 # Intro: che cos'è il MIDI?

Si tratta di uno standard tecnico che descrive un protocolllo di comunicazione, una interfaccia digitale e connettori elettrici e permette ad una grande varietà di strumenti musicali elettronici, computer ed altri apparati, di comunicare tra loro.

Il MIDI si è standardizzato negli anni '80 ma certo non è stato il primo sistema di comunicazione tra dispositivi elettronici associati alla musica.

Già dal momento della sua nascita (1964 circa), il sintetizzatore, uno strumento musicale elettronico in grado di generare sonorità particolari, era un dispositivo modulare. Costituito cioè da diversi componenti indipendenti, ognuno predisposto per apportare una particolare lavorazione al segnale elettrico.

Per ottenere un suono i diversi componenti dovevano essere connessi tra loro utilizzando appositi cavi chiamati pathcords.

Il sistema di controllo allora era basato sul voltaggio (control voltage): variazioni di tenzione, propagate lungo i patchcords, permettevano ad un modulo di controllare il modulo successivo e indurre una particolare modificazione al segnale audio di passaggio.

Con il passare del tempo i sintetizzatori, da monofonici, divennero polifoni e in seguito, con l'avvento dell'era digitale, si arricchirono della capacità di memorizzare le impostazioni per i propri controlli: se la sintesi del suono rimaneva analogica, i controlli, ora digitali, potevano essere letti e immagazzinati sulle memorie RAM, da poco entrate sul mercato.

Agli inizi degli anni '80 la trasformazione digitale è ormai completa ma resta ancora una difficoltà mai completamente risolta: come permettere la comunicazione tra strumenti musicali di marche differenti?

Ogni costruttore adottava un standard proprietario, coerente su tutti i prodotti della proprio catalogo, ma incompatibile spesso con i prodotti di altri costruttori.

Nel Giugno del 1981 il fondatore di Roland propone una standardizzazione al fondatore di Oberheim, il quale a sua volta ne discute con il presidende della Sequential Circuit Inc. Nell'ottobre dello stesso anno se ne parla poi con i rappresentati di Yamaha, Korg e Kawai.

Fu così che nel 1982 lo standard entro sul mercato con i primi due sintetizzatori il Roland Jupiter-6 e il Roland Prophet 600. Seguirono a ruota la prima drum machine come la Roland TR-909, sequencer, e computer in grado di supportare lo standard.

Con l'ingresso del MIDI, produrre musica diventò più semplice. Nascevano gli home studio. Si poteva scrivere musica usando il PC (nascono inotation scoring software). Anche il mondo dei videogames si appropriò della tecnica MIDI.

Dati!

Il MIDI utilizza una comunicazione seriale proprio come l'USB (universa serial bus) ora, se il connettore USB è bidirezionale e permette trasmissione e ricezione utilizzando un solo cavo, nel MIDI invece un singolo cavo corrisponde ad un singolo verso di comunicazione.

Connettori MIDI usano DIN a 5 pin anche se solo 3 sono in uso mentri i restanti 2 previsti per successive implementazioni.

Accenni di matematica binaria

Avendo a che fare con il mondo digitale, dobbiamo sapere che l'unità di base per questo tipo di comunicazioni (quindi di conseguenza anche per il MIDI), è il bit.

Gruppi di otto bit costituiscono i Bytes i quali possono anche essere pensati come coppie di raggruppamenti da 4 bit (i così detti nibbles).

Quanti valori possono essere rappresentati con 2 bit? con 4? e con 8?

Esperimento 1 - Obiettivi

* connettere un dispositivo MIDI trasmittente ad Arduino;
* connettere Arduino al PC per monitorare i messaggi in arrivo;

Il u-controllore principale di Arduino UNO non si occupa della gestione della porta USB. C'è invece un altro piccolo u-controllore in prossimità del connettore USB che è preposto a questo scopo. E' stato programmato appositamente per fungere, anche ma non solo, da **UART** (unversal asyncronous serial receiver/transmitter).

Anche se questo non è un u-controllore sul quale noi possiamo intervenire (magari programmandolo in modo differente), si tratta di un dispositivo comunque utilissimo perchè consente di sgravare il u-controllore principale dall'incombenza di restare in ascolto sulla porta seriale o di predisporre tutto il necessario per l'invio di nuovi messaggi sulla stessa.

E' la UART che se ne occupa mentre il u-controllore principale può continuare ad occuparsi del programma principale ed essere semplicemente notificato dalla UART quanto strettamente necessaio.

problema #1: per comunicare con il mondo esterno, Arduino ha bisogno della UART. Sulla scheda Arduino ne abbiamo soltanto una quando invece l'obiettivo ci richiede di stabilire due connessioni indipendenti: una con il PC e una con il dispositivo MIDI.

problema #2: non abbiamo a bordo dell'Arduino un connettero DIN a 5 pin.

L'uscita dalla UART (lo si vede bene dagli schematici della scheda) è riportata ai pin digitali 0 e 1 (quelli etichettati come RX e TX rispettivamente). Si potrebbe pensare di utilizzare questi, portandoli in breadboard, per usare poi un connettore DIN 5 pin al posto della presa USB. Ma così facendo si creerebbe un conflitto tra le due comunicazioni.

Fortunatamente Arduino dispone di una libreria (lo vedremo meglio appena ci addentreremo sul lato software) che permette di emulare i comportamenti di una UART così da permettere più d'una comincazione seriale.

Detto questo diamo uno sguardo al circuito.

Il circuito

Nello schema è mostrata sia la sezione imputata all'invio sia quella che si occupa della ricezione dei messaggi. Per ora concentriamoci (l'abbiamo già detto) sulla sezione di ricezione dei messaggi.

Qui abbiamo un paio di componenti interessanti:

* **optoaccoppiatore** (o optoisolatore): gli optoaccoppiatori permettono di mantenere i device MIDI elettricamente disaccoppiati dai connettori in modo da prevenire "ground loops" e proteggerli da "voltage spikes";
* **diodo**: impediscono che la corrente fluisca nel senso opposto;

Hands on

Disponiamo sulla breadboard i nostri componenti e seguiamo i passaggi:

1. optoisolatore;
2. connettore DIN 5 pin;
3. connettiamo l'optoisolatore alle linee di terra e alimentazione (secondo il datasheet del componente);
4. cabliamo il DIN alla porta LED del optoisolatore;
5. disponiamo il diodo di protezione;
6. connettiamo alimentazione e terra di Arduino ai rispettivi rail della breadboard.
7. connettiamo l'uscita del optoisolatore al pin 11 di Arduino;

Il codice

Come dicevamo poco fa la libreria **Software Serial** è stata sviluppata appositamente per permette la comunicazione seriale anche su altri pin di Arduino, usando il software per emulare il comportamento della UART.

Usando SoftwareSerial è così possibile utilizzare più di una porta di comunicazione seriale su diversi pin.

Il **baudrate** da impostare per la comunicazione è **31250**.

Analisi dei messaggi

Status Byte / data Byte

Osservazioni: Note ON & Note Off

Da notare che non sembra di individuare messaggi di Note Off, perchè?
Evidentemente la tastiera MIDI che stiamo utilizzando non è stata progettata per inviarli.
Infatti è diventata prasso quella di utilizzare messaggi di Note On con velocity 0 al posto di messaggi di Note Off (questo permentte di risparmiare un byte).

Osservazioni: Running status

Se esaminiamo meglio la lista di valori sul monitor seriale ci accorgiamo che talvolta, dopo la comparsa di primo messaggio di stato (144), sono elencati molti data bytes senza che lo status si ripresenti come ci si potrebbe aspettare.

Si tratta del così detto Running Status. Lo standard infatti stabilisce che lo status byte sia inviato solo se questo cambia rispetto al precedente.

Anche in questo caso si risparmiano byte preziosi.

Esperimento 2 - Obiettivi

* connettere un dispositivo MIDI ricevente ad Arduino;
* inviare messaggi al dispositivo programmaticamente;

Il circuito

Il circuito in questo caso è più semplice

Il codice

Non suona, perchè?? Il problema è causato dala fatto che stiamo trasmettendo su di un canale sbagliato.

Il MIDI consente infatti di trasmetter messaggi a più dispositivi diversi connessi simultaneamente (o in daisy chain) ad un unico trasmettitore.

discorso dei **canali**: da 1 a 16 (da 0 a 15).

per permetter al sistema di emettere suono possiamo agire in due modi diversi:
* cambiare il canale sui cui il dispositivo ricevente resta in ascolto dei dati MIDI in ingresso;
* oppure cambiare il canale da usare nella trasmissione dei dati lato Arduino;

Esperimento 3: recap.

Il cocice

La libreria Software Serial, pur essendo molto versatile, presenta una serie di limitazioni.

Pur potendo essere utilizzata praticamente sulla maggiorparte di pin dell'arduino UNO, non è in grado di gestire 2 flussi di informazioni simultanei (ricezione ed invio).

Per risolvere il problema possiamo usare la libreria **AltSoftwareSerial**.

La AltSoftwareSerial non è libera dalle proprie idiosincrasie come la necessità di usare 2 soli pin specifici: il 9 per la trasmissione e l'8 per la ricezione.

Proviamo uno sketch dove la nostra tastiera MIDI invia messaggi ad Arduino che poi, a sua volta, li reinoltra alla drum machine.

Libreria **MIDI_library**

componenti necessari

* an **6N138** or a **4n28** optocoupler.
* an **1N4148** high speed diode;
* four **220 Ohm** resistors;
*** and two 5 pin DIN**.

materials & TODO

* scarica il repository in locale
* stampa le configurazioni circuitali
* Arduino UNO + Arduino MEGA di scorta
* sladatore + stagno + supporto
* pinzette varie
* 2x trasformatori (uno per la drum machine + uno per la MIDI keyb)
* libro perotti
* drum machine + manuale
* tastiera Axiom + manuale
* 2x cavi MIDI

MIDI in

```
#include <SoftwareSerial.h>

const byte rxPin = 11;
const byte txPin = 10; // not used for the moment

SoftwareSerial mySerial(rxPin, txPin);

// SETUP ////////////////////////////////////////
void setup()
{
  pinMode( rxPin, INPUT );
  pinMode( txPin, OUTPUT);
  mySerial.begin( 31250 );

  Serial.begin( 9600 );
}

// LOOP /////////////////////////////////////////
void loop()
{
  while( mySerial.available() > 0 )
  {
    unsigned char c = mySerial.read();
    Serial.println( c, DEC );
  }
}
```

MIDI out

```
#include <SoftwareSerial.h>

const byte txPin = 10;
const byte rxPin = 11; // not used for the moment

SoftwareSerial mySerial(rxPin, txPin);

int dly = 1000;
int note, velocity;
int ch = 9;

// SETUP ////////////////////////////////////////
void setup()
{
 pinMode( rxPin, INPUT );
 pinMode( txPin, OUTPUT);
 mySerial.begin( 31250 );
}

// LOOP /////////////////////////////////////////
void loop()
{
 note = random(24)+34;
 velocity = 127;

 // note On
 mySerial.write(144 + (ch-1) );
 mySerial.write(note);
 mySerial.write( velocity );

 delay(dly);

 velocity = 0;

 // note Off
 mySerial.write(144 + (ch-1));
 mySerial.write(note);
 mySerial.write( velocity );

 delay(dly);
}
```