pic32/inc/ee_spi.h"
#include "set_memory.h"
#include "ee_cn_config.h"
#include "sleep.h"
#include "ucronos.h"
/* ************************************************************** */
/*                        FUSE Bits                               */
/* ************************************************************** */
/*  Initial Configuration
 *
 *  Main Clock       -> SYSCLK = (Crystal_Freq / FPLLIDIV) * FPLLMUL / FPLLODIV
```

```
 *  Peripheral Clock -> PBCLK  = SYSCLK / FPBDIV
 */
#pragma config POSCMOD = HS
#pragma config FNOSC = PRIPLL
#pragma config FPLLMUL = MUL_20
#pragma config FPLLIDIV = DIV_2
#pragma config FPLLODIV = DIV_1
#pragma config FPBDIV = DIV_1   /* SYSCLK = 80MHz, PBCLK = 80MHz */

#pragma config FWDTEN = OFF

/*******************************************************************/
/*                  Constant Definitions                           */
/*******************************************************************/

#define TEST_PANID 0x000A
#define TEST_COORD_ADDR 0x0001
#define TEST_DEV_ADDR 0x0030
#define TEST_CHANNEL 23
#define TEST_BO 15
#define TEST_SO 6
#define MAX_PCK_LEN 127
#define PACKET_SIZE 10
#define USE_GTS 0
#define DO_NOT_USE_GTS 0
#define SIMPLE_ROUTER 0
#define DATA_PORT 9761
#define HOP_LIMIT 64

#define SEND_OFFSET  1000
#define SEND_PERIOD 10000

#define LOC_ON 1
#define LOC_OFF 0

#define MOBILE_ADDR 0x0030

#define TYPE_CMD_LOC 1 //from coordinator in multicast
                            (on/off localization system)
#define TYPE_CMD_FIND 2 //from mobile to anchor
#define TYPE_CMD_RSSI 3 //from anchor to coordinator


#define sw2                 PORTDbits.RD5 //button 2 pin
```

```
#define BUTTONPRESSED sw2==0

#define SPI_PORT EE_SPI_PORT_1
#define ACC_TRIS_CS TRISDbits.TRISD9
#define ACC_CS PORTDbits.RD9

/*****************************************************************/
/*                        Function Definition                    */
/*****************************************************************/
//console_descriptor_t *my_console_1;
#ifdef __32MX795F512L__
#define SERIAL EE_UART_PORT_1B
#else //! __32MX795F512L__
#define SERIAL EE_UART_PORT_2
#endif

uint8_t rcv_packet[MAX_PCK_LEN];
uint8_t snd_packet[PACKET_SIZE];
uint16_t src_IP_addr[8];

uint16_t dst_IP_addr[8];

struct ucronos_time_t route_time;

union dato_send_app_t {
int8_t bytes[4];
uint32_t t;
}dato_send_app;



static void rx_data(int8_t status, uint8_t *data, uint16_t len,
uint16_t *addr, uint16_t dst_port, uint16_t src_port)
{

if (len <= MAX_PCK_LEN) {
memcpy(rcv_packet, data, len);
memcpy(src_IP_addr, addr, sizeof(uint16_t) * 8);

ActivateTask(RECEIVE);
}
}
/*****************************************************************/
/*                          MAIN                                 */
```

```
/*****************************************************************/
EE_INT8 retv;
int main(void)
{
EE_system_init();
EE_leds_init();
EE_timer_soft_init(EE_TIMER_1, 100000);
EE_timer_start(EE_TIMER_1);

/* Init the Network Stack */
/* lwp_udp_set_rx_callback(rx_data);

retv = netInt_init_device(TEST_DEV_ADDR, TEST_COORD_ADDR,
TEST_PANID, TEST_CHANNEL);

if (retv < 0) {
//ERRORE
for (;;) ;
}

retv = lwp_net_init(SIMPLE_ROUTER);
if (retv < 0) {
//ERRORE
for (;;) ;
}
*/
/*******************************DAQ*********************************/
ucronos_init(80000);
/*******************************DAQ*********************************/

tag_init();
rfid_init();
MOS_TRIS_CS = 0;

unsigned char s=0xaa;
s=read_register(0x065);

EE_led_on(1);
putByte(s);


SL900A_PIN_WAKE_INIT();
SL900A_PIN_WAKE_START();
AD1PCFG = 0x08;
```

```
//Put PIC in sleep mode
    OSCCONSET = 0x10; // Device will enter SLEEP mode
    PowerSaveSleep();

/******************************DEBUG*******************************/
ucronos_start(0);
/******************************DEBUG*******************************/

    //EE_leds_toggle();
    //unsigned char ciao[2]={0x01, 0x02};
    //writeStr(ciao, 2);
     /* Forever loop: background activities (if any) should go here */
for (;;) {

}
return 0;
}
/*******************************************************************/
/*                            TASKs                                */
/*******************************************************************/
/* ----------------------------------------------------------------
|  Test 6LoWPAN TX                                                 |
 ---------------------------------------------------------------- */
TASK(SEND)
{
uint8_t packet_size;

snd_packet[0] = TYPE_CMD_FIND;
packet_size = 1;

lwp_net_addr_set_multRouter(dst_IP_addr);

lwp_udp_send(snd_packet, packet_size, dst_IP_addr, HOP_LIMIT,
DATA_PORT, DATA_PORT);
}

/* ----------------------------------------------------------------
|  Test 6LoWPAN RX                                                 |
 ---------------------------------------------------------------- */
TASK(RECEIVE)
{
char str [100];
```

```
if(src_IP_addr[7] == TEST_COORD_ADDR){
switch(rcv_packet[0]){
case TYPE_CMD_LOC:
if(rcv_packet[1] == LOC_ON){
//SetRelAlarm(SEND_ALARM, SEND_OFFSET, SEND_PERIOD);

/**********************************DAQ***************************/
ucronos_get(0, &route_time);
dato_send_app.t = ucronos_2us(&route_time);


//time[0] = dato_send_app.bytes[3];
//time[1] = dato_send_app.bytes[2];
//time[2] = dato_send_app.bytes[1];
//time[3] = dato_send_app.bytes[0];
long unsigned int microseconds = ((unsigned int)
    dato_send_app.bytes[3]) *1000000
    +((unsigned int) dato_send_app.bytes[2]) *10000
    + ((unsigned int) dato_send_app.bytes[1] *100)
    + dato_send_app.bytes[0];
sprintf(str, "Duration: %lu\r\n", microseconds);
//writeStr(time, 4);
writeStr(str, strlen(str));

sprintf(str, "Numero Letture: %u\r\n", rcv_packet[2]);
writeStr(str, strlen(str));

EE_delay_us(100000000);

/*******************************DAQ*********************************/
}

if(rcv_packet[1] == LOC_OFF) {
CancelAlarm(SEND_ALARM);
}
break;
}
}
}




/*******************************************************************/
```

```
/*                              ISRs                                  */
/*********************************************************************/

/*********************************************************************/
/*                            TRAPs                                   */
/*********************************************************************/

/*********************************************************************/
/* General Excepiton Handler: overrides the default handler          */
/*********************************************************************/
static enum
{
EXCEP_IRQ = 0, // interrupt
EXCEP_AdEL = 4, // address error exception (load or ifetch)
EXCEP_AdES, // address error exception (store)
EXCEP_IBE, // bus error (ifetch)
EXCEP_DBE, // bus error (load/store)
EXCEP_Sys, // syscall
EXCEP_Bp, // breakpoint
EXCEP_RI, // reserved instruction
EXCEP_CpU, // coprocessor unusable
EXCEP_Overflow, // arithmetic overflow
EXCEP_Trap, // trap (possible divide by zero)
EXCEP_IS1 = 16, // implementation specfic 1
EXCEP_CEU, // CorExtend Unuseable
EXCEP_C2E
// coprocessor 2
} cp0_exception_code;

static unsigned int cp0_exception_cause; // CP0: CAUSE register
static unsigned int cp0_exception_epc; // CP0: Exception Program Counter

void _general_exception_handler(void)
{
asm volatile("mfc0 %0,$14" : "=r" (cp0_exception_epc));
asm volatile("mfc0 %0,$13" : "=r" (cp0_exception_cause));
cp0_exception_code = (cp0_exception_cause >> 2) & 0x0000001F;
while (1)
;
}
```

Appendice E

Implementazione della GUI

```
 /*
 *
 * Created on: 15/dic/2011
 *   Author: Cosimo Salvatore
 */

#ifndef GESTIONE_FILE_H
#define GESTIONE_FILE_H

#include <QFile>
#include <QTextStream>
#include <QCoreApplication>
#include <qtest.h>

class Gestione_file : public QObject{
Q_OBJECT

public:
Gestione_file(QObject *parent = 0);
virtual ~Gestione_file();
void leggiarea(QString id_area,QString id_persona);
//QString nome_trovato;
//qint64 soglia_ms;
signals:
void passa_nome(QString nome);
void non_autorizzato();
};

#endif

 /*
```

```
 *
 * Created on: 15/dic/2011
 *   Author: Cosimo Salvatore
 */

#include "gestione_file.h"
#include <QDateTime>
#include <QDebug>
#include <QMutex>
#include <QTime>
#include <QTextStream>

Gestione_file::Gestione_file(QObject * parent )
:QObject(parent)
{
}

Gestione_file::~Gestione_file()
{
}

void Gestione_file::leggiarea(QString id_area,QString id_persona)
{
  bool trovato=FALSE;
  static QMutex mutex;
  mutex.lock();
  if(id_area.contains("01")){
     QFile file_conf(QCoreApplication::
                     applicationDirPath()+ "/produzione.txt");
     if (!file_conf.open(QIODevice::ReadOnly |QIODevice::Text)){
       //qDebug()<< "Error to open"+ (QCoreApplication::
                       applicationDirPath() + "/config_file.txt");

     }
     TextStream in(&file_conf);
     while (!in.atEnd()&&!trovato){
        QString line1 = in.readLine();
        QString ID_database=line1.mid(0,4);
        if (ID_database.contains(id_persona)){
           rovato = TRUE;
           int openIdx1 = 0;
           openIdx1 = line1.indexOf('[', openIdx1+1);
           int closeIdx1 = line1.indexOf(']', openIdx1);
           if((openIdx1 == -1) || (closeIdx1 == -1)){
```

```
            //qDebug()<<"Errore";
         }
         QString nome_trovato =line1.mid(openIdx1 + 1,
                                         closeIdx1-openIdx1 - 1);
         emit passa_nome(nome_trovato);
      }
      if(trovato == FALSE){
         emit non_autorizzato();
      }
   }
}
else if(id_area.contains("02")){
   QFile file_conf(QCoreApplication::applicationDirPath()+
                                              "/caldaia.txt");
   if (!file_conf.open(QIODevice::ReadOnly | QIODevice::Text)){
      //qDebug()<< "Error to open"+ (QCoreApplication::
                         applicationDirPath()+"/config_file.txt");
   }
   QTextStream in(&file_conf);
   while (!in.atEnd()&&!trovato){
      QString line1 = in.readLine();
      QString ID_database=line1.mid(0,4);
      if (ID_database.contains(id_persona)){
         trovato = TRUE;
         int openIdx1 = 0;
         openIdx1 = line1.indexOf('[', openIdx1+1);
         int closeIdx1 = line1.indexOf(']', openIdx1);
         if((openIdx1 == -1) || (closeIdx1 == -1)){
            //qDebug()<<"Errore";
         }
         QString nome_trovato =line1.mid(openIdx1 + 1,
                                         closeIdx1-openIdx1 - 1);
         emit passa_nome(nome_trovato);
         }
         if(trovato == FALSE){
            emit non_autorizzato();
         }
      }
   }
   else if(id_area.contains("03")){
      QFile file_conf(QCoreApplication::
                      applicationDirPath()+ "/magazzino.txt");
      if (!file_conf.open(QIODevice::ReadOnly | QIODevice::Text)){
         //qDebug()<< "Error to open"+ (QCoreApplication::
```

```
                            applicationDirPath()+ "/config_file.txt");

        }
        QTextStream in(&file_conf);
        while (!in.atEnd()&&!trovato){
           QString line1 = in.readLine();
           QString ID_database=line1.mid(0,4);
           if (ID_database.contains(id_persona)){
              trovato = TRUE;
              int openIdx1 = 0;
              openIdx1 = line1.indexOf('[', openIdx1+1);
              int closeIdx1 = line1.indexOf(']', openIdx1);
              if((openIdx1 == -1) || (closeIdx1 == -1)){
                 //qDebug()<<"Errore";
              }
              QString nome_trovato =line1.mid(openIdx1 + 1,
                                              closeIdx1-openIdx1 - 1);
              emit passa_nome(nome_trovato);
           }
           if(trovato == FALSE){
              emit non_autorizzato();
           }
       }
    }
    mutex.lock();
}

 /*
 *
 * Created on: 15/dic/2011
 *   Author: Cosimo Salvatore
 */
#include "virtual_lcd_debug.h"


Virtual_LCD_Debug::Virtual_LCD_Debug(QWidget *parent, Qt::WFlags flags)
: QMainWindow(parent, flags)
{
ui.setupUi(this);
ui.textBrowser_Led->setStyleSheet("background-color: green");
ui.textBrowser_Led_2->setStyleSheet("background-color: green");
ui.textBrowser_Led_3->setStyleSheet("background-color: green");
database=new(Gestione_file);
```

```
//ReceiverThread = new Receiver(com, this);
QObject::connect(this, SIGNAL(receivedMessage(QString)),this,
                                        SLOT(answer_parsing(QString)));
QObject::connect(ui.pushButton_ricevi,SIGNAL(clicked()),this,
                                                    SLOT(ricevi()));
QObject::connect(ui.pushButton_stop,SIGNAL(clicked()),this,
                                                  SLOT(ferma_ricevi()));
QObject::connect(ui.pushButton,SIGNAL(clicked()),this,
                                                   SLOT(open_serial()));
QObject::connect(database,SIGNAL(passa_nome(QString)),this,
                                            SLOT(scrivi_nome(QString)));
QObject::connect(database,SIGNAL(non_autorizzato()),this,
                                               SLOT(errore_persona()));
}

Virtual_LCD_Debug::~Virtual_LCD_Debug()
{
}

void Virtual_LCD_Debug::ricevi()
{
ReceiverThread = new Receiver(com, this);
QObject::connect(ReceiverThread, SIGNAL(receivedMessage(QString)),
                                     this, SLOT(answer_parsing(QString)));
this->ReceiverThread->initReceiver();
this->ReceiverThread->start();
}

void Virtual_LCD_Debug::ferma_ricevi()
{
this->ReceiverThread->stopReceiver();
this->ReceiverThread->exit(0);
//this->ReceiverThread->terminate();
this->ReceiverThread->wait(500);
/* sync to thread */
while (this->ReceiverThread->isRunning())
{
this->ReceiverThread->exit(0);
this->ReceiverThread->wait(500);
}
}
void Virtual_LCD_Debug::answer_parsing(QString message)
{
```

```
QByteArray messaggio;
QString comando;
QString lungh_dati;
unsigned long time=3;
int crc;
QString SOF;
int crc_calc;
int lung_dat;
int lung,i, colore, riga, colonna, j;
bool ok;
QString risp;
messaggio.append(message);
messaggio=messaggio.toHex();
lung=messaggio.size();
QString ans;
//ui.textBrowser->append(messaggio);
area = messaggio.mid(0,2);
term=messaggio.mid(10,2);
if(!term.contains("ef")){
reset();
}
else if(area.contains("ff")){
reset();
}
else if(area.contains("fe")){
x=messaggio.mid(2,4);
y=messaggio.mid(6,4);
ulong x_dec=x.toULong(&ok,16);
ulong y_dec=y.toULong(&ok,16);
x_dec=x_dec;
y_dec=y_dec;
QString x_str= QString::number(x_dec,10);
QString y_str= QString::number(y_dec,10);
ui.textBrowser_x->setText(x_str);
ui.textBrowser_y->setText(y_str);
}
else{
ID =messaggio.mid(2,4);
maschera=messaggio.mid(6,4);
database->leggiarea(area,ID);
ulong maschera_hex=maschera.toULong(&ok,16);
if(maschera_hex==0){
//ui.textBrowser_oggetti->setStyleSheet("background-color: green");
aut_oggetti=TRUE;
```

```
}
else{
aut_oggetti=FALSE;
//ui.textBrowser_oggetti->setStyleSheet("background-color: red");
for(int i=0;i<16;i++){
if(maschera_hex&1){
switch(i){
case 0:
ui.textBrowser_oggetti->append("GIUBBETTO");
break;
case 1:
ui.textBrowser_oggetti->append("MASCHERA");
break;
case 2:
ui.textBrowser_oggetti->append("OCCHIALI");
break;
}
}
maschera_hex=maschera_hex >>1;
}
}
if(aut_oggetti==TRUE && aut_persona==TRUE){
//risp=("01");
//ferma_ricevi();
//this->com->sendCommand(risp, &ans);
//ricevi();
this->com->sendCommand(risp, &ans);
ui.textBrowser_accesso_ok->setStyleSheet("background-color: green");
//ui.textBrowser_accesso->setText("ACCESSO CONSENTITO");

}
else{
//risp=("01");
//ferma_ricevi();
//this->com->sendCommand(risp, &ans);
//ricevi();
ui.textBrowser_accesso_no->setStyleSheet("background-color: red");
//ui.textBrowser_accesso->setText("ACCESSO NEGATO");
}
}
}

void Virtual_LCD_Debug::addSerialReader()
{
```

```
int i;
bool ok;
CSimpleArray<UINT> ports;
CSimpleArray<CString> sFriendlyNames;
QStringList sl;
QString s;
QStringList baud;

baud.append("1200");
baud.append("2400");
baud.append("4800");
baud.append("9600");
baud.append("14400");
baud.append("38400");
baud.append("57600");
baud.append("115200");


if (CEnumerateSerial::UsingCreateFile(ports))/*, sFriendlyNames))*/
{
for (i=0; i<ports.GetSize(); i++)
{
sl.append(QString("COM%1").arg(ports[i]));
}
}
ui.page_cb_com->addItems(sl);
ui.page_cb_baud->addItems(baud);
}
bool Virtual_LCD_Debug::receive()
{
return true;
}

void Virtual_LCD_Debug::open_serial()
{
QString devicePath;
int port;
int speed;
QString msg;
bool ok;
bool connesso;
devicePath=ui.page_cb_com->currentText();
msg=devicePath.mid(3,1);
port=msg.toInt(&ok,10);
```

```
AMSCommunication::Error err;
msg=ui.page_cb_baud->currentText();
if(msg=="1200")
speed=8;
else if(msg=="2400")
speed=10;
else if(msg=="4800")
speed=11;
else if(msg=="9600")
speed=12;
else if(msg=="14400")
speed=13;
else if(msg=="19200")
speed=14;
else if(msg=="38400")
speed=15;
else if(msg=="57600")
speed=17;
else if(msg=="115200")
speed=19;

com = new UARTQtCommunication(port,speed,&receive);
err = com->hwConnect(devicePath,speed);
if (err!=0)
{
ui.textBrowser->setText("Impossibile aprire porta");
ui.textBrowser_Led->setStyleSheet("background-color: red");
}
else
{
ui.textBrowser_Led->setStyleSheet("background-color: green");
}
}

void Virtual_LCD_Debug::scrivi_nome(QString nome_passato)
{
//ui.textBrowser->setStyleSheet("background-color: green");
ui.textBrowser->setText(nome_passato);
aut_persona=TRUE;
}

void Virtual_LCD_Debug::errore_persona()
{
aut_persona=FALSE;
```

```
//ui.textBrowser->setStyleSheet("background-color: red");
ui.textBrowser->setText("PERSONA NON AUTORIZZATA");
}

void Virtual_LCD_Debug::show(void)
{
QMainWindow::show();

addSerialReader();
}

void Virtual_LCD_Debug::reset(void){
ui.textBrowser_accesso_ok->setStyleSheet("background-color: white");
ui.textBrowser_accesso_no->setStyleSheet("background-color: white");
//ui.textBrowser_accesso->clear();
ui.textBrowser->clear();
ui.textBrowser->setStyleSheet("background-color: white");

ui.textBrowser_oggetti->setStyleSheet("background-color: white");
ui.textBrowser_oggetti->clear();
}

/******************* RECEIVER THREAD*****************************/

Receiver::Receiver(AMSCommunication* com,Virtual_LCD_Debug* creator,
                                  QThread *parent) : QThread(parent)
{
this->com = com;
this->creator = creator;
this->receiving = false;
}

Receiver::~Receiver()
{
}

void Receiver::initReceiver()
{
this->receiving = true;
}

void Receiver::stopReceiver()
{
this->receiving = false;
```

```
}

void Receiver::run()
{
QString answer;
//int openIdx, closeIdx;
QChar comando;
QChar lungh_dati;

while(receiving)
{
// check if we already received something
this->com->sendCommand(NULL, &answer);

// if we received something, try to display it
if(!answer.isEmpty())
{

//openIdx = answer.indexOf(QChar(0x02));
//closeIdx = answer.indexOf(QChar(0x03), openIdx);
//if((openIdx < 0xffffffff) && (closeIdx < 0xffffffff))
//{
// answer = answer.mid(openIdx , closeIdx - openIdx);
emit receivedMessage(answer);
//}
}
answer="";
msleep(10);
}

}
```

Bibliografia

[1] INAIL, Statistiche per la prevenzione, Istituto nazionale per l'assicurazione contro gli infortuni sul lavoro, Centro di i nformazione e di documentazione infortunistica, Roma, 1977 e sgg

[2] Decreto legislativo 81/2008 versione aggiornata a settembre 2010.

[3] AA.VV.La sicurezza sul lavoro - legislazione e giurisprudenza, ed. AmbienteDiritto.it - Tortorici 2001-2010

[4] IEEE, IEEE 802.15.4a-2007, http://standards.ieee.org/getieee802/download/802.15.4a2007.pdf

[5] IEEE, IEEE 802.15.4-2006, http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf

[6] IEEE, IEEE 802.15.4-2003, http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf

[7] IPv6 over Low power WPAN (6lowpan). http://www.ietf.org/dyn/wg/charter/6lowpan-charter.html.

[8] Paolo Talone, Giuseppe Russo, RFID Fondamenti di una tecnologia silenziosamente pervasiva, 2a Edizione Ottobre 2008

[9] Paolo Magrassi, A World Of Smart Objects: The Role Of Auto Identification Technologies. Retrieved 2007-06-24

[10] Luigi Battezzati, Jean-Louis Hygounet, RFID - Identificazione automatica a radiofrequenza; la tecnologia, le applicazioni, i principali trend HOEPLI

[11] Antonio Vellucci, Aspetti tecnici e regolamentari sulluso degli RFID in Italia - Ministero delle Comunicazioni - Direzione Generale Pianificazione e Gestione dello Spettro Radioelettrico, maggio 2006

[12] EPC global standards, http://www.gs1.org/epcglobal.

[13] ISO/IEC 18000-1/6:2004

[14] ISO/IEC 15693-1/3:2000

[15] ISO/IEC 14443-1/4:2001

[16] L. Zhang and Z. Wang, Integration of RFID into Wireless Sensor Networks: Architectures, Opportunities and Challenging Problems, Proc. 5th Intl. Conf. Grid Coop. Comput. Wksps. (GCCW06), 2006;

[17] A.G. Ruzzelli, R. Jurdak, and G.M.P. OHare, On the RFID wake-up impulse for multi-hop sensor networks, In Proceedings of 1st ACM Workshop on Convergence of RFID andWireless Sensor Networks and their Applications (SenseID) at the Fifth ACM Conference on Embedded Networked Sensor Systems (ACM SenSys 2007), Sydney, Australia, November, 2007;

[18] Chao Chen, Design of a Child Localization Systemon RFID andWireless Sensor Networks, Journal of Sensors Volume 2010, Article ID 450392, February 2010

[19] Daniel Patrick Pereira, Wanderson Roger Azevedo Dias, Model to integration of RFID intoWireless Sensor Network for tracking and monitoring animals, IEEE International Conference on Computational Science and Engineering, November 2008

[20] Bolivar Torres, Qing Pang, Gordon W. Skelton, Integration of an RFID reader to a Wireless Sensor Network and its use to identify an individual carrying RFID tags, International Journal of Ad hoc, Sensor and Ubiquitous Computing (IJASUC) Vol.1, No.4, December 2010

[21] Ben Ammar Hatem, Hamam Habib, Bus Management System Using RFID In WSN, European and Mediterranean Conference on Information Systems 2010

[22] Po Yu Chen, Wen Tseun Chen, Cheng Han Wu. A Group Tour Guide System with RFIDs and Wireless Sensor Networks, IPSN07, Cambridge, Massachusetts, USA ,April 2007.

[23] Taxonomy and Challenges of the Integration of RFID and Wireless Sensor Networks, IEEE Network, November/December 2008

[24] Evidence srl. http://www.evidence.eu.com/.

[25] Eclipse Framework. http://www.eclipse.org/.

[26] Paolo Gai, Giuseppe Lipari, Luca Abeni, Marco di Natale, and Enrico Bini. Archichetture for a portable open source real-time kernel environment. In Proceedings of the Second Real-Time Linux Workshop and Hand's on Real-Time Linux Tutorial November 2000.

[27] Microchip Technology Inc. http://www.microchip.com/.

[28] Sensor ID snc. http://www.sensorid.it/

[29] IDS RFID. http://www.ids-microchip.com/

[30] Giorgio C. Buttazzo. Sistemi in Tempo Reale chapter 12 Sistemi Operativi Esistenti. Third edition, 2006.

[31] Decreto Ministeriale 13 luglio 2007 n. 369