Capire e usare LoRa®

1

di PIER CALDERAN

Conosciamo approfonditamente la tecnologia LoRa/LoRaWAN per l'uso specifico di dispositivi in ambito IoT, appoggiandoci a un hardware che realizza un end-device LoRa. Prima puntata.

Sono già trascorsi cinque anni da quando, sulle pagine della nostra rivista, apparve un articolo in tre puntate che spiegava in modo approfondito la tecnologia LoRa, abbinando la teoria alla pratica mediante l'utilizzo di uno shield per Arduino, da noi progettato ad hoc: si trattava dei fascicoli di ottobre, novembre, dicembre 2015. All'epoca si parlava ancora poco di LoRaWAN e in generale di tecnologie LoRa, quindi ci limitammo a sperimentare con la tecnologia, ma oggi, con la crescita esponenziale dei dispositivi IoT, l'uso della tecnologia LoRaWAN è diventata un must per questo tipo di applicazioni, tanto che abbiamo ritenuto opportuno approfondire la materia dedicandole il corso che inizia qui.

COSA SONO LORA E LORAWAN

Anche se appartenenti a uno stesso concetto di trasmissione/ricezione a lungo raggio (*Long Range*) le due tecnologie si differenziano totalmente per la loro finalità. Procediamo con ordine e capirete il perché.

LoRa è una tecnologia di comunicazione dati, sviluppata e brevettata nel 2008 da Cycleo (Grenoble, Francia) e acquisita dalla Semtech Corporation (USA) nel 2012; dal sito ufficiale della Semtech (*https://semtech.com*), leader mondiale nella produzione di chip LoRa/LoRaWAN, apprendiamo che sono oltre 100 milioni i dispositivi distribuiti a oggi. Questo panorama offre l'idea di come questa tecnologia sarà sempre più determinante per creare il nostro *Smart Planet*.

La tecnologia LoRa rientra nell'ambito delle reti **LPWAN**, ovvero *Low Power Wide Area Network*, quindi nasce per assicurare comunicazioni a lungo raggio ma con consumi molto più bassi di quelli che dispositivi tradizionali consentirebbero; ciò viene ottenuto con un particolare algoritmo di trasmissione. Quando sono collegati a una rete LoRaWAN, i dispositivi LoRa possono ospitare una

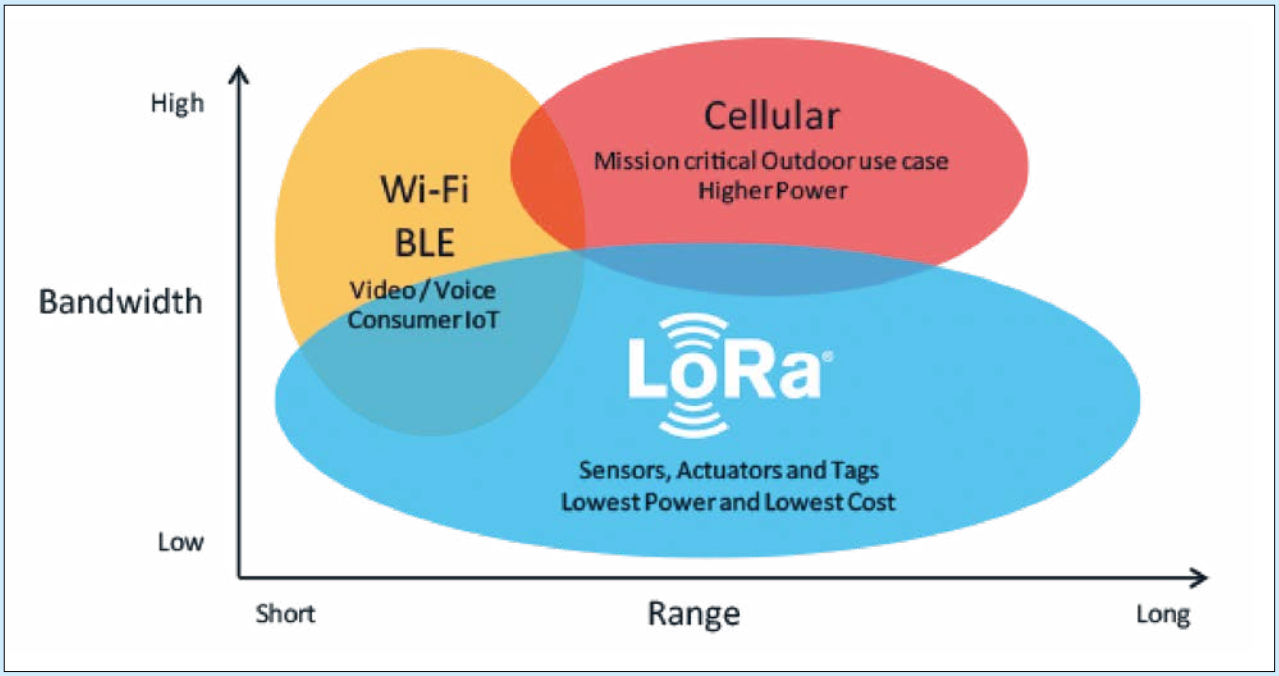


Fig. 1 - L'efficienza della copertura LoRa rispetto ad altre tecnologie.

vasta gamma di applicazioni IoT trasmettendo pacchetti di dati senza bisogno di eccessiva potenza, a differenza delle reti cellulari o basate su Wi-Fi/BLE che richiedono elevata larghezza di banda e alta potenza o che hanno una portata o capacità limitata per superare ostacoli e penetrare in ambienti interni profondi. Lo schema della **Fig.1** (dal sito Semtech) mostra chiaramente come LoRa si colloca rispetto a tali tecnologie.

LoRa ALLIANCE

Fondata nel 2015, LoRa Alliance è un'associazione senza scopo di lucro dedicata alla standardizzazione delle reti geografiche a bassa potenza (LPWAN) e alla promozione globale dello standard aperto LoRaWAN. La missione di LoRa Alliance è quello di supportare e promuovere l'adozione globale dello standard LoRaWAN garantendo l'interoperabilità di tutti i prodotti e tecnologie per un futuro di ecosostenibilità. LoRa Alliance comprende oltre 500 membri di vari settori ed è in rapida crescita. Semtech è un membro sponsor insieme ai leader del settore Alibaba Cloud, Comcast machineQ, Google Cloud, Cisco, IBM e innumerevoli altri che lavorano insieme per dare vita alle soluzioni basate su LoRa. Per fare un esempio di cosa sta succedendo nel mondo, vediamo un caso di copertura LoRa da parte di una compagnia telefonica in Europa. In **Fig. 2** possiamo vedere la copertura LoRa offerta da Swisscom (https://www.swisscom.ch) in Svizzera. Chiamata semplicemente LPN (*Low Power Network*), la rete svizzera per applicazioni IoT copre già oltre il 96,7% del territorio.

FREQUENZE LoRa LIBERE PER TUTTI

Come vedremo, le frequenze LoRa sono libere, ovvero non c'è nessun proprietario che possa imporre una licenza di utilizzo, come succede per altre tecnologie. Chiunque può creare un ecosistema LoRa senza pagare royalty a chicchessia; esattamente come fa Swisscom in Svizzera e altri operatori in tutto il mondo.
Addirittura un privato che voglia costruirsi la sua rete LoRa personale può farlo semplicemente acquistando o producendo in proprio i dispositivi LoRa da collegare a un gateway LoRaWAN per connetterli a un cloud pubblico o privato, e offrire il servizio a chiunque. Vediamo come.

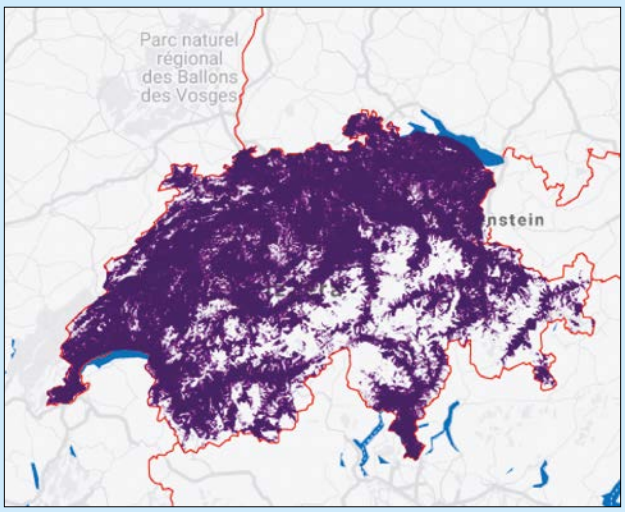


Fig. 2 - La copertura LoRa in Svizzera.

COME FUNZIONA LA TECNOLOGIA LORA

LoRa utilizza bande di frequenza radio sub-gigahertz di libero utilizzo (ovvero senza dover chiedere licenza):

- 169 MHz;
- 433 MHz;
- 868 MHz (Europa);
- 915 MHz (Nord America).

La tecnica di modulazione di LoRA è del tipo CSS (*Chirp Spread Spectrum*), traducibile come "spettro diffuso che utilizza impulsi *chirp*". Questi impulsi chirp, simili a un cinguettio (vedi **Fig.** 3), sono modulati in frequenza lineare a banda larga per codificare le informazioni. È un segnale sinusoidale in aumento o diminuzione della frequenza nel tempo.
Nella **Fig. 3** vedete un esempio di *upchirp*, in cui la frequenza aumenta linearmente nel tempo, e di *downchirp* con la frequenza in diminuzione nel tempo.
Con la modulazione CSS, LoRa consente trasmissioni a lungo raggio da 2 a 5 km in aree urbane con ostacoli e più di 10 km nelle aree rurali, con un basso consumo energetico. Come vedremo, con la sola comunicazione LoRa si può costruire un sistema RF di ricetrasmittenti per un utilizzo "punto-punto", efficiente, economico, ma non connesso o difficilmente connettibile a Internet, a differenza di LoRaWAN che è un protocollo appositamente studiato per l'accesso a Internet.

MODULAZIONE CSS E CHIRP SPREAD SPECTRUM

Cerchiamo di capire meglio come funziona il CSS in ambito LoRa: diciamo subito che mantiene le stesse caratteristiche di bassa potenza della modulazione FSK. In pratica, la tecnica a spettro diffuso che utilizza impulsi chirp modulati in frequenza lineare a larga banda codifica i dati digitali da trasmettere. LoRa usa il CSS per codificare i dati e utilizza un numero di bit che viene diffuso da un certo numero di chirp per bit. Questo numero viene definito fattore di diffusione o *Spread Factor* (SF).
Il CSS LoRa usa fattori di diffusione da 7 a 12:

- valori di diffusione bassi forniscono elevate velocità di trasmissione dati e richiedono meno tempo *Over The Air* (OTA);
- valori alti forniscono basse velocità di trasmissione dati e richiedono un tempo OTA più elevato.

Possiamo così riassumere le caratteristiche della modulazione CSS LoRa:

- larghezza di banda scalabile;
- inviluppo costante in rapporto alla bassa potenza;
- elevata robustezza del segnale;
- resistenza al fading;
- resistenza all'effetto Doppler;
- funzionalità a lungo raggio;
- possibilità di localizzazione

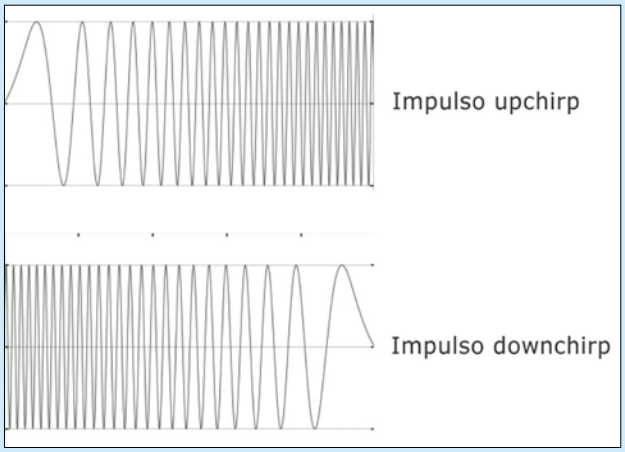


Fig. 3

Per un approfondimento del principio su cui si basa il *LoRa Spread Spectrum* leggete il riquadro asd esso dedicato in queste pagine.

LORAWAN

LoRaWAN definisce il protocollo di comunicazione e un'architettura di sistema per la rete. In pratica il livello fisico LoRa consente il collegamento a lungo raggio, mentre LoRaWAN è responsabile della gestione delle frequenze, della velocità e della gestione dei dati.
I dispositivi nella rete sono asincroni e trasmettono quando dispongono di dati da inviare.
I dati trasmessi da un dispositivo finale LoRa (chiamato *end device* o *end node*) sono ricevuti da uno o più gateway, che inoltrano i pacchetti di dati a un server di rete centralizzato. Il server di rete filtra i pacchetti, esegue i controlli di sicurezza e gestisce la rete. I dati vengono quindi inoltrati a uno o più server delle applicazioni.

ECOSISTEMA LORA/LORAWAN

Un esempio di ecosistema LoRa/LoRaWAN è illustrato in **Fig. 4**.

- Gli *end device* LoRa raccolgono localmente i dati da sensori come, per esempio, temperatura, umidità, luce, acqua, gas, inquinamento, dati GPS e così via.
- Gli *end device* trasmettono i dati a uno o più gateway (detti anche concentratori).
- I gateway ricevono i dati dagli *end device* e li mandano a un cloud Internet via LAN o WLAN.
- Il cloud è connesso a un server di applicazioni.
- Il server delle applicazioni consente la gestione e la visualizzazione dei dati.

DA DOVE INIZIARE

Giunti a questo punto, le strade che possiamo intraprendere per iniziare il nostro cammino nel mondo

LORA SPREAD SPECTRUM

La modulazione LoRa di Semtech affronta tutte le problematiche associate ai sistemi digitali si spettro diffuso per fornire un'alternativa economica, a basso consumo, ma soprattutto robusta, alle tradizionali tecniche di comunicazione a spettro diffuso. Nella modulazione LoRa la diffusione dello spettro si ottiene generando un segnale chirp che varia continuamente in frequenza. Un vantaggio di questo metodo è che gli offset di frequenza e la frequenza tra trasmettitore e ricevitore sono equivalenti, riducendo notevolmente la complessità del design del ricevitore. La larghezza di banda di frequenza di questo chirp equivale alla larghezza di banda spettrale del segnale.
La relazione tra il bit rate dei dati desiderati e la modulazione LoRa può essere espressa come segue e definire il bit rate di modulazione, indicato con Rb nella seguente formula:

$$R_b = SF * \frac{1}{\left[\frac{2^{SF}}{BW}\right]} \; bits/sec$$

dove:
SF = Spreading Factor (da 7... a 12)
BW = Modulation bandwidth (Hz)

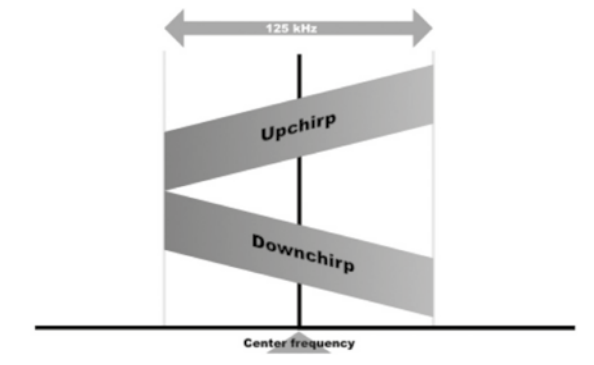
DATA RATE E SPREADING FACTOR

Diamo uno sguardo alla relazione fra la velocità dei dati o *Data Rate* (DR), il fattore di diffusione o *Spread Factor* (SF) e larghezza di banda o *Bandwidth* (BW). Questi tre termini sono correlati, ma il collegamento tra questi non è del tutto ovvio.
Abbiamo visto che LoRa utilizza il cosiddetto protocollo "chirp". Il protocollo chirp utilizza una modulazione di frequenza ad ampiezza fissa. Può utilizzare l'intero spettro allocato per trasmettere segnali producendo un segnale che attraversa il canale. Ci sono due tipi di chirp: il "chirp ascendente" che si muove verso l'alto in frequenza e un "chirp" che si muove verso il basso in frequenza, cioè quando c'è un chirp che si muoverà attraverso la frequenza allocata in entrambe le direzioni.

LARGHEZZA DI BANDA

LoRa utilizza tre larghezze di banda:
- 125 kHz
- 250 kHz
- 500 kHz

Il chirp utilizza l'intera larghezza di banda.



FATTORE DI DIFFUSIONE (SPREAD FACTOR)

I fattori di diffusione sono, in breve, la durata del chirp. Come già detto, LoRa opera con fattori di spread da 7 a 12. SF7 è il più breve tempo in ricetrasmissione, SF12 è il più lungo.
Ogni passaggio nel fattore di diffusione raddoppia il tempo in ricetrasmissione per trasmettere la stessa quantità di dati.
Con la stessa larghezza di banda più tempo in ricetrasmissione ovviamente si ottengono meno dati trasmessi per unità di tempo.

DATA RATE

LoRaWAN usa una diversa configurazione di frequenze, fattori di diffusione e larghezze di banda a seconda del continente in cui opera. Per le bande EU868, EU433, CN780 e AS923 le velocità dati o *Data Rate* (DR) sono quelle riportate in tabella.

DATA RATE	CONFIGURAZIONE	LARGHEZZA DI BANDA	BIT/S	MAX PAYLOAD (CARICO UTILE MASSIMO)
DR0	SF12	125kHz	250	59
DR1	SF11	125kHz	440	59
DR2	SF10	125kHz	980	59
DR3	SF9	125kHz	1760	123
DR4	SF8	125kHz	3125	230
DR5	SF7	125kHz	5470	230
DR6	SF7	250kHz	11000	230
DR7	FSK: 50kpbs	50kpbs	50000	230

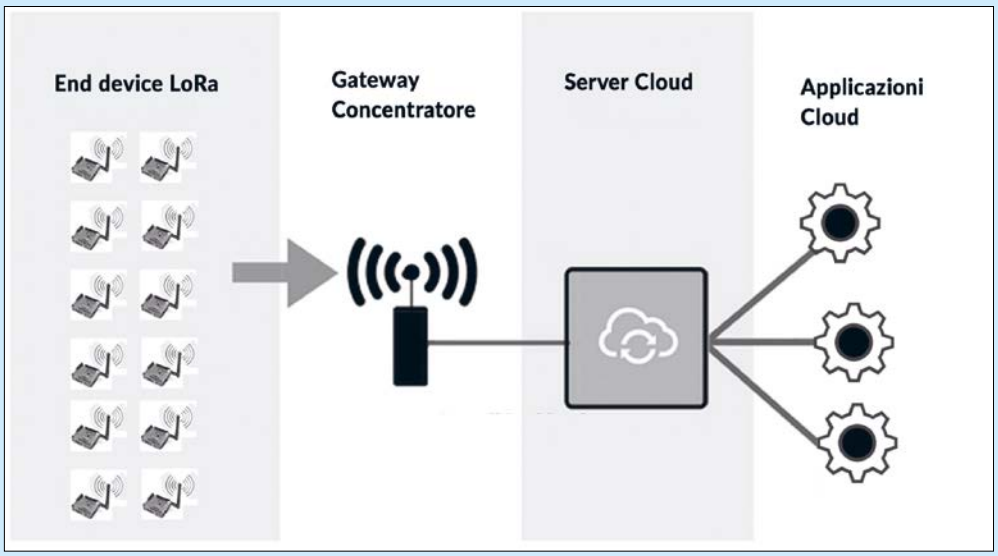


Fig. 4 - Ecosistema Io/LoRaWAN.

LoRaWAN sono essenzialmente due:
1. acquistare un sistema "chiavi in mano";
2. costruirci un sistema LoRa/LoRaWAN.

Confidando sul fatto che chi ci legge sia potenzialmente un maker appassionato del fai-da-te, scartiamo la prima ipotesi a piè pari e cominciamo la nostra avventura per costruire uno o più end device LoRa e anche un gateway LoRaWAN. Cercheremo di creare una rete personalizzata senza costi eccessivi e senza chiedere il permesso di utilizzo delle frequenze a nessuno. Per fare questo sfrutteremo le piattaforme a noi più consone, ovvero Arduino e Raspberry Pi, e tutte le risorse open source messe a disposizione nell'ambito LoRa.

END DEVICE LORA

Fra i molti chip disponibili in commercio, per il nostro primo progetto, che chiameremo *end device LoRa*, ne utilizzeremo uno molto diffuso, ovvero il chip Semtech SX1276, che viene dato già montato e pronto all'uso in molte schede breakout. Per comodità e affidabilità abbiamo posto la nostra attenzione sul breakout RFM95W prodotto dell'azienda Hoperf (*www.hoperf.com*), venduto già per la frequenza di 868 MHz. Il suo aspetto e la piedinatura sono visibili nella **Fig. 5**.
Le caratteristiche principali del modulo Hoperf RFM95W sono di seguito elencate.

- modem LoRa;
- budget di collegamento massimo di 168 dB;
- +20 dBm - uscita RF costante di 100 mW rispetto all'alimentazione;
- PA ad alta efficienza (+14 dBm);
- bit-rate programmabile fino a 300 kbps;
- sensibilità fino a -148 dBm;
- IIP3 di -12,5 dBm;
- corrente in RX di 10,3 mA;
- corrente mantenimento registro di 200 nA;
- sintetizzatore integrato con risoluzione di 61 Hz;
- modulazione LoRa, FSK, GFSK, MSK, GMSK e OOK;
- sincronizzatore di bit integrato per il recupero del clock;
- rilevazione del preambolo;
- gamma dinamica 127 dB RSSI;
- motore di pacchetti fino a 256 byte con CRC;
- sensore di temperatura incorporato e indicatore di batteria scarica;
- dimensioni 16x16 mm.

Il modulo deve essere alimentato ufficialmente a 3,3 V, ma abbiamo testato la tensione di una LiPo da 3,7 V senza problemi e questo si è rivelato ottimo per il nostro progetto. Per funzionare, il modulo ha bisogno di un microcontrollore che comunichi tramite la porta SPI (MISO, MOSI, SCK, NSS) e i sei registri digitali I/O (da DIO0 a DIO5), che possono essere usati per le impostazioni interne.
Il pin di reset serve a inizializzare il modulo mentre al pin ANT va collegata l'antenna tarata sulla lunghezza d'onda di 868 MHz. Per le prime prove, useremo un semplice filo di rame lungo circa 8,6 cm, ovvero ¼ della lunghezza d'onda (34,56 cm/4).

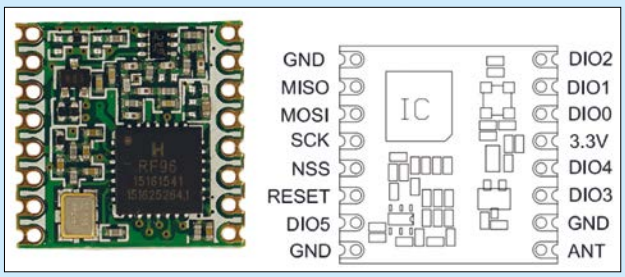
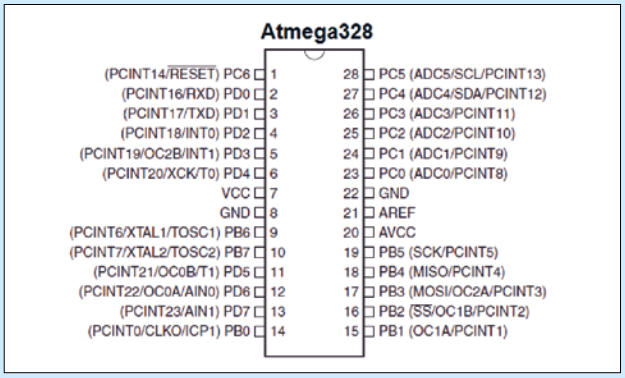


Fig. 5 - Il modulo RFM95W di Hoperf.

IL MICROCONTROLLORE

Per controllare il modulo LoRa abbiamo pensato di utilizzare la piattaforma Arduino o, meglio, il suo microcontrollore, che nello specifico è il chip ATmega328. Il motivo di questa scelta sta nel fatto che, oltre alla sua semplicità di utilizzo, per esso si trovano in rete molte risorse open source dedicate a LoRa e LoRaWAN. Riportiamo la piedinatura e il collegamento con il modulo LoRa nella **Fig. 6**.
Oltre alla connessione alla porta SPI, vanno collegati



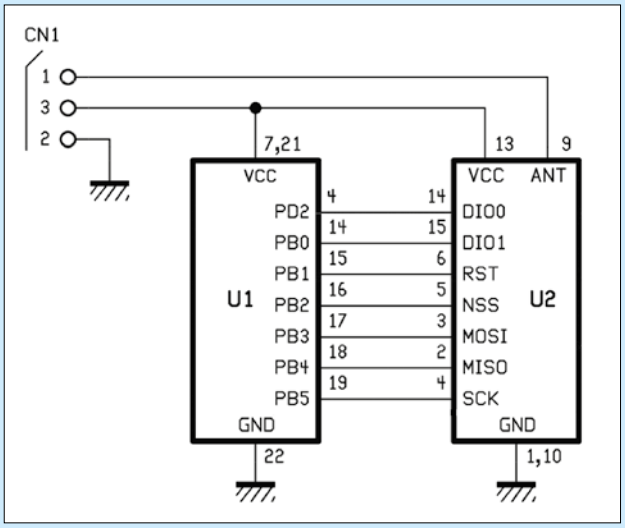


Fig. 6 - Connessione del modulo LoRa al chip ATmega328.

anche i pin DIO0 e DIO1 del modulo che servono per impostare, come vedremo, le funzioni di IRQ callback per la trasmissione e ricezione. Gli altri pin DIO non verranno usati. Notare che nella **Fig. 6** il modulo LoRa risulta capovolto rispetto alla piedinatura serigrafata nella parte inferiore del modulo. Quindi, fate molta attenzione a saldare il componente dalla parte giusta!
Notare che il chip ATmega328 verrà usato senza i consueti condensatori da 22 µF e il quarzo da 16 MHz che di norma andrebbero collegati ai pin 8 e 9 per fornire il clock a 16 MHz. Per il nostro progetto abbiamo preferito usare il chip ATmega328 sfruttando il clock interno a 8 MHz,

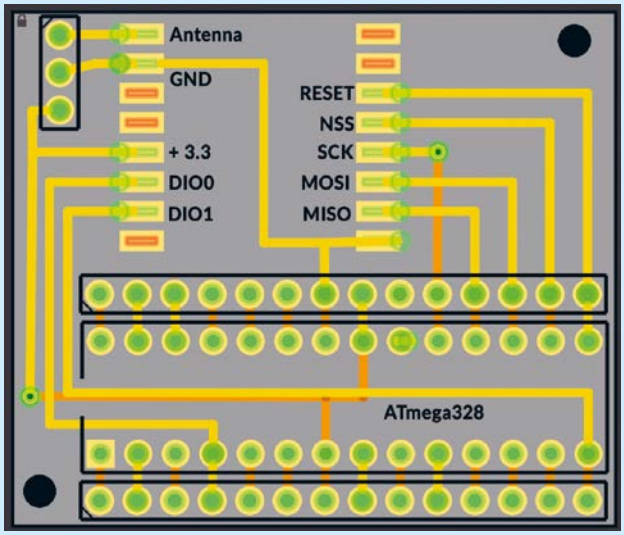


Fig. 7 - Il circuito stampato del nostro end device LoRa.

risparmiando componenti e soprattutto riducendo le dimensioni del circuito stampato a soli 40 x 35 mm.
La **Fig. 7** illustra lo schema del circuito stampato. Per facilitarvi il compito, nell'area download potete scaricare i file Gerber per la stampa del PCB. Per chi vuole modificare il circuito, magari per aggiungere qualche sensore, LED o altro, è disponibile anche il file Fritzing. Coloro che desiderano avere tutto pronto, nello store della Futura Elettronica è disponibile il kit completo dell'end device LoRa (codice prodotto: 8350-SC102), comprensivo di modulo RFM95W, ATmega328, zoccolo 28 pin e i connettori con i pin per l'ATmega328 e l'alimentazione (**Fig.8**).
Ad ogni modo, una volta montati i componenti sul circuito stampato, il nostro end-device sarà simile a quello proposto nella **Fig. 9**. Per l'antenna, basterà saldare uno spezzone di filo di circa 8,6 cm al pin Antenna. Per provare il funzionamento è più che sufficiente. In seguito si potrà

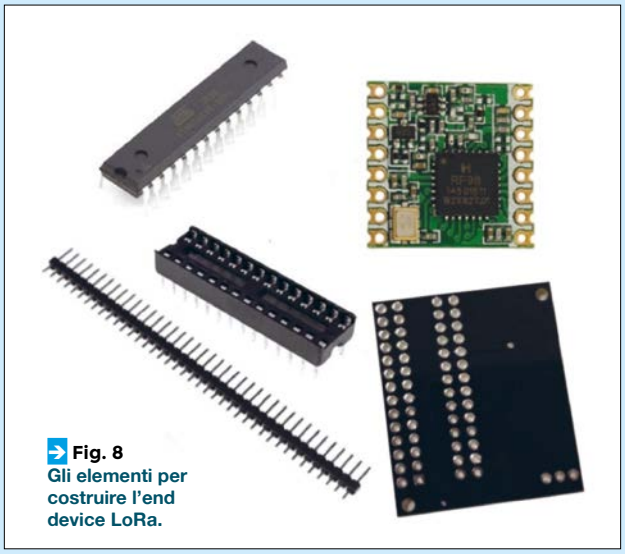


Fig. 8
Gli elementi per costruire l'end device LoRa.

Fig. 9 - Il nostro end device LoRa montato.

sostituire l'antenna a filo con una migliore, per esempio con un'antenna elicoidale, per ottenere un maggior guadagno.

ANALISI DEL CIRCUITO

Per far capire meglio il funzionamento tra microcontrollore e modulo LoRa spieghiamo in dettaglio i collegamenti.
La comunicazione con il ricetrasmettitore LoRa avviene tramite l'interfaccia SPI (*Serial Peripheral Interface*). Questa interfaccia utilizza quattro pin:

- **MOSI**: Master Output Slave Input
- **MISO**: Master Input Slave Output
- **SCK**: Serial Clock
- **SS** o **CS** ovvero Slave Select o Chip Select

La **Tabella 1** espone come devono essere collegati alla porta SPI del chip ATmega328.
La connessione al pin SS è flessibile, ovvero si può dedicare un pin qualsiasi dell'ATmega328, che deve venire configurato via software, come vedremo.

SPI RFM95W	PIN ATMEGA328
SCK	19
MISO	18
MOSI	17
CS/SS	16

Tabella 1

PIN DIO

I pin digital I/O (DIO da 0 a 5) sulla scheda del ricetrasmettitore possono essere configurati per varie funzioni. La libreria che utilizzeremo otterrà informazioni sullo stato istantaneo del ricetrasmettitore, attraverso **DIO0** e **DIO1**.

Ad esempio, quando inizia una trasmissione LoRa, il pin **DIO0** è configurato come uscita **TxDone**.
Quando la trasmissione è completa, il pin **DIO0** viene portato allo stato *high* dal ricetrasmettitore,
Gli altri pin DIO possono essere lasciati scollegati. Sul lato ATmega328, questi due pin del modulo possono essere connessi a qualsiasi pin e vanno configurati via software. In modalità LoRa i pin DIO vengono utilizzati come segue:

- **DIO0**: TxDone e RxDone;
- **DIO1**: RxTimeout.

PIN RESET

Il ricetrasmettitore ha un pin di reset (**RST**) che può essere utilizzato per ripristinarlo in modo esplicito. La libreria utilizza il reset per garantire che il chip LoRa sia in uno stato coerente all'avvio. In pratica, questo pin può essere lasciato disconnesso, dal momento che il ricetrasmettitore si troverà già nello stato normale all'accensione, ma in alcuni casi il collegamento del pin di reset potrebbe evitare problemi di riavvio della connessione. Sul lato ATmega328, è possibile configurare qualsiasi pin per il reset.

RXTX

Il ricetrasmettitore LoRa offre due connessioni di antenna separate: una per **RX** e una per **TX**. Di solito, una scheda ricetrasmittente contiene un chip di commutazione dell'antenna, per commutare le connessioni RX e TX.
In genere, a un commutatore di antenna deve essere detto quale posizione dovrebbe avere, attraverso un pin di ingresso, spesso etichettato come RXTX. Il modo più semplice per controllare il commutatore di antenna consiste nell'utilizzare il pin RXTX del chip SX1276. Questo pin viene automaticamente impostato *high* durante TX e *low* durante RX. Le schede RFM95W sembrano non avere questo tipo di connessione e non espongono alcun pin RXTX, pertanto non è necessario alcun collegamento RXTX all'ATmega328.

LA LIBRERIA LoRa PER INIZIARE

È possibile provare il funzionamento dei nostri end device sfruttando una semplice libreria LoRa, sviluppata da Sandeep Mistry e disponibile per il download dal suo repository GitHub: *https://github.com/sandeepmistry/arduino-LoRa*.
Questa libreria va installata nell'IDE di Arduino come di consueto. Fra gli esempi della libreria possiamo trovare due sketch molto utili al nostro test: **LoRaSender** e **LoRaReceiver**. Il primo serve a mandare un pacchetto LoRa, mentre il secondo serve a riceverlo.

Attenzione! Questa libreria non consente nessun invio di pacchetti LoRaWAN a un cloud, ma effettua una semplice comunicazione peer-to-peer fra due end device. In pratica, fa la stessa cosa di un radiocomando RX/TX per apricancello.
Fra le molte librerie LoRa di questo tipo, abbiamo scelto quella di Sandeep Mistry perché è già configurata per il nostro end device e non serve apportare nessuna modifica o impostare parametri particolari nel file *cpp* della libreria, come accade in altre librerie.
Per il test degli sketch *LoRaSender* e *LoRaReceiver* occorre aver assemblato almeno due end-device.

LO SKETCH LORASENDER

Il codice dello sketch LoRaSender è visibile per intero nel **Listato 1**. Analizzando l'esempio, possiamo vedere che dopo l'inclusione delle librerie SPI e LoRa, nella funzione setup viene inizializzato il modulo LoRa, con la funzione **LoRa.begin(915E6)**. L'unica modifica da apportare al codice è proprio questa: **LoRa.begin(868E6)**.
Questa funzione serve a inizializzare il modulo LoRa alla frequenza europea di 868 MHz, essendo la frequenza di 915 MHz usata per gli USA.
Se non ci sono errori di inizializzazione del modulo, ovvero se non appare il messaggio *Starting LoRa failed!*, lo sketch proseguirà nel loop. Con le funzioni **LoRa.print("hello ")** e **LoRa.print(counter)**, vengono mandati il messaggio di testo *"hello"* e il numero di volte che il messaggio viene trasmesso. La variabile **counter** viene incrementata di 1 ogni 5 secondi.
Con la funzione **LoRa.endPacket()** viene trasmesso il comando di fine trasmissione del pacchetto LoRa. Per una migliore comprensione dell'esempio abbiamo aggiunto i nostri commenti al codice.

LO SKETCH LORARECEIVER

Il codice dello sketch LoRaReceiver è visibile per intero nel **Listato 2**. Come per il precedente sketch, vengono incluse le librerie SPI e LoRa. Nella funzione setup viene inizializzato il modulo LoRa, con la funzione **LoRa.begin(915E6)**. Ricordarsi anche qui di modificare il codice così: **LoRa.begin(868E6)**.
Se non ci sono errori di inizializzazione del modulo, ovvero se non appare il messaggio *Starting LoRa failed!*, lo sketch proseguirà nel loop e, con la funzione **LoRa.parsePacket()**, tenterà di rilevare il pacchetto LoRa inviato dal LoRaSender. Se viene ricevuto un pacchetto con l'istruzione **if (packetSize)**, nel monitor seriale verrà stampato *"Received*

Listato 1

```
#include <SPI.h> //libreria SPI
#include <LoRa.h> //libreria LoRa

int counter = 0;//variabile di conteggio

void setup() {
  Serial.begin(9600); //inizializzazione del monitor seriale
  while (!Serial); //attende che la porta seriale si connetta. Necessario per schede con USB nativo

  Serial.println("LoRa Sender"); //stampa il messaggio "LoRa Sender"

  if (!LoRa.begin(868E6)) {  //Attenzione! Modificare il parametro 915E6 con 868E6
    Serial.println("Starting LoRa failed!"); //stampa "Starting LoRa failed" se il modulo non viene inizializzato
    while (1); //ciclo perpetuo
  }
}

void loop() {
  Serial.print("Sending packet: "); //stampa il messaggio "Sending packet: "
  Serial.println(counter); //stampa il valore della variabile di conteggio counter

  // send packet
  LoRa.beginPacket(); //invia il commando di inizio di trasmissione del pacchetto LoRa
  LoRa.print("hello "); //stampa il messaggio "hello " nel pacchetto LoRa
  LoRa.print(counter); // stampa il valore della variabile di conteggio nel pacchetto LoRa
  LoRa.endPacket();//invia il commando di fine trasmissione del pacchetto LoRa

  counter++; //incremento della variabile di conteggio

  delay(5000); //ritardo di 5 secondi prima di un nuovo invio
}
```

Listato 2

```
#include <SPI.h> //libreria SPI
#include <LoRa.h> //libreria LoRa

void setup() {
  Serial.begin(9600); //inizializzazione del monitor seriale
  while (!Serial); //attende che la porta seriale si connetta. Necessario per schede con USB nativo

  Serial.println("LoRa Receiver"); //stampa il messaggio "LoRa Receiver"

  if (!LoRa.begin(868E6)) {
    Serial.println("Starting LoRa failed!"); //stampa "Starting LoRa failed" se il modulo non viene inizializzato
    while (1); //ciclo perpetuo
  }
}

void loop() {
  // try to parse packet
  int packetSize = LoRa.parsePacket(); //cerca di rilevare un pacchetto LoRa
  if (packetSize) { //se viene ricevuto un pacchetto LoRa…
    // received a packet
    Serial.print("Received packet '"); //…stampa "Received packet"

    // read packet
    while (LoRa.available()) { //mentre è disponibile la connessione LoRa…
      Serial.print((char)LoRa.read());//… stampa il contenuto del pacchetto
    }

    // print RSSI of packet
    Serial.print(" with RSSI "); //stampa " with RSSI "
    Serial.println(LoRa.packetRssi()); //… e il valore in dB del segnale (RSSI)
  }
}
```

packet" e, fintantoché rimane disponibile la connessione nel ciclo **while (LoRa.available())**,verranno letti i dati ricevuti tramite la funzione **LoRa.read()**. Nel monitor seriale verrà stampato il messaggio *"hello "*, il numero di conteggio della variabile counter, seguito dal messaggio *" with RSSI "* e il dato in decibel della forza del segnale (RSSI).

CARICARE GLI SKETCH NEGLI END DEVICE

Vediamo adesso come caricare lo sketch Arduino nei microcontrollori degli end device; per fare ciò non si può operare come con una scheda Arduino, perché il microcontrollore ATmega328 non è un device di quelli contemplati dall'IDE, però esiste la possibilità di programmare il micro stand-alone. Allo scopo è necessario disporre di una scheda Arduino UNO (o simile), impostarla come programmatore ISP e collegarla al chip ATmega328 attraverso la porta SPI.

Chi non ha dimestichezza con questa procedura può seguire il tutorial sul sito Arduino, alla pagina https://www.arduino.cc/en/Tutorial/ArduinoToBreadboard. Nella sezione *Minimal Circuit (Eliminating the External Clock)* viene spiegato in dettaglio come scaricare e installare la scheda "ATmega328 on a breadboard (8 MHz internal clock)". Lo schema di collegamento di Arduino UNO al chip ATmega328 sulla breadboard è visibile in **Fig. 10**.

Una volta fatto questo, è possibile caricare il bootloader per la scheda ATmega328 a 8MHz e gli sketch tramite il menu *Sketch > Carica tramite un programmatore.*

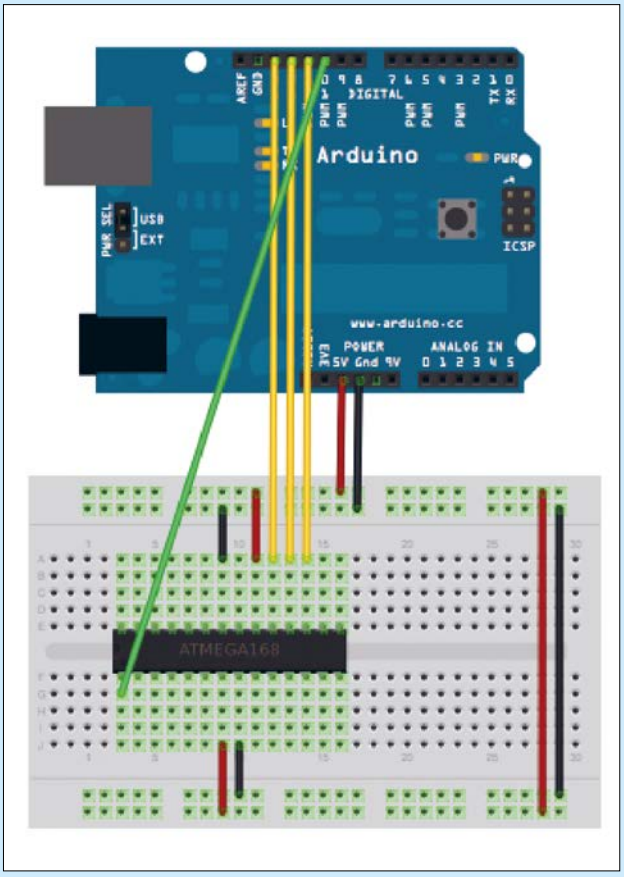


Fig. 10 - Il collegamento ISP di Arduino UNO e ATmega328.

Una volta caricati gli sketch, è sufficiente smontare gli ATmega328 dalla breadboard e introdurli negli zoccoli dei due end device *LoraSender* e *LoRaReceiver*.
Con questa operazione gli end device sono pronti per essere utilizzati nelle nostre applicazioni.

MONITOR SERIALE

Per poter visualizzare i dati in arrivo al nostro *LoRaReceiver*, è necessario collegare la porta seriale del chip ATmega328 alla porta seriale di Arduino UNO, come illustrato nella **Fig. 11**. Basta collegare il TX (pin 3) dell'ATmega328 al pin RX (pin 0) di Arduino UNO, oltre ovviamente, alla massa.
Quindi bisogna caricare un semplice sketch nella scheda Arduino per poter leggere i dati provenienti dalla porta seriale e stamparli sul monitor dell'IDE di Arduino.
Lo sketch da caricare nell'Arduino UNO affinché siano visualizzate le informazioni è illustrato qui di seguito:

```
void setup() {
Serial.begin(9600);
}

void loop() {
if Serial.available()
      {
char in_byte = Serial.read();
Serial.print(in_byte);
      }
}
```

Come vedete, si tratta di un codice molto semplice ed essenziale, eseguendo il quale, se non ci sono errori in trasmissione e/o ricezione degli end device, si otterrà

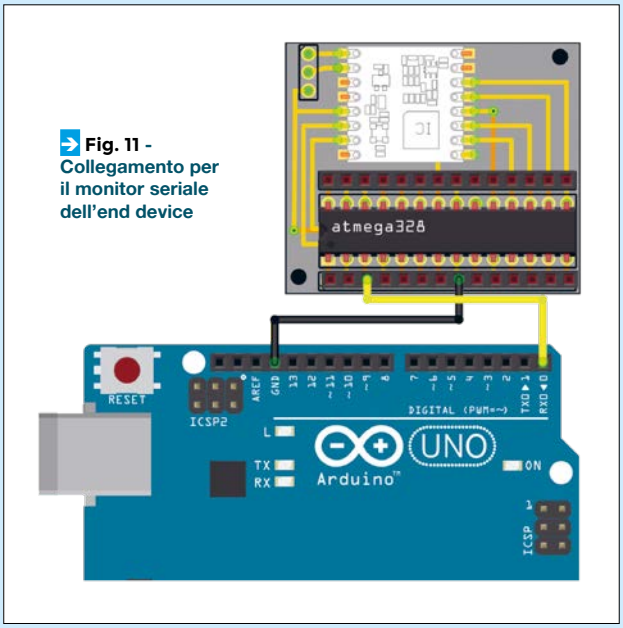


Fig. 11 - Collegamento per il monitor seriale dell'end device

qualcosa di simile a quanto mostrato nella **Fig. 12**, la quale riporta la relativa schermata del Serial Monitor. Se necessario, per verificare i dati inviati dal *LoRasender* basta collegarlo allo stesso modo alla porta seriale della scheda Arduino UNO.

CONCLUSIONI

Bene, per il momento ci fermiamo qui. Con la speranza di essere stati sufficientemente chiari, vi diamo appuntamento alla prossima puntata. Parleremo più a fondo dei pacchetti LoRa e di come usarli con i nostri end device in una rete LoRaWAN. A presto!

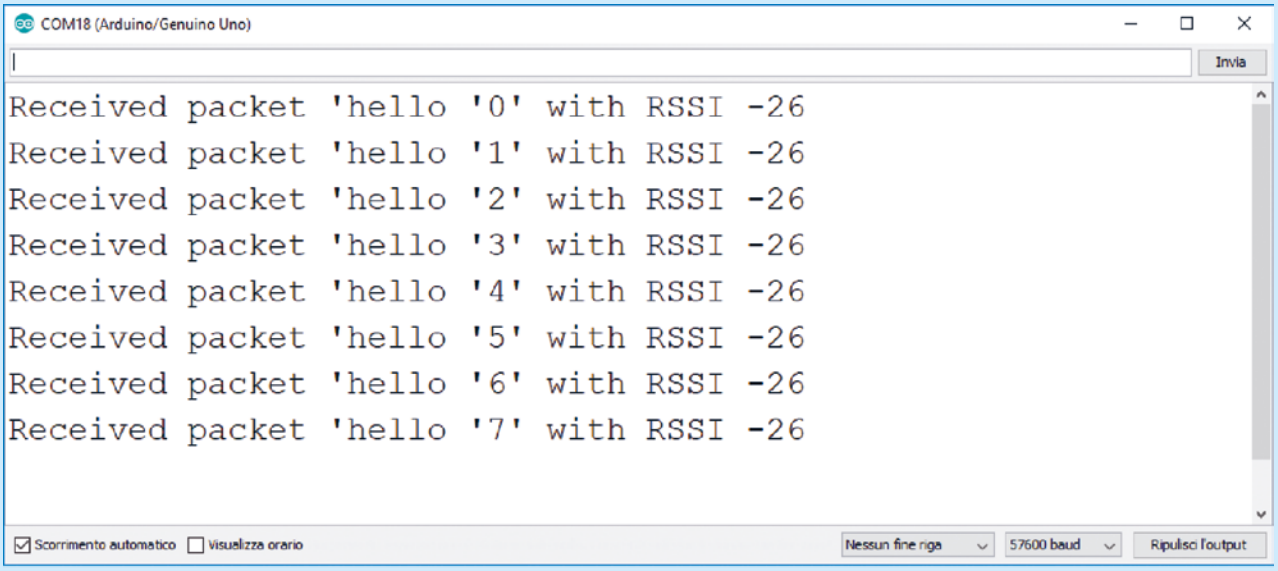


Fig. 12 - Il monitor seriale del receiver.