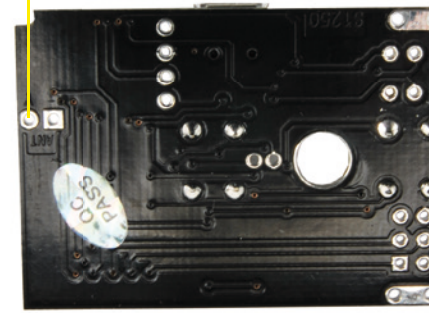
RADIOCOMANDO LoRa *(cod. FT1250M)*

Trasmettitore palmare basato sul modulo radio LoRa SX1278 e gestito dal microcotrollore ATmega32u4. Permette di realizzare comandi via radio long-range. Il trasmettitore viene alimentato tramite una batteria a 12V tipo A23 con un consumo medio di circa 20 mA durante l'attività.

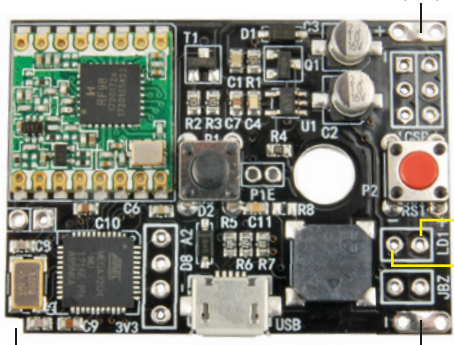
Il telecomando LoRa

Il telecomando viene fornito con il bootloader già caricato e con la possibilità di poterlo programmare via USB con l'IDE Arduino. Gli unici componenti da saldare sono le linguette del polo positivo e negativo della batteria, i fili che vanno ad alimentare il LED presente sul contenitore del telecomando e il filo che funge da antenna. Nelle foto qui sotto sono indicati i punti dove vanno collocati gli elementi da saldare. Il filo che funge da antenna deve essere lungo 17 cm (1/4 d'onda) e va arrotolato con cura nella la parte inferiore del PCB, fissandolo con del nasto adesivo al contenitore. La libreria è scaricabile

ANTENNA (FILO ISOLATO DA 17 cm)



LINGUETTA POLO POSITIVO (+) BATTERIA (12V)



POLO POSITIVO (+) LED

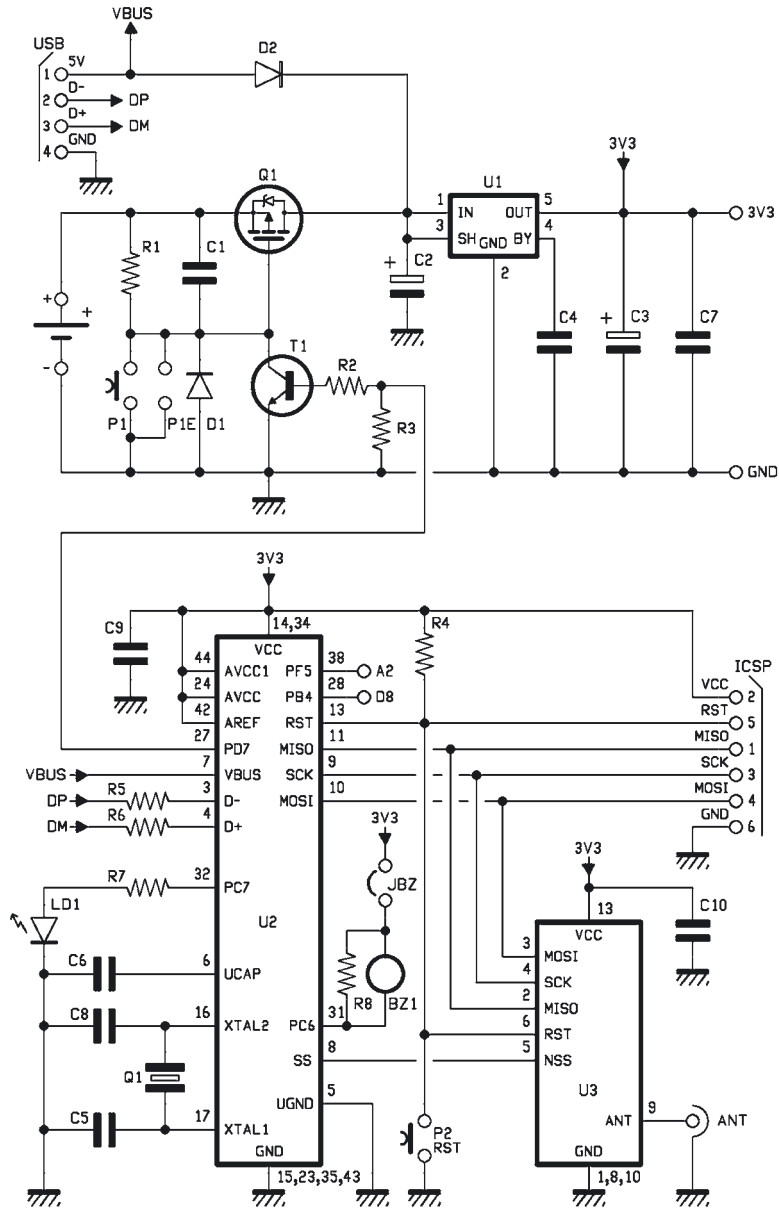
POLO NEGATIVO (-) LED

IL PCB IN QUESTO PUNTO È SAGOMATO PER PERMETTERE IL PASSAGGIO DEI FILI ISOLATI CHE VANNO COLLEGATI AL LED PRESENTE SUL CONTENITORE DEL TELECOMANDO.

LINGUETTA POLO NEGATIVO (-) BATTERIA

ATTENZIONE I FILI CHE VANNO AD ALIMENTARE IL LED E QUELLO CHE FUNGE DA ANTENNA DEVONO ESSERE ISOLATI.

Schema elettrico ricevitore



dalla scheda on-line del prodotto.
È possibile costruire la propria rete secondo necessità. Con la libreria è possibile gestire facilmente due stazioni che si parlano, oppure un centro di raccolta e una moltitudine di sensori sparsi, o altri tipi di strutture di rete più complicate. La libreria può essere utilizzata con funzioni base senza nessun indirizzamento, spedendo via etere un payload giusto per verificare la copertura. Un esempio lo troviamo negli sketch "LoRaTxEcho" e "LoRaRxEcho" presenti negli esempi allegati. In questo caso vediamo le funzioni fondamentali:

- inclusione della libreria
  #include <LoRa.h>
- istanza della classe
  LORA LR;
- inizializzazione
  LR.begin();
- setting dei parametri principali
  LR.setConfig(SF,BW,CR); //tipicamente 12,7,4;
- spedisce messaggio
  LR.sendMess(tbuffer);
- si pone in ricezione
  LR.receiveMessMode();
- riceve messaggio (loop n volte)
  if (LR.dataRead(rbuffer,maxlength) >0) break;

Ma la libreria è organizzata con funzioni più sofisticate così da definire ed utilizzare indirizzi di apparati. In particolare è possibile definire un indirizzo di un apparato nell'ambito di una rete identificata da un indirizzo complessivo. Ovvero è possibile utilizzare un indirizzamento del tipo:

Indirizzo di rete – Indirizzo locale (di device)
La dimensione di indirizzamento della rete è legata alla dimensione dell'indirizzamento locale. Infatti l'indirizzo complessivo definito dalla libreria è sempre basato su 16 bit (unsigned integer per Arduino). Per cui se per esempio si decide di avere uno spazio indirizzabile di 2^8 =256 device, per la rete rimangono $2^{(16-8=16)}$=256 possibili valori.
In realtà la dimensione dell'indirizzo locale è minore di una unità, perché l'indirizzo 0 identifica una trasmissione broadcast (cioè a tutti i locali). Per definire questa suddivisione si utilizza la funzione:

LR.defDevRange(cod);

dove "cod" è uguale all'esponente di 2; per esempio 3 per il range 1-7.
La funzione ritorna il numero massimo di indirizzo di rete per un eventuale controllo.
Ecco alcuni esempi:
Codice 3 : Indirizzo locale 1-7
→ indirizzi di rete da 0 a 8191;
Codice 4 : Indirizzo locale 1-15
→ indirizzi di rete da 0 a 4095;
Codice 5 : Indirizzo locale 1-31
→ indirizzi di rete da 0 a 2047;
Codice 6 : Indirizzo locale 1-63
→ indirizzi di rete da 0 a 1023;
Codice 7 : Indirizzo locale 1-127
→ indirizzi di rete da 0 a 511;
Codice 13 : Indirizzo locale 1-8191
→ indirizzi di rete da 0 a 7.

A questo punto non rimane che decidere l'indirizzo della rete che deve essere compreso entro il limite stabilito dalla precedente suddivisione.
Questo può essere fatto dalla funzione:

LR.defNetAddress(addr);

Ritorna errore se "addr" non è dentro i limiti.
Poiché le portate radio sono consistenti

è necessario proteggere le informazioni; non a caso il protocollo LoRaWAN prevede una crittografia AES128.
Anche la libreria da noi proposta implementa una crittografia AES (ma AES256). È quindi necessario stabilire una identica chiave tra gli apparati.
Questo può essere fatto tramite la stessa funzione di inizializzazione, passandogli un intero come chiave:

LR.begin(key);

Poiché la chiave AES è formata da 32 Byte, in realtà il numero "key" serve come "seme" per inizializzare il generatore di numeri random; la funzione stessa si occupa poi di generare la chiave di 32 Byte. In questo modo è possibile scambiare un semplice numero come chiave di comunicazione.
Ciò si mantiene vero solo nel caso che si usi il compilatore di Arduino per tutti i device. Infatti a partire dal "seme" la routine del compilatore produrrà la stessa sequenza pseudo-randomica in tutti i device Arduino.
Le funzioni principali da utilizzare in una rete LoRa proprietaria possono allora essere elencate così:

- inclusione della libreria:
  #include <LoRa.h>
- istanza della classe:
  LORA LR;
- inizializzazione:
  LR.begin(key);
- setting dei parametri principali:
  LR.setConfig(SF,BW,CR);
  //tipicamente 12,7,4
- definisce il range di ind. locale:
  LR.defDevRange(cod);
- definisce l'indirizzo di rete:
  LR.defNetAddress(address);
- spedisce il messaggio:
  LR.sendNetMess(destinatario locale, mittente locale, mess, lung);
- si pone in ricezione:
  LR.receiveMessMode();
- riceve messaggio
  (loop n volte):
  if (LR.receiveNetMess(dest. locale,mitt. locale, buffer, maxlung)>0) break;
- estrae e decodifica il testo:
  LR.getMessage();
- estrae e dec. l'indirizzo mittente:
  LR.getSender();

La funzione di ricezione torna 0 se non sono arrivati messaggi o non sono arrivati messaggi inviati al destinatario. Inoltre ritorna -1 se il messaggio non è stato spedito dal mittente indicato. Se nel messaggio il destinatario ha valore 0 il messaggio è sempre accettato (broadcasting). Se il parametro mittente della funzione ha valore 0, vengono accettati i messaggi a prescindere dal mittente.
Negli esempi inclusi nella libreria è possibile trovare due sketch "LoraTxAddressing" e "LoraRxAddressing" che realizzano un server che dialoga con diversi moduli periferici.
Tenete comunque presente che la gestione diretta dei registri dello SX1278 può sempre essere fatta dalle funzioni base della classe SX della libreria.

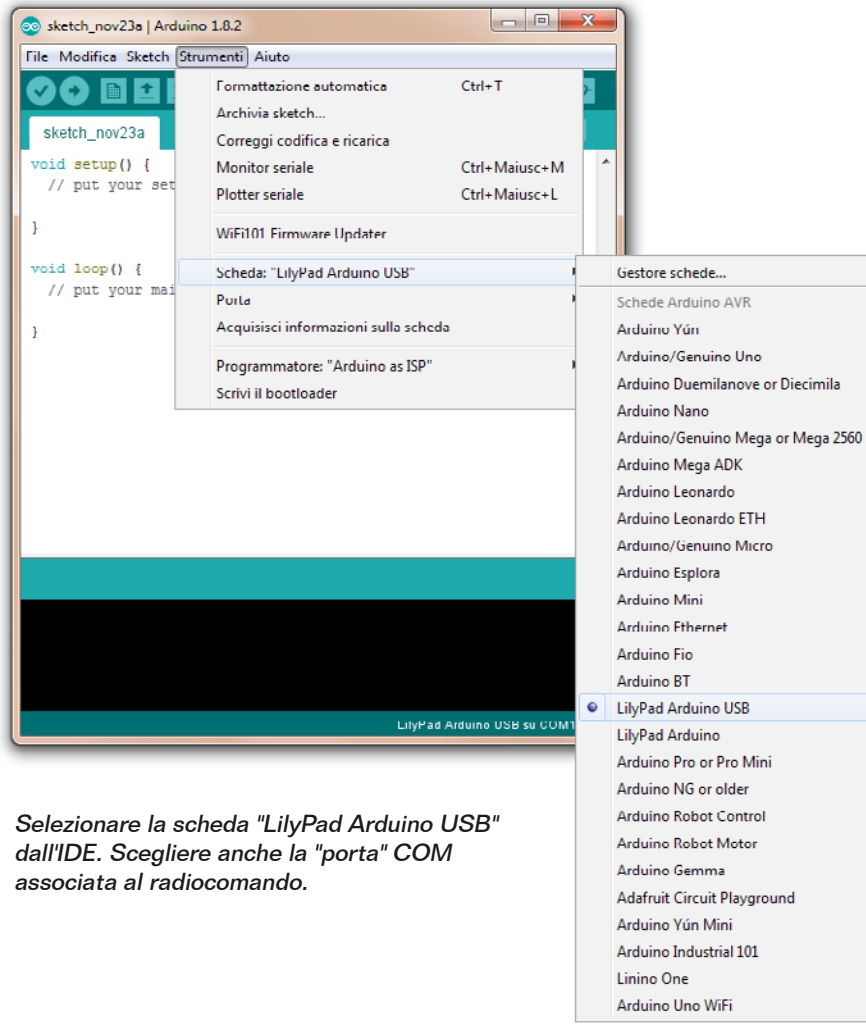
Il telecomando e la libreria

Il telecomando contiene un Arduino compatibile come processore e velocità di clock con la versione "LilyPad con USB". Il telecomando, però, invia un messaggio sempre uguale alla pressione del tasto. Ne risulta una trasmissione sempre identica (anche se criptata).
Chiunque, se dotato di opportuna apparecchiatura, potrebbe registrare e ripetere la trasmissione senza bisogno di decodificarla. È il problema di tutti i telecomandi attuatori. Problema che si cerca di risolve-

re, per esempio, rendendo la trasmissione sempre diversa con la tecnica del "rolling code". Per risolvere questo problema nella libreria è stato predisposto un "marker", ovvero un byte con valore casuale. Il marker di un messaggio ricevuto, può essere letto e confrontato con gli ultimi marker arrivati. Nel caso fosse presente in questa lista, il messaggio sarebbe rigettato come sospetto.

La grandezza della lista sarà decisa a piacere. Non troppo lunga per non avere troppe probabilità di trovare un marker già utilizzato.

Né troppo corta per avere un controllo su un sufficiente numero di trasmissioni



Selezionare la scheda "LilyPad Arduino USB" dall'IDE. Scegliere anche la "porta" COM associata al radiocomando.

già effettuate. Il telecomando si accende ogni qual volta viene pigiato il pulsante di comando. In questa situazione non possiamo usare il generatore random di Arduino che sarebbe inizializzato ogni volta allo stesso valore. Per avere un valore random ad ogni accensione, dobbiamo ricorrere al rumore elettronico. Infatti possiamo leggere il valore di una o più porte analogiche di Arduino, non utilizzate e lasciate ad alta impedenza. Questo valore può essere usato come "seme" per inizializzare il generatore random (o meglio pseudo-random). Per leggere il marker si usa la funzione:

LR.getMarker();

Si tenga presente che il messaggio risulta crittografato nel suo complesso, eccetto il destinatario. Quindi, anche a causa della modalità operativa della crittografia AES, il contenuto, compreso il marker, si trova distribuito come rumore su tutta la lunghezza del messaggio, rendendo la sicurezza molto efficace.

Configurazione IDE di Arduino

Il telecomando LoRa comprende sia il microprocessore Arduino che il modulo radio. È stato scelto il processore Atmega32u4 con quarzo a 8 MHz, lo stesso della scheda LilyPad con USB. Il quarzo da 8 MHz al posto di quello a 16 MHz di Arduino standard, è necessario a causa dell'alimentazione ridotta a 3,3V. In ogni caso, poiché è la stessa configurazione di LilyPad con USB, è possibile (e necessario) selezionare questa piattaforma dall'elenco dell'IDE di programmazione. La programmazione ha sempre le stesse caratteristiche di quella destinata ad "Arduino Uno", in quanto è l'IDE che si preoccupa di tenere conto della diversa velocità di clock in fase di compilazione. Per la programmazione è stata prevista sul telecomando una presa micro USB. Il trasmettitore viene alimentato dalla pressione del pulsante di comando; questo significa che il processore deve essere veloce nell'intervenire sull'interruttore elettronico, in modo da mantenere l'alimentazione al rilascio del pulsante. Infatti questa è la

Il trasmettitore nel suo contenitore

prima cosa che fa, senza problemi visti i tempi di intervento umano che hanno durate di alcune decine di millisecondi. Il processore utilizza il pin D6 per portare a massa il gate del POWERMOS Q1 (a canale P) tramite il transistor T1. Il transistor T1 agisce in parallelo al pulsante e quindi mantiene in conduzione il POWERMOS Q1 anche quando il pulsante viene rilasciato. Una volta che ha provveduto a mantenere l'alimentazione, il processore inizializza il modulo radio con le caratteristiche LoRa definite dallo sketch e con l'indirizzo di rete e quelli locali. Inoltre inizializza il generatore di numeri pseudo-random, del compilatore C di Arduino, con un paio di valori letti dai pin analogici non utilizzati (A0 e A1) e moltiplicati fra loro. In questo modo utilizza un rumore elettronico variabile ad ogni accensione. Quest'attività è necessaria per produrre un marker casuale del messaggio. A questo punto spedisce il messaggio con il pattern stabilito ed attende la risposta. Quando riceve una risposta positiva, segnala l'avvenuta esecuzione con un codice bip convenuto, tramite il cicalino. Altrimenti dopo una certa attesa (qualche secondo), oppure dopo aver ricevuto una risposta negativa segnala con un diverso codice di bip. Quindi si spegne, ovvero pone basso il pin D6. Nel Listato1 è mostrato lo sketch preparato come esempio per il telecomando LoRa.

In questo modo il processore ed il modulo radio sono alimentati solo per pochi secondi (con un consumo medio di circa 20 mA durante l'attività). Una semplice batteria del tipo A23 può quindi farlo funzionare per parecchio tempo. Il server che riceve e gestisce il comando può essere implementato come riportato nello sketch di esempio presente nella libreria. Il server che riceve e gestisce il comando non deve far altro che mettersi in ascolto della trasmissione da parte del telecomando, verificare il pattern concordato e verificare che il marker non sia presente nella lista di quelli già arrivati. Nel caso sia un duplicato, scarta il messaggio e risponde negativamente. Nel caso, invece, sia originale, lo inserisce in una posizione casuale della lista e manda la conferma dopo aver eseguito l'azione. L'inserimento del marker nella lista in una posizione casuale, sovrascrivendo quello eventualmente presente, serve a evitare una qualunque periodicità.

Naturalmente è sempre presente la possibilità di ricevere un messaggio con un marker nella lista anche se originale. Ma poiché la trasmissione è bidirezionale, il server avverte della mancata attivazione e si può semplicemente ripetere il comando.

Uso alternativo dell'hardware del telecomando

Se in parallelo al pulsante colleghiamo un contatto pulito (sfruttando le piazzole P1E), come ad esempio un contatto reed di quelli per porte e finestre (quello degli antifurto, ndr), realizzeremo un efficiente sensore wireless di allarme anti-intrusione. Per esempio collegandolo ad una porta potremo ricevere sul server, anche molto distante, la segnalazione dell'apertura che potrà essere utilizzata come allarme o come registrazione di evento. Infatti lo shield LoRa può essere montato su RandA (la nostra shield per Raspberry Pi che integra il core di Arduino UNO) dove il processore Raspberry Pi può agire come un sofisticato server, eventualmente collegato in Internet.

La miniaturizzazione del telecomando e la sua auto-alimentazione lo predispongono ad essere posizionato ovunque sia necessario un sensore attivato meccanicamente. Se poi aggiungessimo un piccolo circuito che realizzi un timer a bassissimo consumo (di pochi nanoampere), potrem-

mo azionare elettronicamente il pulsante per esempio tramite un transistor MOS che agisca come un interruttore. In questo caso potremmo periodicamente spedire al server un valore analogico letto sul pin A2 o un valore digitale sul pin D8.

L'articolo completo del progetto è pubblicato su Elettronica In n. 217

A tutti i residenti nell'Unione Europea
Importanti informazioni ambientali relative a questo prodotto

Questo simbolo riportato sul prodotto o sull'imballaggio, indica che è vietato smaltire il prodotto nell'ambiente al termine del suo ciclo vitale in quanto può essere nocivo per l'ambiente stesso. Non smaltire il prodotto (o le pile, se utilizzate) come rifiuto urbano indifferenziato; dovrebbe essere smaltito da un'impresa specializzata nel riciclaggio. Per informazioni più dettagliate circa il riciclaggio di questo prodotto, contattare l'ufficio comunale, il servizio locale di smaltimento rifiuti oppure il negozio presso il quale è stato effettuato l'acquisto.

Distribuito da:
FUTURA GROUP SRL
Via Adige, 11 - 21013 Gallarate (VA) Tel. 0331-799775 Fax. 0331-792287
web site: www.futurashop.it
supporto tecnico: www.futurashop.it/Assistenza-Tecnica