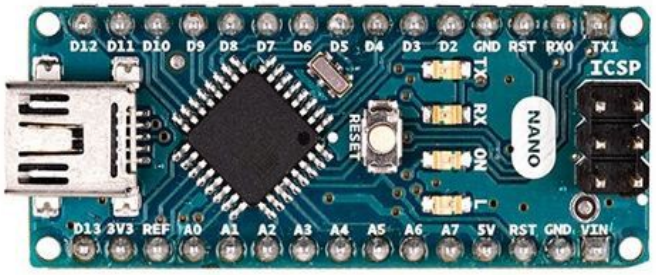
Sistema theremino

Theremino ArduHAL
V1.4
Istruzioni

Principio di funzionamento

Si collega un Arduino Nano alla USB.



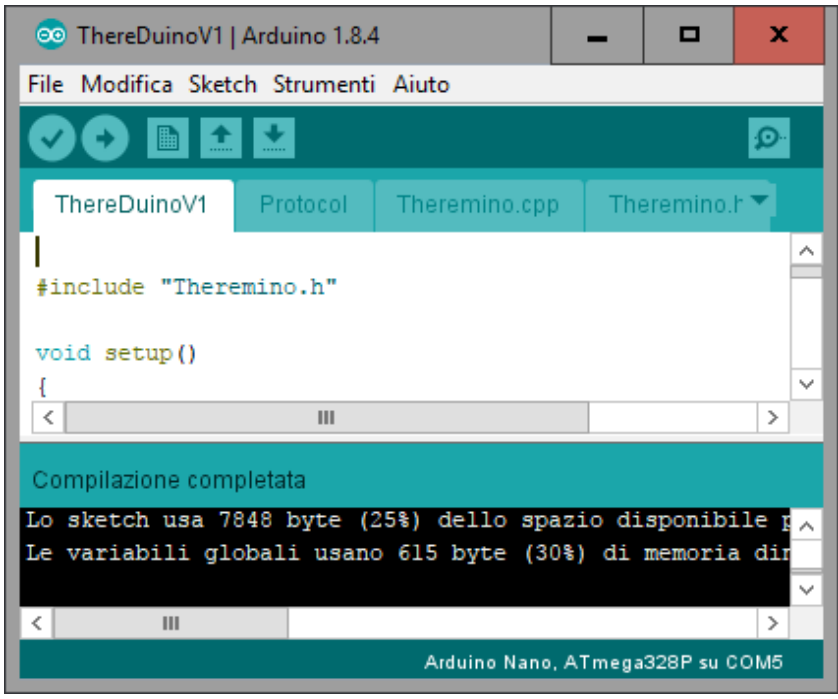
Si possono utilizzare anche altri Arduino ma leggete prima ***questa pagina***.

Si utilizza l'Arduino IDE per programmare l'Arduino con il nostro firmware (come spiegato in ***questa pagina***).

OPPURE

È possibile acquistare un Arduino Nano pre-programmato sul sito ***Store-ino*** o su eBay.

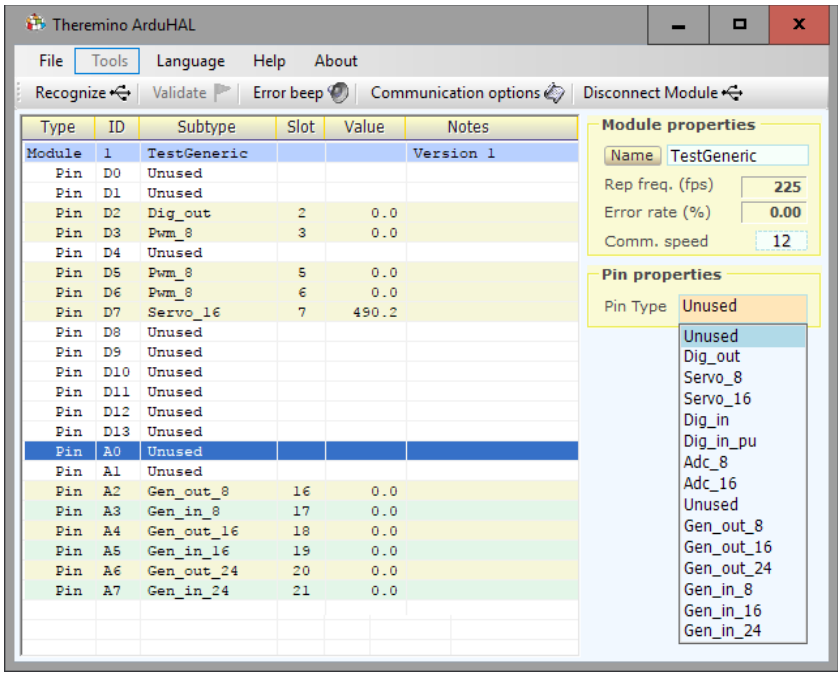
Con i moduli pre-programmati basta connettere il cavo USB e si evita di dover installare l'Arduino IDE e programmare l'Arduino.



Si lancia la applicazione ***ArduHAL*** e si configurano i Pin di ingresso uscita per leggere sensori, muovere motori ecc..

Si utilizzano le oltre cento applicazioni del sistema theremino che coprono quasi tutti i campi, dagli esperimenti scientifici alla musica, alla radioattività, alla didattica... vedere ***questa pagina***.

Il tutto funziona immediatamente senza scrivere una sola riga di firmware o di software.



Risolvere i problemi più comuni

Il modulo Arduino non appare nella lista dell'ArduHAL

Se il modulo non appare nella lista controllare nelle “Opzioni di comunicazione” che le porte COMM corrispondano.

Oppure, sempre nelle opzioni di comunicazione, impostare “Ports ALL” (che vuol dire “tutte le porte sono valide”).

Controllare anche che i Baud siano gli stessi di quelli configurati nel firmware (solitamente 500000).

Non si riesce a comunicare con l'Arduino e nemmeno a programmarlo

Se il modulo contiene un convertitore da USB a seriale di tipo CH340 potrebbe essere necessario installare il driver per CH340, come spiegato in ***questa pagina***.

La comunicazione avviene lentamente e con errori

Se il modulo contiene un convertitore da USB a seriale di tipo FT232, la casella che indica gli errori non segnerà zero e la comunicazione avverrà in modo discontinuo.

In questi casi si deve impostare il modo “Polling”, come spiegato in ***questa pagina***.

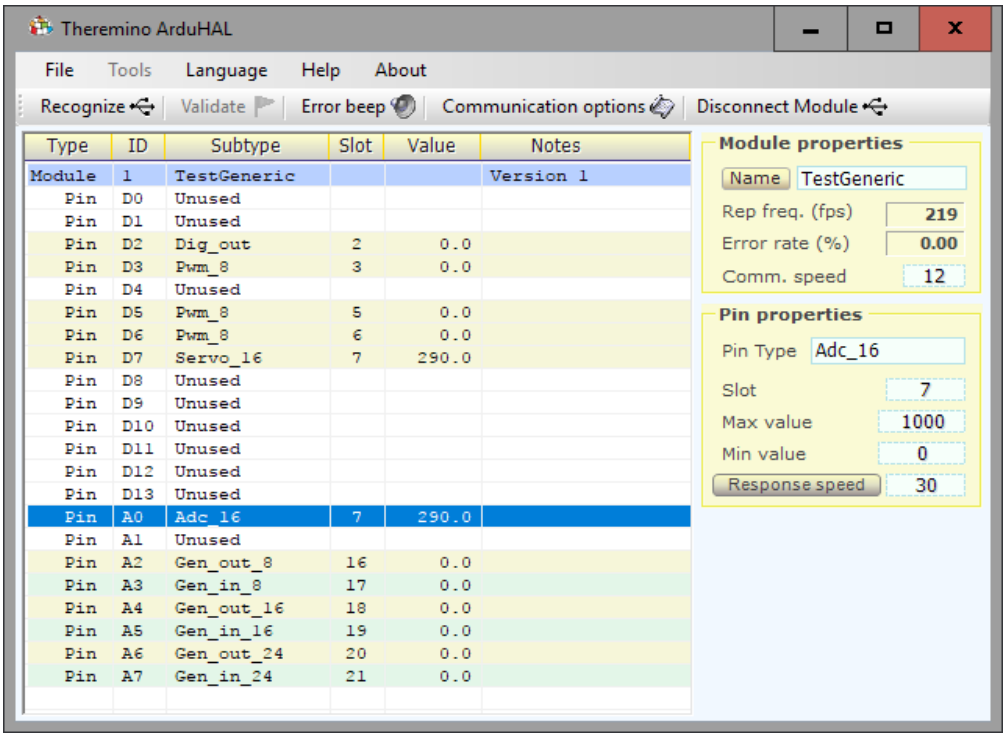
Leggere anche la pagina dedicata al ***chip FT232***.

La comunicazione è molto lenta

Se lo Sketch che si programma su Arduino contiene delle pause o dei calcoli lenti nella funzione loop(), la frequenza di scambio (fps) potrebbe abbassarsi notevolmente.

In questi casi si ottiene un buon miglioramento delle prestazioni con il modo “Asincrono”, come spiegato in ***questa pagina***.

La applicazione ArduHAL



Theremino ArduHAL con un modulo Arduino Nano collegato

La applicazione ***ArduHAL*** (Hardware Abstraction Layer), che si scarica da ***questa pagina***, appare con una interfaccia abbastanza semplice, ma svolge operazioni complesse, con algoritmi che beneficiano di molti anni di ricerche e sviluppo con i moduli Master.

Theremino ***ArduHAL*** è il cuore della comunicazione con l'hardware, sa comunicare con molti moduli Arduino contemporaneamente, conosce il protocollo di comunicazione seriale, conosce tutti i più comuni tipi di Input-Output e permette di configurarli facilmente.

Senza HAL comunicare con l'hardware di Arduino sarebbe difficile e richiederebbe molto tempo e lavoro. Per ogni tipo di InOut, ad esempio muovere un motore o anche solo accendere un LED, si dovrebbe scrivere del firmware apposito.

Nel sistema Theremino esistono anche altri due HAL, il primo si chiama semplicemente HAL e comunica via USB con i moduli Master, il secondo si chiama NetHAL e comunica via WiFi, rete e Internet con i moduli NetModule. In questo documento a volte ci riferiremo con il nome generico HAL per indicare tutte e tre le applicazioni.

Se si usano i moduli hardware *allora l'HAL è indispensabile e deve rimanere acceso, si può minimizzarlo, ma deve restare in funzione.*

Se non si usa hardware *allora l'HAL non è necessario, le applicazioni del sistema possono comunicare tra di loro, attraverso gli Slot, anche senza HAL.*

Quando si aggiungono o tolgono moduli*, alcune righe rosse avvertono che la configurazione è cambiata. Con il pulsante "Valida" si sceglie di perdere la vecchia configurazione e adeguarsi all'hardware attuale.*

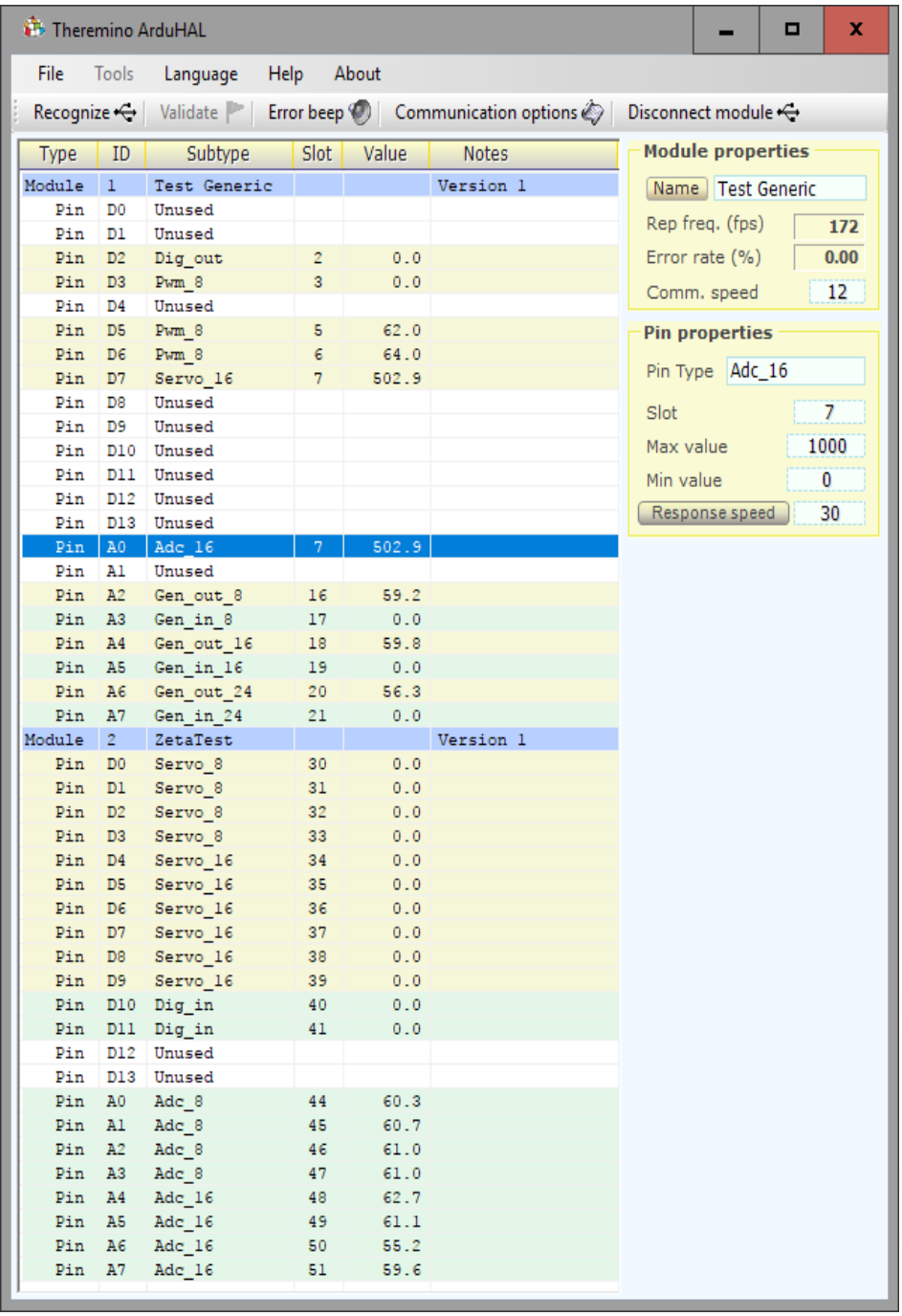
Collegare più moduli Arduino

La applicazione Theremino ArduHAL può comunicare con un qualunque numero di moduli Arduino.

I Pin di tutti i moduli vengono letti e scritti tutti contemporaneamente e alla massima velocità consentita da ogni modulo Arduino. In altre parole, un Arduino lento non provoca un rallentamento della comunicazione con gli altri.

Nella lista i moduli verranno presentati in ordine alfabetico per cui la lista apparirà uguale anche se si scambiano le porte USB.

Theremino ArduHAL collegato a due moduli Arduino Nano (44 Pin in totale)



Type	ID	Subtype	Slot	Value	Notes
Module	1	Test Generic			Version 1
Pin	D0	Unused			
Pin	D1	Unused			
Pin	D2	Dig_out	2	0.0	
Pin	D3	Pwm_8	3	0.0	
Pin	D4	Unused			
Pin	D5	Pwm_8	5	62.0	
Pin	D6	Pwm_8	6	64.0	
Pin	D7	Servo_16	7	502.9	
Pin	D8	Unused			
Pin	D9	Unused			
Pin	D10	Unused			
Pin	D11	Unused			
Pin	D12	Unused			
Pin	D13	Unused			
Pin	A0	Adc_16	7	502.9	
Pin	A1	Unused			
Pin	A2	Gen_out_8	16	59.2	
Pin	A3	Gen_in_8	17	0.0	
Pin	A4	Gen_out_16	18	59.8	
Pin	A5	Gen_in_16	19	0.0	
Pin	A6	Gen_out_24	20	56.3	
Pin	A7	Gen_in_24	21	0.0	
Module	2	ZetaTest			Version 1
Pin	D0	Servo_8	30	0.0	
Pin	D1	Servo_8	31	0.0	
Pin	D2	Servo_8	32	0.0	
Pin	D3	Servo_8	33	0.0	
Pin	D4	Servo_16	34	0.0	
Pin	D5	Servo_16	35	0.0	
Pin	D6	Servo_16	36	0.0	
Pin	D7	Servo_16	37	0.0	
Pin	D8	Servo_16	38	0.0	
Pin	D9	Servo_16	39	0.0	
Pin	D10	Dig_in	40	0.0	
Pin	D11	Dig_in	41	0.0	
Pin	D12	Unused			
Pin	D13	Unused			
Pin	A0	Adc_8	44	60.3	
Pin	A1	Adc_8	45	60.7	
Pin	A2	Adc_8	46	61.0	
Pin	A3	Adc_8	47	61.0	
Pin	A4	Adc_16	48	62.7	
Pin	A5	Adc_16	49	61.1	
Pin	A6	Adc_16	50	55.2	
Pin	A7	Adc_16	51	59.6	

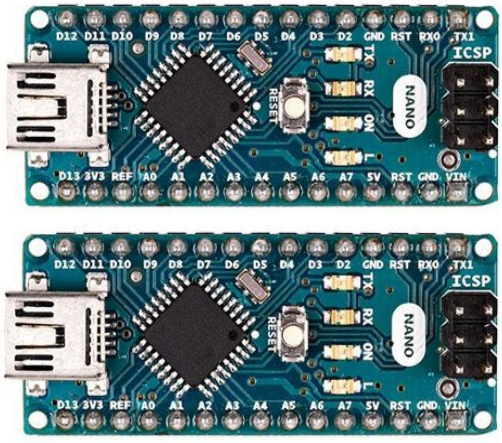
In questo esempio abbiamo collegato i due moduli Arduino tramite un unico cavo USB. Il cavo va a un HUB USB che fornisce quattro porte e i due Arduino sono stati collegati a due di queste porte.

Un qualunque numero di moduli potrebbe essere collegato con qualunque configurazione di HUB e di porte USB.

Si possono utilizzare cavi USB lunghi fino a una decina di metri, dato che si utilizza un modo di comunicazione abbastanza lento (rispetto alle capacità delle USB).

Ai due moduli sono stati dati i nomi “Test Generic” e “ZetaTest”,

I nomi vengono memorizzati negli Arduino stessi, per riconoscerli anche se si scambiano le porte USB.



Confronto tra i moduli Arduino e i nostri Master

Vantaggi

- Con Arduino è possibile aggiungere sensori collegati in I2C, SPI o in seriale. Abbiamo preparato dei tipi “generici” di comunicazione con il PC per cui basta scrivere qualche riga e importare una libreria per leggere anche i sensori più strani.
- Anche chi ha scarse conoscenze di programmazione può aggiungere qualche semplice funzione. Invece modificare il firmware dei nostri Master è quasi impossibile. Nemmeno un esperto potrebbe farlo, impiegherebbe troppo tempo e non ne varrebbe la pena.
- Con Arduino si possono programmare piccole operazioni da eseguire velocemente, ad esempio alzare una uscita e abbassarla dopo un tempo di pochi microsecondi. Queste stesse operazioni non sono invece possibili con un modulo Master, perché tutto va programmato sul PC e il minimo tempo di scambio consentito dalla USB è di un millisecondo o due.
- Un Arduino Nano costa meno della metà di un Master. Lo si trova anche a meno di tre euro su eBay.

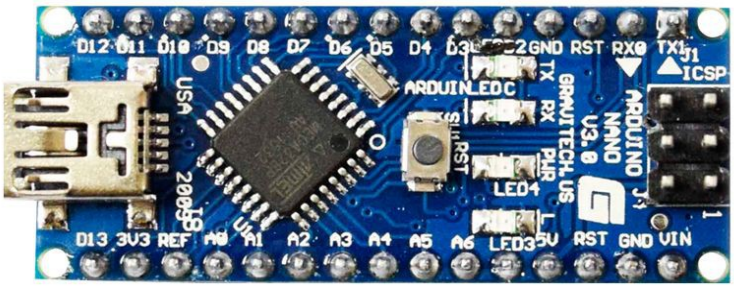
Svantaggi

- Il linguaggio semplificato di Arduino (gli Sketch) permette di programmare facilmente il suo firmware. Ma questa facilità comporta una grande perdita di prestazioni. Un esempio per tutti: gli Adc dei nostri Master vengono letti 16 volte e viene poi fatta la media, mentre su Arduino li possiamo leggere una volta sola, ed è già un carico non indifferente. Per cui gli Adc del Master danno quasi 14 bit (sovra-campionati) mentre quelli di Arduino solo 10 bit.
- La velocità di comunicazione con il PC è solo un quarto dei nostri Master anche nelle migliori condizioni (circa 250 scambi al secondo contro i normali 800...900 dei Master).
- I Pin sono configurabili con solo una decina i tipi contro i quasi trenta dei Master.
- Molti tipi di Pin, ad esempio i tasti capacitivi, i FastPwm e gli Stepper, non sono realizzabili, le prestazioni sarebbero troppo scarse, meglio usare un Master.
- Niente CapSensor e quindi niente Theremin (lo strumento musicale) con le caratteristiche cui siamo abituati.
- L'Adc24 non è collegabile, non ci abbiamo provato ma sappiamo per esperienza che la comunicazione SPI con il modulo Adc24 richiede tempi difficili da ottenere. Ci siamo riusciti sul nostro Master, perché avevamo a disposizione tutte le possibilità date da un processore molto potente. E per riuscirci abbiamo dovuto costruire un meccanismo molto complesso di Ping-Pong tra due interrupt. Cose del genere su Arduino sono praticamente impossibili (o per lo meno noi non abbiamo la minima idea di come si possano fare).

ATTENZIONE: Bisogna studiare bene quello che si scrive nel loop di Arduino, perché ogni ritardo si riflette sulla frequenza degli scambi con il PC. Basta una sola pausa di 100 mS per degradare la frequenza di scambio da 200 al secondo a meno di 10 (vedere ***questa pagina***).

I moduli Arduino Nano

Noi consigliamo di utilizzare l'Arduino Nano che è il più simile ai Master del sistema theremino.



L'Arduino Nano è piccolo ed economico, ma ha buone prestazioni e tutto il nostro firmware è stato sviluppato per lui. Abbiamo provato altri modelli, anche più costosi, ma andavano peggio.

Attenzione che ne esistono due versioni. Le versioni con il chip CH340 comunicano più velocemente. Il chip CH340 si trova sul lato inferiore. Per riconoscerlo leggere la prossima pagina.

I Nano che hanno il chip CH340 arrivano normalmente oltre i 300 scambi al secondo, mentre altri modelli non raggiungono nemmeno i 70. E alcuni modelli (ad esempio Zero, Leonardo, Micro, Esplora, Yun e e altri), non hanno la funzione “serial event” e quindi funzioneranno solo selezionando lo “Async mode” e quindi più lentamente.

Inoltre molti modelli di Arduino hanno un numero di Pin diverso dai 22 del Nano e richiederebbero di modificare la applicazione ArduHAL.

Lasciamo volentieri agli esperti di Arduino il piacere di fare esperimenti con gli Arduino più strani e di modificare il nostro firmware per farli funzionare. Ma attenzione, per fare un lavoro ben fatto in molti casi si dovrebbe modificare anche l'ArduHAL.

Consigliamo di comprare anche la base con i connettori a vite. I moduli Arduino non hanno i +5V e i GND disposti comodamente sui Pin come i nostri Master, per cui senza la base diventa difficile collegare i sensori e gli attuatori.

Con la base si possono collegare, ad esempio, i Servo spellando i loro fili ed avvitandoli. Non è proprio comodo come con i Master ma almeno si riesce a farlo senza saldature.

L'Arduino Nano ha gli ingressi e le uscite che lavorano tra 0 e 5V. Sui connettori è disponibile un Pin con il 5V e uno con il 3.3V per alimentare sensori che ne hanno bisogno.

I moduli Arduino con il chip FTDI

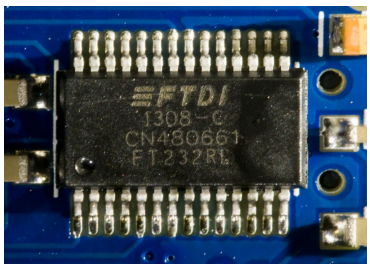
Ci sono tre tipi di Arduino, a seconda che il convertitore USB seriale sia un Atmega (ATMEL), un CH340 (Quin Heng Technology), o un FT232 (FTDI).

Alcune versioni di Arduino Nano hanno il chip FT2232, altre il chip CH340.

E' sempre meglio utilizzare le versioni con il CH340 perché vanno più veloci e creano meno problemi.

Convertitore AtMega

Convertitore CH340



Convertitore FT232

Il driver del convertitore FT232 non è "Thread safe", cioè se chiamato da applicazioni multithreading va in "deadlock". Il costruttore è al corrente del problema, nella Knowledge Base lo chiamano "lockup", e dicono che: *"alcune applicazioni multithread possono bloccarsi durante la comunicazione con il dispositivo".* Vedere la pagina: ***www.ftdichip.com/Support/Knowledgebase***

FTDI suggerisce anche una soluzione, *"To prevent lockup with a multithreaded application..."*, che naturalmente abbiamo provato, ma che non risolve il problema.

Pertanto abbiamo dovuto aggiungere alla applicazione ArduHAL l'opzione "Polling mode", che utilizza il driver in "polling", un metodo a singolo thread, quindi più lento, ma con il quale si possono far funzionare anche gli Arduino con il chip FT232.

Con il chip FT232 la velocità di comunicazione resta comunque bassa. Utilizzando l'opzione "Polling mode" si eliminano gli errori di comunicazione, ma si arriva al massimo a 70 fps.

Modificando le opzioni del driver dello FT232 è possibile incrementare questa velocità, non proprio a 350..400 come con i chip CH340, ma almeno a 170..200.

- Controllare che l'Arduino sia collegato alla USB
- Aprire "Gestione dispositivi"
- Nella sezione Porte (COM e LPT) aprire "USB Serial Port"
- Aprire "Impostazioni della porta" e poi "Avanzate"
- Impostare "Tempo di latenza (msec)" con il valore "1"

Il chip FT232 può creare ulteriori problemi, questa volta generati appositamente ***dal costruttore (FTDI)***. In alcuni casi il driver provoca volontariamente errori (driver di alcuni anni fa) o aggiunge messaggi nei dati seriali (driver recenti), per danneggiare i chip della concorrenza.

In questi casi non ci sono soluzioni, tranne sostituire i driver con un procedimento molto difficile. Chi volesse provarci può trovare in internet le informazioni, ***ad esempio qui***.

Moduli Arduino con il chip CH340

Alcuni modelli di Arduino comunicano attraverso un chip CH340, che converte i dati da USB a seriale. Questo chip ha bisogno di un driver apposito, che in alcuni computer potrebbe mancare.

Il chip CH340 funziona bene e velocizza la comunicazione, ma senza il driver giusto potrebbe accadere di non riuscire a comunicare con il modulo Arduino.

Sintomi

- Il sistema operativo segnala un errore quando si collega il cavo USB.
- Collegando il cavo USB non appare una nuova porta COM.
- Non si riesce a programmare l'Arduino
- La applicazione ArduHAL non visualizza il modulo Arduino nella lista

In questi casi si risolve il problema installando manualmente il driver.

Scaricare e installare il driver per Windows

Questa è la pagina ufficiale del produttore “Quin Heng Technology” per il download del driver, per tutti i tipi di CH340 (CH340G, CH340C, CH340B, CH340E, CH340T, CH340R, CH341A, CH341T, CH341H).

Il driver è valido per i sistemi operativi Windows 10/8.1/8/7/VISTA/XP e Server 2016..2000/ME/98, sia a 32 che a 64 bit, e certificato da Microsoft Digital Signature.

http://www.wch.cn/download/CH341SER_EXE.html

Per installare il driver

- *Aprire la pagina con un click sul link qui sopra*
- *Premere il tastone azzurro “DOWNLOAD”*
- *Attendere il completamento del download*
- *Fare click sul file CH341SER.EXE*
- *Acconsentire alle modifiche*
- *Premere “Install”*

Il firmware per Arduino

Il nostro sistema apre nuove possibilità per chi vuole utilizzare Arduino come Input Output per PC.

Con le ***oltre cento applicazioni del sistema Theremino*** si possono fare misure, automazione, strumenti musicali, giochi ed esperimenti di ogni genere, sia per la didattica che per i laboratori scientifici.

Molti hanno abbandonato Arduino in un cassetto dopo essersi accorti che programmarlo per comunicare con il PC non è facile. Ora questi moduli abbandonati potranno essere recuperati.

Per anni abbiamo atteso che qualche esperto di Arduino preparasse un protocollo di comunicazione con il nostro sistema. Ma alla fine abbiamo dovuto imparare il linguaggio di Arduino e scriverlo noi stessi.

Caratteristiche del firmware

Il nostro firmware occupa solo il 25% della memoria di programma, e il 25% della memoria dinamica di un Arduino Nano. Quindi più di due terzi dell'Arduino restano liberi per aggiungere altre librerie e firmware dell'utente.

In questo 25% abbiamo fatto stare tutto il protocollo di comunicazione seriale, nonché la gestione dei tipi di InOut più comuni, tra cui anche i Servo motori e la libreria “Servo” che li implementa.

Struttura del firmware

Abbiamo implementato il firmware di comunicazione con l'ArduHAL in una libreria.

Una libreria può facilmente essere caricata accedendo al menu “Sketch” e utilizzando il comando “include libreria” e infine scegliendo “Aggiungi libreria da file ZIP” (che si scarica da ***questa pagina***).

Inoltre una libreria può essere condivisa da molti progetti e quindi rende più facile la creazione di nuovi progetti con parti di firmware scritte dall'utente.

Struttura del file principale “ThereDuinoV1.ino”

La nostra classe si inizializza e funziona in modo del tutto automatico, ha solo bisogno delle parti seguenti nel file “ThereDuinoV1.ino”:

- All'inizio del file deve esserci la riga : `#include <Theremino.h>`
- Nella funzione “Setup” viene inizializzata la seriale. Normalmente il baud rate è 500'000, lo si può cambiare ma deve essere lo stesso che si imposta nelle “CommOptions”, accessibili della applicazione ArduHAL.
- Nello Sketch di Arduino, la funzione “Loop” è completamente vuota e potrà essere utilizzata dall'utente per scrivere il suo firmware e, eventualmente, le funzioni comunicazione “GenericRead” e “GenericWrite” per comunicare dati numerici con il NetHAL (leggere la pagina di questo documento che spiega le funzioni Generic).
- Fare attenzione a quello che si scrive nel loop di Arduino, perché ogni ritardo si riflette sulla frequenza degli scambi con il PC. Se non si abilita il “Modo asincrono” basta una sola pausa di 100 mS per degradare la frequenza di scambio, da 300 al secondo a meno di 10.

Scrivere il firmware nel modulo Arduino

Prima di tutto si deve scaricare l'Arduino IDE da questa pagina:
https://www.arduino.cc/en/main/software

Chi ha poca esperienza può scegliere il “Windows installer”. Altrimenti è meglio scaricare la versione “Windows zip” che potrà poi essere scompattata in qualunque cartella e utilizzata sul posto senza installarla. Inoltre la versione zip può facilmente essere spostata o copiata da un PC a un altro. L'unico suo difetto è che si deve fare tutto manualmente e quindi saper usare gli ZIP, le cartelle e i collegamenti.

Ed ecco la sequenza per scrivere il firmware nell'Arduino

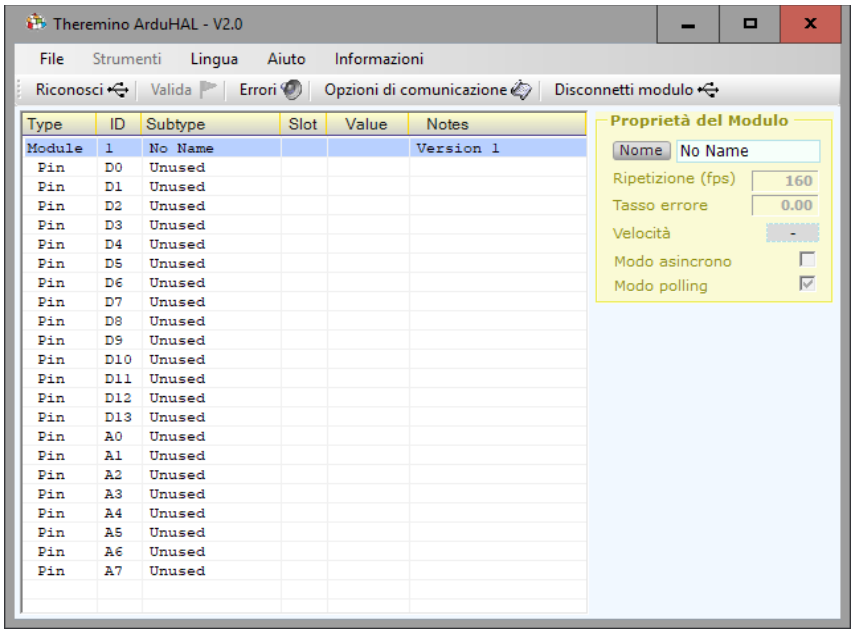
- Scaricare il file “ThereminoLibrary.zip” da ***questa pagina***.
- Avviare l'Arduino IDE
- Aprire il menu “Sketch”, “#include libreria” e scegliere “Aggiungi libreria da file ZIP”
- Nella finestra di dialogo individuare la cartella che contiene il file “ThereminoLibrary.zip”, sceglierlo e premere “Apri”.
- Aprire il menu “File”, scegliere “Apri” e ndividuare la libreria “Theremino” che dovrebbe essere in “Documenti/Arduino/Libraries”
- Aprire la cartella “ThereDuinoV1 “, selezionare il file “ThereDuinoV1.ino” e premere “Apri”
- Aprire il menu “Strumenti”.

  Selezionare la scheda (probabilmente Arduino Nano),

  Selezionare il processore (probabilmente ATmega328P)

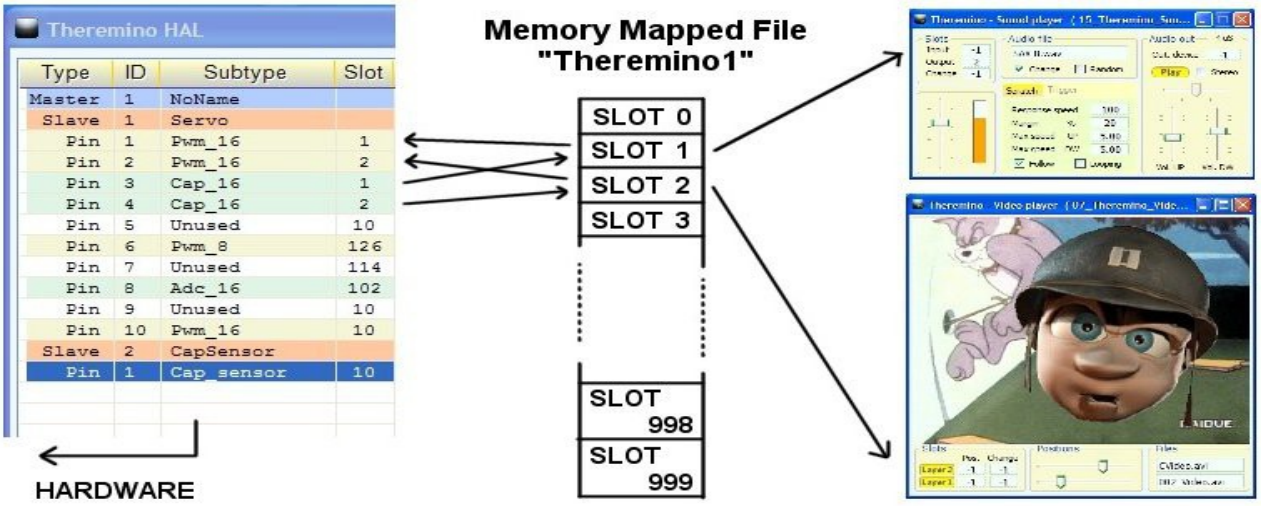
  Selezionare la porta COM. Per capire quale è la porta giusta, provare a collegare e scollegare Arduino dalla USB e ogni volta aprire “Strumenti / Porta”.
- Aprire il menu “Sketch” e premere “Carica”
- Se tutto è andato bene, la barra verde crescerà fino a destra e l'Arduino è programmato.

Infine si lancia la applicazione ArduHAL e devono apparire ventidue righe “Unused”, come si vede in questa immagine.



Gli "Slot"

Gli “Slot” del sistema Theremino sono identificati con un numero da 0 a 999 e fanno tutti parte del MemoryMappedFile con nome “Theremino1”. Ogni Slot contiene un numero “Float” che può essere letto o scritto da qualunque componente del sistema Theremino.



In questa immagine soltanto l’HAL scrive negli Slot ma in realtà tutti i componenti del sistema possono sia leggere che scrivere in uno qualunque degli Slot, anche se già usato da altri.

Nella scelta degli Slot da usare si deve fare attenzione a due cose:

- Controllare di non usare lo stesso Slot per sbaglio per due funzioni diverse.
- Evitare di scrivere in due sullo stesso Slot.

I Pin di ingresso, i quali scrivono negli Slot, sono evidenziati in verde chiaro. Se due o più Pin di ingresso hanno lo stesso Slot, allora la applicazione HAL avverte con righe rosse e con la dicitura **SLOT CONFLICT**.

Molte applicazioni e molti Pin possono leggere lo stesso Slot ma si deve evitare di configurare più di un Pin in scrittura sul medesimo Slot, facendolo non si rompe nulla ma si ottengono risultati indefiniti.

Type	ID	Subtype	Slot	Value	Notes
Master	1	TestSlotCo...			
Slave	1	MasterPins			Firmware V5.0
Pin	1	Adc_16	1	105.3	
Pin	2	Adc_16	2	99.5	
Pin	3	Dig_in	4	0.0	SLOT CONFLICT
Pin	4	Dig_in	4	0.0	SLOT CONFLICT
Pin	5	Dig_in	5	0.0	
Pin	6	Dig_in	6	0.0	
Pin	7	Dig_in_pu	7	1000.0	
Pin	8	Unused			

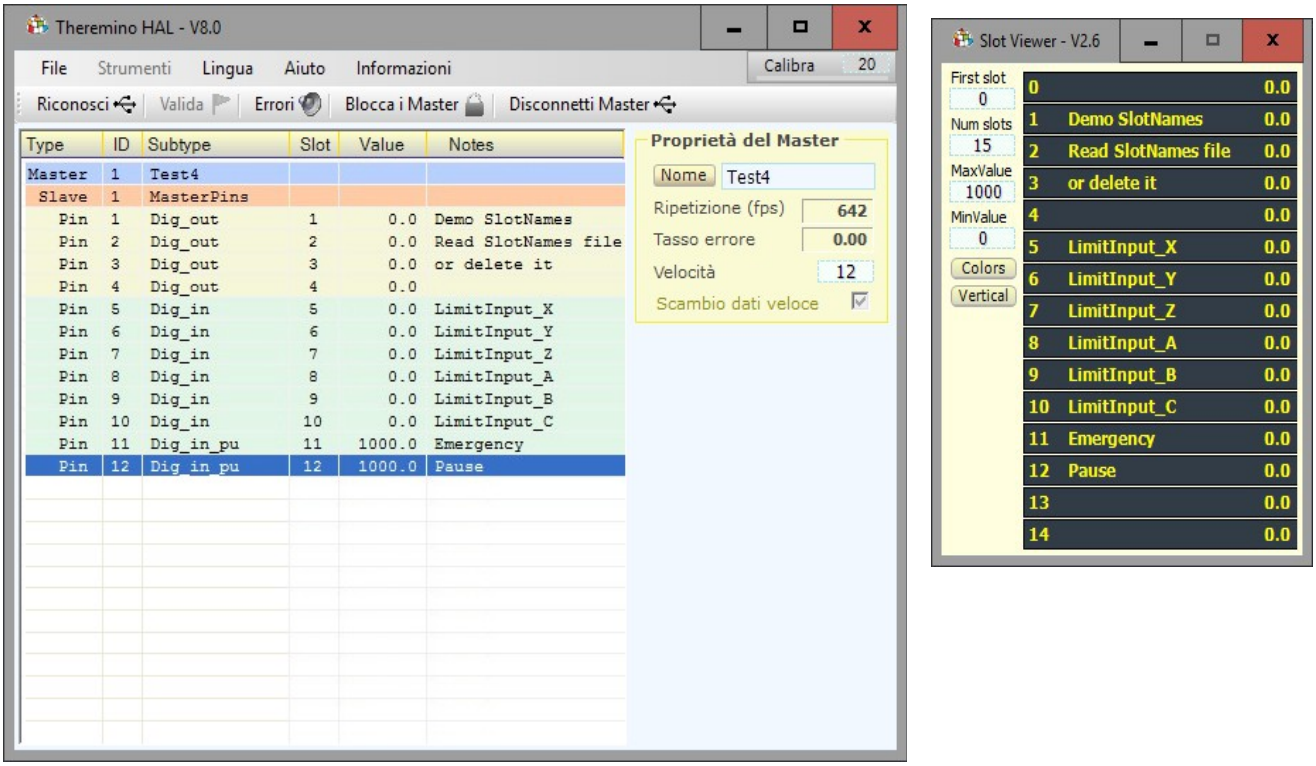
Se si inviano più flussi di dati verso lo stesso Slot allora i dati si mischiano e vince l’ultimo a scrivere, se si vogliono unire i dati in modo ordinato sono necessarie delle regole.

Per stabilire regole matematiche e logiche tra gli Slot, ed anche per scrivere algoritmi di comportamento complessi, si usano Theremino_Automation o Theremino_Script, oppure qualunque linguaggio di programmazione come C++, CSharp, VbNet o VB6 ma è anche possibile usare linguaggi visuali come Excel, MaxMSP, Processing, PureData, LabView e EyesWeb. Per MaxMSP sono pronti i Plugin e gli esempi qui: ***www.theremino.com/downloads/foundations***

Altre informazioni sulle comunicazioni in queste pagine:
www.theremino.com/technical/communications
www.theremino.com/technical/pin-types

I nomi degli Slot

Tutte le applicazioni HAL e anche lo SlotViewer possono visualizzare i nomi degli Slot (oppure annotazioni o commenti).



Importante notare che i nomi non sono legati ai Pin fisici, ma agli Slot.

I nomi si scrivono in un file, che deve chiamarsi “SlotNames.txt” e che deve stare nella stessa cartella di “Theremino_ArduHAL.exe” e di “Theremino_SlotViewer.exe”.

Se il file “SlotNames.txt” non esiste il campo dei commenti rimarrà vuoto.

Per modificare i nomi degli Slot, si apre il menu “File”, si sceglie “Modifica il file SlotNames.txt”, e lo si modifica con l'editor predefinito (normalmente NotePad o WordPad). Infine si salva il file e questo viene ricaricato automaticamente dall'HAL.

Le regole di scrittura sono semplici e sono mostrate nel file di esempio, che si trova nelle ultime versioni di HAL e SlotViewer.

Ogni linea del file inizia con il numero dello Slot, seguito da uno spazio e dal testo da visualizzare. La linea può anche continuare con una parte di commento, che non verrà visualizzata, preceduta da un apice singolo.

Se si vuole usare lo stesso file di commenti, sia per l’ArduHAL che per lo SlotViewer, si devono tenere i file “SlotNames.txt”, “SlotViewer.exe” e “ArduHAL.exe”, tutti nella stessa cartella.

Lo Slot dei comandi

Le applicazioni del sistema Theremino, o altre applicazioni create dagli utenti, possono inviare comandi e ricevere dati dall'ArduHAL, utilizzando uno slot speciale per comunicare.

Per esempio una applicazione potrebbe modificare i parametri di tutti i Pin, riscrivendo il file delle configurazioni e poi inviando il comando “Riconosci”. Oppure una applicazione potrebbe verificare quanti moduli sono effettivamente collegati, inviando il comando Riconosci, e poi leggendo il loro numero sullo slot dei comandi. Oppure una applicazione musicale potrebbe calibrare alcuni Pin, inviando il comando “Calibra” (attualmente questi Pin non sono implementati nei moduli Arduino).

Utilizzare altri Slot al posto dello Slot zero

Solitamente lo Slot dei comandi è lo zero, ma potrebbe accadere di voler utilizzare più applicazioni indipendenti sullo stesso PC. In questi casi ogni applicazione risiederebbe in cartelle separate insieme al suo ArduHAL e ai suoi moduli Arduino, utilizzando il file “CommOptions.txt”. In questi casi si può assegnare ad ogni ArduHAL uno Slot dei comandi diverso. Per i comandi si può utilizzare qualunque Slot (da 0 a 999) facendo però attenzione a non assegnarlo a nessun Pin.

Per assegnare un numero diverso da zero allo Slot dei comandi, si modifica manualmente l'ultima riga del file “Theremino_ArduHAL_INI.txt”. Quindi per utilizzare, ad esempio, lo slot 300, si scriverebbe: `CommandSlot= 300`. Attenzione a non eliminare il segno “=”. Se si sbaglia qualcosa l'HAL utilizza lo Slot zero e riscrive la linea corretta nel file INI.

Come inviare i comandi

Attualmente sono definiti due comandi:

- Riconosci — Si invia “NAN_Recognize”, oppure il numero “1”
- Calibra (*) — Si invia “NAN_Calibrate”, oppure il numero “2”

(*) Attualmente (Settembre 2018) nessun tipo di Pin dei moduli Arduino richiede la calibrazione.

Le applicazioni che non sono in grado di inviare i numeri speciali NAN (Not A Number), possono utilizzare i numeri “1” e “2” al posto dei valori “NAN_Recognize” e “NAN_Calibrate”.

Se si usano i numeri “1” e “2”. questi devono essere preceduti da una sequenza. La sequenza prevede due numeri (333 e 666) che corrispondono in realtà ai numeri in virgola mobile, con sette cifre di precisione, 333.0000 e 666.0000. Quindi è praticamente impossibile che un ADC o altri dispositivi possano inviare questa sequenza per errore.

Messaggi di risposta

Le risposte, e i messaggi di errore, vengono comunicati con numeri nello slot dei comandi.

- -1 — Il comando “Riconosci” è ancora in esecuzione.
- 0 — Non sono stati trovati moduli, la lista dei moduli è vuota.
- da 1 in su — Il numero di moduli che sono stati riconosciuti.
- NAN_MasterError — Uno dei Master collegati ha smesso di comunicare.

Lo Slot dei comandi - Esempi

Per inviare il comando “Riconosci”, si scrive:

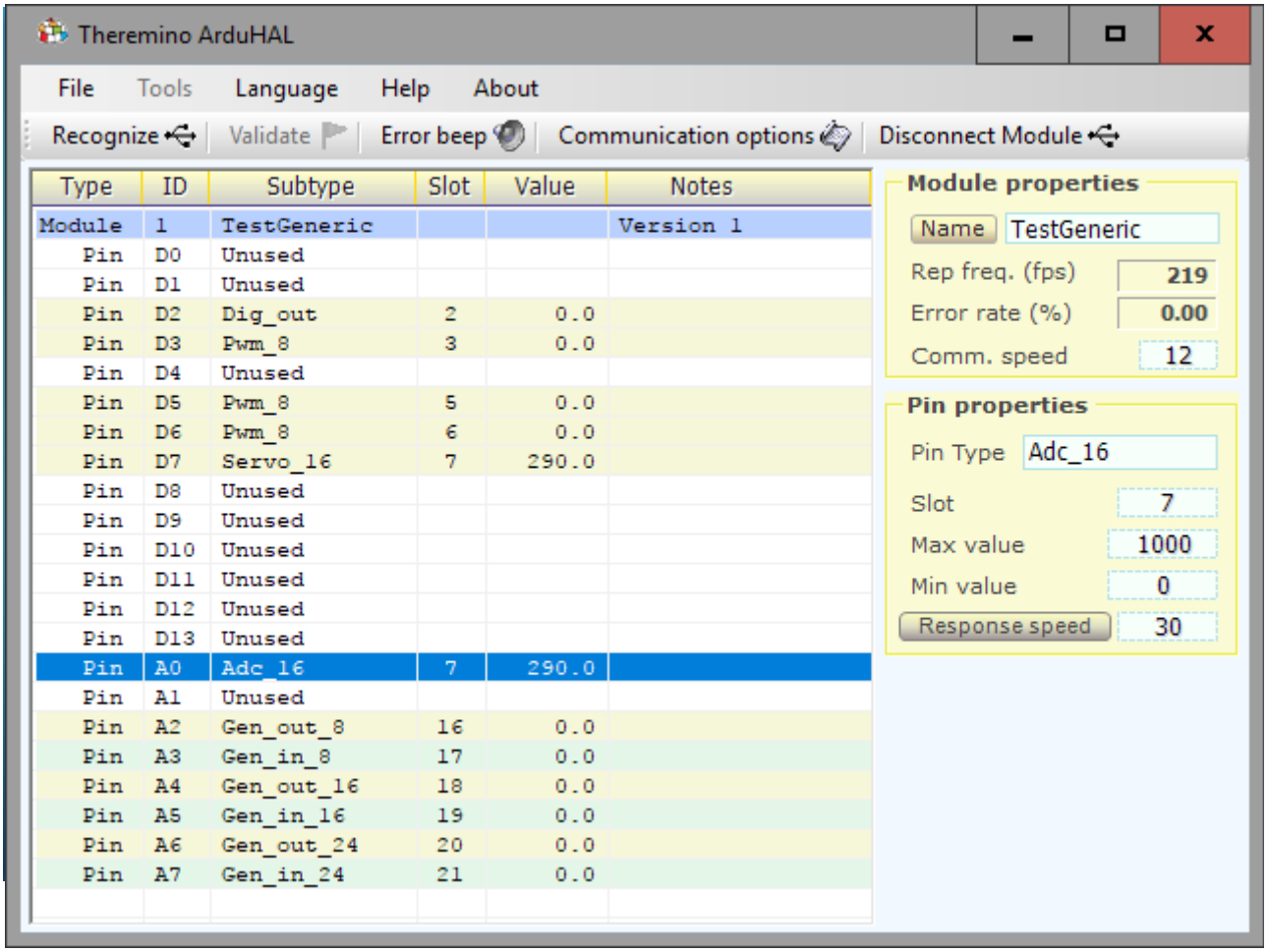
```
---------------------------------------- VbNet
Slots.WriteSlot (0, NAN_Recognize)

---------------------------------------- CSharp
Slots.WriteSlot(0, NAN_Recognize);

---------------------------------------- Theremino Script
WriteSlot (0, NAN_Recognize)
```

Come spiegato nella pagina precedente, alcune applicazioni (ad esempio Theremino Automation), potrebbero non essere in grado di utilizzare i numeri speciali NAN. Se non si utilizzano i NAN gli esempi precedenti diventano:

```
---------------------------------------- VbNet
Slots.WriteSlot (0, 333)
System.Threading.Thread.Sleep(50)
Slots.WriteSlot (0, 666)
System.Threading.Thread.Sleep(50)
Slots.WriteSlot (0, 1)

---------------------------------------- CSharp
Slots.WriteSlot(0, 333);
System.Threading.Thread.Sleep(50);
Slots.WriteSlot(0, 666);
System.Threading.Thread.Sleep(50);
Slots.WriteSlot(0, 1);

---------------------------------------- Theremino Automation
Slot 0 = 333
Wait Seconds 0.05
Slot 0 = 666
Wait Seconds 0.05
Slot 0 = 1

---------------------------------------- Theremino Script
WriteSlot (0, 333)
Threading.Thread.Sleep(50)
WriteSlot (0, 666)
Threading.Thread.Sleep(50)
WriteSlot (0, 1)
```

Le pause di 50 millisecondi servono per dare tempo all'HAL di leggere lo Slot.

I colori dell' HAL



Lo schema di colori aiuta a riconoscere i componenti e la loro configurazione

Il modulo Arduino (con nome TestGeneric) fornisce ventidue Pin:

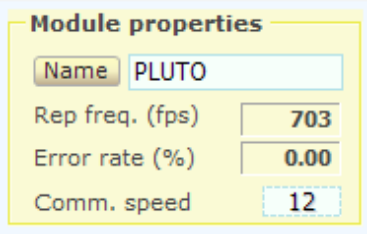
I Pin D0, D1, d4, da D8 fino a D13 e A1 sono "Non usati" e hanno lo sfondo bianco

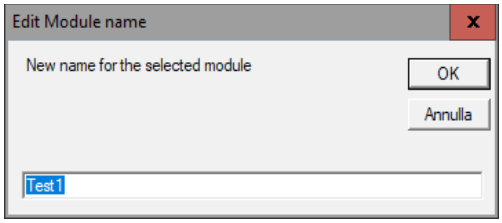
I Pin con sfondo giallo chiaro sono uscite (DigOut, Pwm, Servo e GenericOut)

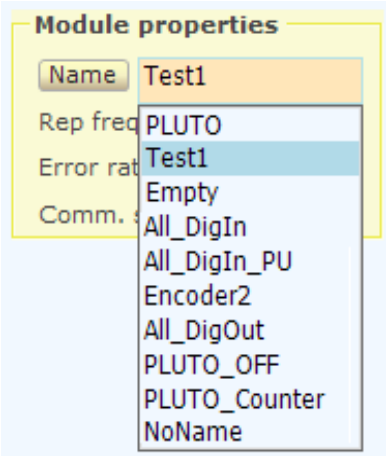
I Pin con sfondo verde chiaro sono ingressi (DigIn, Adc e GenericIn)

La riga "Pin A0 Adc16" di colore blu è la riga selezionata e le sue proprietà sono mostrate sulla destra.

Le proprietà del Modulo - Il nome







Il nome del Modulo selezionato può essere modificato in due modi:

- Premendo il bottone "Name" e modificando il suo nome.
- Cliccando sulla casella del nome e scegliendo una diversa configurazione dal menù a cascata.

Il nome del modulo viene scritto nel modulo hardware e serve per riconoscerlo quando lo si ricollega.

Un nuovo modulo appena collegato viene chiamato "NoName". E' bene prenderle l'abitudine di dargli subito un nome diverso, per distinguerlo da tutti gli altri.

Nei nomi dei moduli il "case" delle lettere (maiuscole o minuscole) non conta.

Se nel database dei moduli ci sono due moduli con lo stesso nome allora viene usata la configurazione del primo per ambedue. E' quindi importante dare nomi diversi ad ogni modulo (a meno che si vogliano avere moduli di ricambio con lo stesso nome di quello principale)

I moduli vengono sempre listati in ordine alfabetico, per cui se si cambia la porta USB l'ordine dei moduli non cambia.

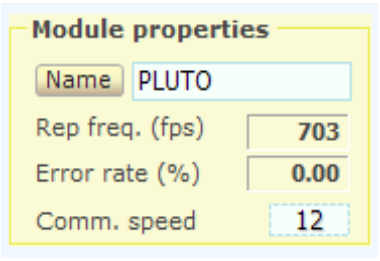
- - - - -

Il programma HAL riesce quasi sempre a usare la giusta configurazione quando si scollegano, sostituiscono e ripristinano i componenti, ma se si cambia nome ai moduli usando un diverso computer o con un altra applicazione HAL (situata una cartella separata - quindi con parametri separati) o in altri casi difficili e complessi, allora l'allineamento tra configurazione e hardware si perde.

Se si perde l'allineamento si dovrebbe ripristinare la configurazione manualmente, un Pin alla volta, ma gli esperti possono editare il file di configurazione e eventualmente copiare questo file interamente, o solo parte delle configurazioni, da una applicazione HAL a un'altra, su un altro computer o in un'altra cartella.

Quando la configurazione non è valida modificare il nome dei moduli non modifica il file di configurazione ma solo il nome scritto nell'hardware è quindi possibile modificare i nomi dei moduli fino a farli coincidere a quelli giusti nella configurazione.

Le proprietà del modulo - Comunicazione



- Numero di comunicazioni al secondo
- Percentuale di errori sulla linea seriale (normalmente zero)
- Regolazione della velocità di comunicazione

Il numero di comunicazioni al secondo "fps" dovrebbe normalmente essere superiore a 150 e spesso anche superiore a 200, aumentando il numero di Pin utilizzati questa frequenza potrebbe diminuire leggermente.

Per molte applicazioni una frequenza di 100 fps è più che sufficiente, per alcune applicazioni particolari è bene mantenere fps più alto possibile, almeno 200.

Le applicazioni che necessitano della massima banda passante, ad esempio il WaveAnalyzer non possono utilizzare moduli Arduino ma devono per forza utilizzare i Master o i NetModule.

La percentuale di errori normalmente è zero. Si dovrebbero avere errori solo in caso di gravi disturbi elettrici o se si seleziona una velocità in baud troppo alta.

Suggerimenti per aumentare la frequenza "fps"

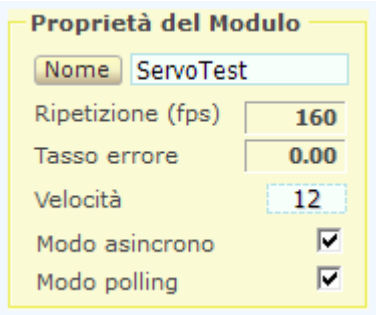
- Ridurre il carico di lavoro nell'Arduino (qualunque istruzione che perde tempo nel “Loop” rallenta la comunicazione).
- Aumentare la "Comm speed".
- Diminuire il numero di byte usati configurando come "Unused" tutti i Pin possibili.
- Diminuire il numero di byte usati configurando a 8 bit tutti i Pin che non necessitano di grande risoluzione.
- Disabilitare (se possibile) i modi Asincrono e Polling (vedere la pagina seguente).

Regolare la frequenza “fps”:

Con il valore "Comm speed" è possibile regolare la frequenza di aggiornamento “fps”.

Per aumentare la velocità di risposta sarebbe bene aumentare al massimo la frequenza di scambio, quindi impostare “Comm speed” a “12”. Ma per molte applicazioni cento scambi al secondo sono più che sufficienti, per cui normalmente si può abbassare "Comm speed" e caricare meno la CPU.

Le proprietà del modulo – Modi Asincrono e Polling



<-- Modo asincrono
<-- Modo polling

Modo asincrono

Se si abilita il modo asincrono la comunicazione con Arduino non è più sincrona con la funzione loop(). Per cui la frequenza di comunicazione (fps) non viene rallentata da quello che l'utente scrive in quella funzione.

Si può quindi utilizzare la funzione delay() anche con tempi di attesa molto lunghi. E si possono ottenere temporizzazioni tra gli eventi anche senza tecniche speciali (macchine a stati o interrupts) che per i principianti sarebbero del tutto incomprensibili.

Non sempre utilizzare il modo asincrono è un vantaggio, ma stabilire se utilizzarlo è facile. Basta controllare la frequenza di ripetizione (fps) e utilizzarlo solo se la fa aumentare.

Modo polling

Il modo polling (che vuol dire “interrogazione ciclica”) permette di utilizzare anche gli Arduino con il chip FT232 (per maggiori informazioni sullo FT232 leggere ***questa pagina***).

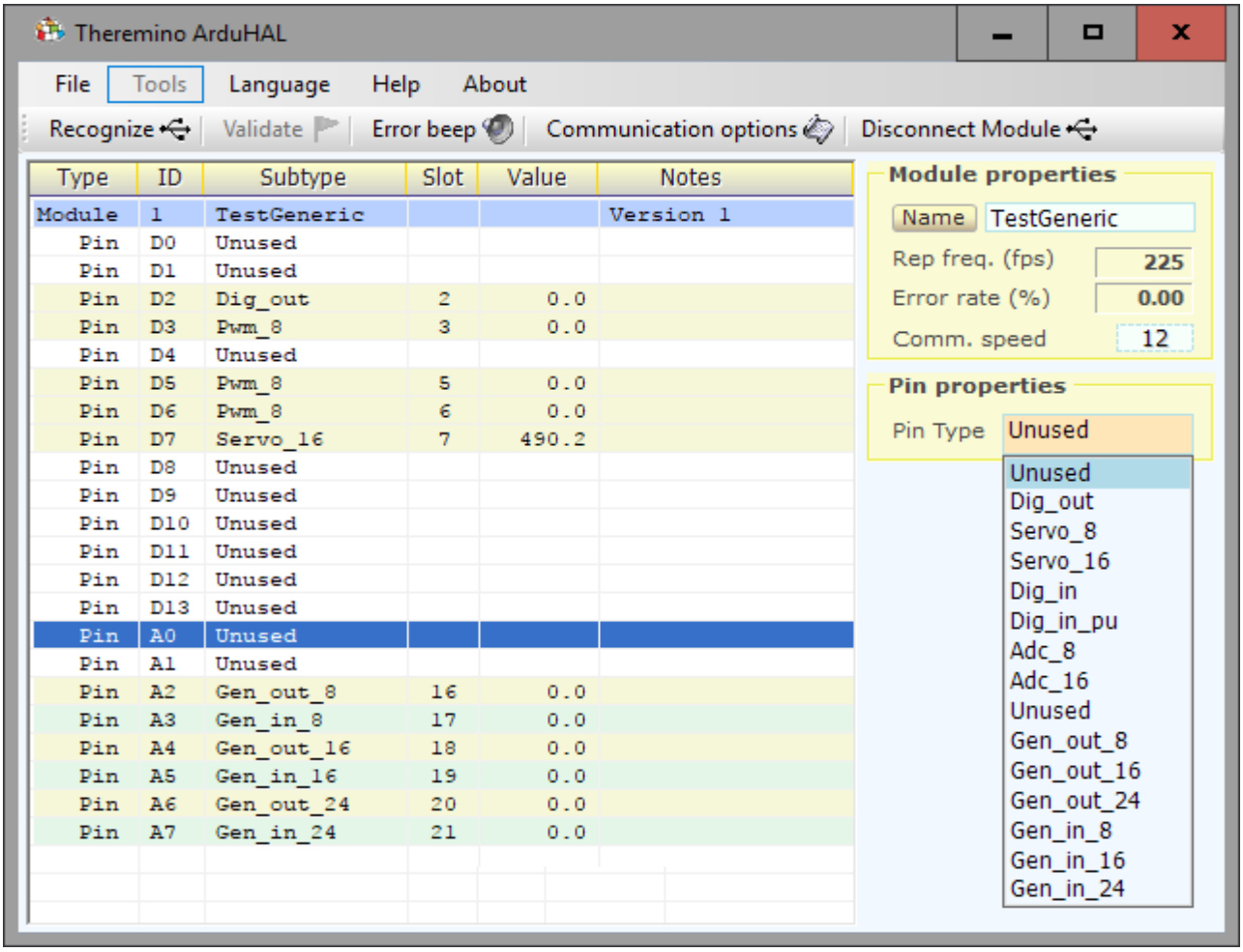
Utilizzare il modo Polling diminuisce la frequenza di scambio (fps), ma senza di esso si avrebbero continui errori di comunicazione e quindi un funzionamento a strappi.

Per stabilire se utilizzare il polling si può procedere in due modi:

- O si controlla se c'è un chip FT232 e in tal caso si abilita il modo polling.
- Oppure si controlla la casella “Tasso errore”.

Se la casella non segna 0.00 si abilita il modo “polling”. Poi si ricontrolla la casella degli errori, che dovrebbe azzerarsi completamente.

I moduli Arduino e i loro Pin



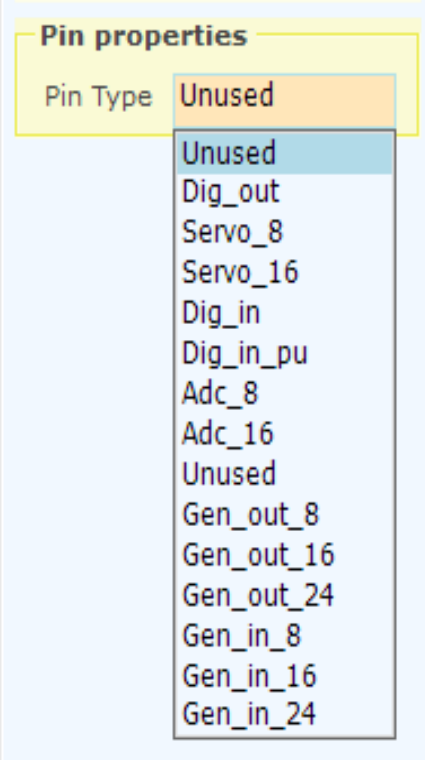
I Pin sono quasi tutti uguali tra loro e sono configurabili in molti modi diversi.

- D0 ... D13 non possono essere configurati come Adc.
- A0 ... A7 possono essere configurati in tutti i modi, Adc compreso.
- Solo i Pin D3, D5, D6, D9, D10 e D11 sono configurabili come Pwm.
- Se si utilizzano dei pin di tipo Servo allora D9 e D10 non sono più utilizzabili come Pwm. Non abbiamo bloccato questi Pin perché alcuni modelli di Arduino hanno diverse limitazioni. Maggiori informazioni sulla libreria Servo in ***questa pagina***

Il nostro firmware è stato scritto e provato per L'Arduino Nano.
Ed è stato parzialmente provato anche sull'UNO (che ha due pin ADC in meno)
Altri modelli potrebbero funzionare parzialmente o non funzionare del tutto.
Per alcuni modelli si dovrebbero modificare sia la libreria che la applicazione ArduHAL.

I tipi di Pin

I Pin sono configurabili come:



- Non usato

- Uscita digitale
- Uscita PWM (490 o 960 Hz a seconda dei Pin)
- Uscita per servo-comandi

- Ingresso digitale
- Ingresso ADC per potenziometri e trasduttori
- Ingresso di conteggio, frequenza e periodo

- Ingressi e uscite generici da 8, 16 o 24 bit

I Pin speciali:

- I Pin D0 e D1 accettano solo i tipi Gen_in e Gen_out
- I Pin da D2 a D13 accettano tutti i tipi, tranne l'ADC
- I Pin da A0 ad A7 accettano tutti i tipi, ADC compreso
- Solo i Pin D3, D5, D6, D9, D10 e D11 sono configurabili come Pwm
- Se si utilizzano pin di tipo Servo allora D9 e D10 non sono più utilizzabili come Pwm

Sono anche disponibili i tipi di pin “Generici” che servono per trasferire dati tra il firmware e il PC. Per mezzo di questi tipi è possibile aggiungere qualche riga di firmware e scambiare valori numerici attraverso la applicazione ArduHAL e infine con gli Slot del sistema theremino. Si possono ad esempio leggere sensori collegati in I2C e inviare i valori al PC. Maggiori informazioni sui Pin generici nella pagina ***I tipi di pin generici***.

Tutti i Pin sono anche configurabili come "Non usato", questo permette di diminuire il numero di byte che transitano sulla linea seriale e sulla USB e massimizzare il numero di scambi per secondo.

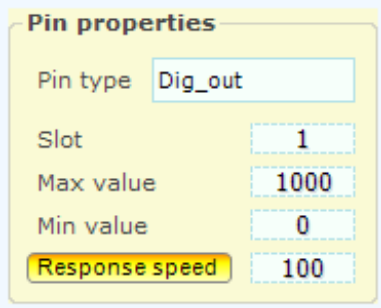
La scelta tra 8 e 16 bit, disponibile per alcuni tipi di Pin, permette di avere la massima risoluzione (16 bit) o una minore risoluzione (8 bit), ma un maggiore risparmio di bit e quindi una maggiore velocità di comunicazione.

I tipi con pullup, il cui nome termina per "_pu", permettono di collegare facilmente interruttori, pulsanti e dispositivi open-collector, senza dover aggiungere resistori esterni (corrente di PullUp tipica = 250 uA).

Questi tipi di Pin sono validi per **l'Arduino Nano.**

Altri modelli di Arduino hanno possibilità simili ma non li abbiamo provati.

Il parametri comuni di tutti i Pin



"Slot" indica dove scrivere o leggere i dati. Gli Slot sono mille, numerati da 0 a 999, e possono essere letti o scritti da tutti i Pin e da tutte le applicazioni del sistema Theremino.

Attenzione: molte applicazioni e molti Pin possono leggere dallo stesso Slot,
ma si deve evitare di configurare più di un Pin in scrittura sul medesimo Slot,
facendolo non si rompe nulla ma si ottengono risultati indefiniti.

"Max value" normalmente si tiene a mille e indica il valore che il Pin deve avere quando è al massimo.

"Min value" normalmente si tiene a zero e indica il valore che il Pin deve avere quando è al minimo.

Regolando Max value e Min value con valori diversi da 0 e 1000 si può ottenere qualunque rapporto di scala e taratura. Se si scambiano i due valori (valore min maggiore di max) allora la scala si ribalta, questo è utile per invertire il movimento dei servocomandi o ribaltare la lettura di sensori che agiscono al contrario.

"Response speed" regola il filtro IIR (Infinite Impulse Response) per il migliore compromesso tra rumore e velocità di risposta. Con il valore 100 il filtro è disabilitato e si ha la massima velocità di risposta, con il valore 1 si ha il massimo filtraggio (eliminazione di ogni tremolio) ma una risposta molto lenta (circa un secondo). Normalmente si usa il valore 30 che fornisce un buon filtraggio e risponde abbastanza velocemente.

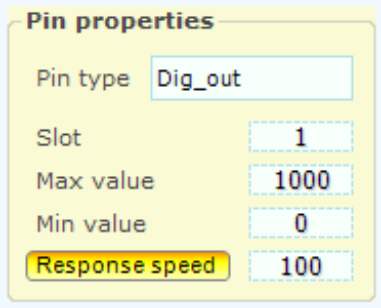
Se il pulsante **"Response speed"** è premuto, il filtro IIR si adatta alle variazioni in modo da ottenere una maggiore reattività quando vi sono ampie variazioni e un maggiore smorzamento, quando le variazioni sono minori. Come risultato si ottiene una buona stabilità delle cifre, senza penalizzare troppo il tempo di assestamento.

I segnali di alcuni sensori potrebbero funzionare male con "velocità di risposta" premuto. Questo è particolarmente vero per i sensori che producono un segnale con piccole variazioni intorno ad un alto valore di base. In questo caso il segnale non arriva al valore finale, o è molto lento ad arrivare. Se si sperimenta questo, disabilitare **"Response speed"**.

Le tensioni di IN-OUT tra 0 e 5V sono valide per Arduino Nano, altri modelli possono essere diversi.

I tipi di Pin in "Output" --> Dig / Pwm / Servo

Dig_out

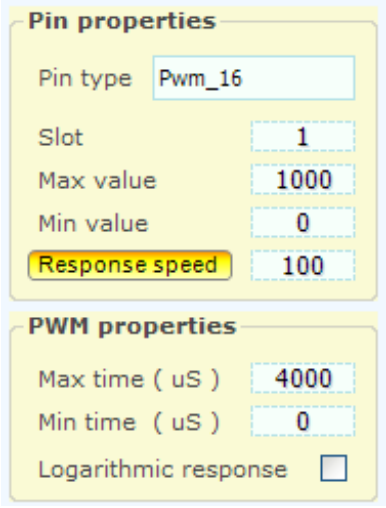


Questo tipo di Pin fornisce una uscita digitale.

Il valore in arrivo da uno Slot, limitato tra "Min value" e "Max value" e filtrato da "Response speed" viene confrontato con il valore intermedio tra "Min value" e "Max value", se lo supera il Pin si accende altrimenti si spegne.

Il Pin può assumere solo le tensioni 0V (spento) e 5V (acceso) e la corrente di uscita è limitata a circa +/-10 mA

Pwm_8 e Pwm_16



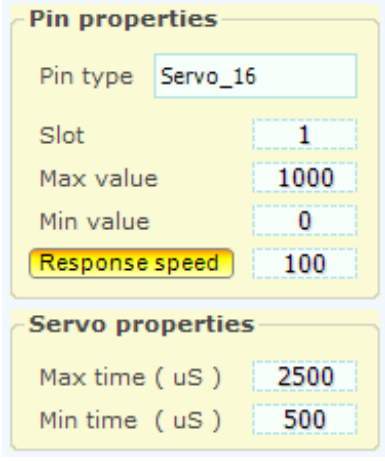
Uscita PWM cioè “Modulazione della larghezza degli impulsi”

Il valore in arrivo da uno Slot, limitato tra "Min value" e "Max value" e filtrato da "Response speed" viene trasformato in impulsi di larghezza tra "Min time (uS)" e "Max time (uS)"

Il tempo di ripetizione degli impulsi è 2000 uS (500 Hz) abbastanza veloce da poter accendere un led con intensità variabile. Per utilizzi che richiedono una vera tensione variabile si aggiunge un filtro passa basso composto da un resistore e un condensatore.

Il Pin fornisce impulsi tra le tensioni 0V (spento) e 5V (acceso) e la corrente di uscita è limitata a circa +/- 10 mA

Servo_8 e Servo_16



Questo tipo di Pin pilota direttamente i servo comandi.

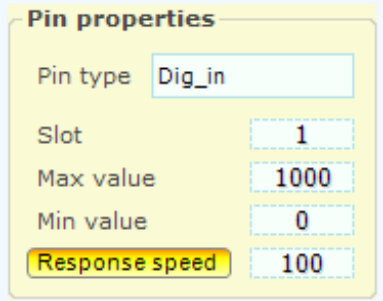
Il valore in arrivo da uno Slot, limitato tra "Min value" e "Max value" e filtrato da "Response speed" viene trasformato in impulsi di larghezza tra "Min time (uS)" e "Max time (uS)"

Il tempo di ripetizione degli impulsi è adeguati ai normali servocomandi per modellistica che tra il tempo min e max ruotano di circa 180 gradi.

Il Pin fornisce tensioni di 0 e 5Volt, adeguate a tutti i normali servocomandi alimentati da 4 a 6 Volt e una corrente sufficiente a pilotare decine di servo in parallelo.

I tipi di Pin in "Input" <-- Dig / Adc

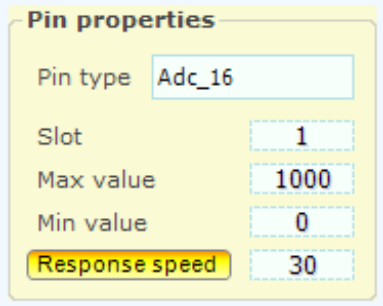
Dig_in e Dig_in_pu



Questo tipo di Pin fornisce un ingresso digitale.

Il valore di tensione viene letto con uno Schmitt Trigger, con soglia bassa = 1.5V e soglia alta = 3V, e trasformato in una informazione Acceso-Spento che infine diventano "Max value" e "Min value". Il valore viene infine filtrato con "Response speed" e poi scritto nello Slot. Il filtraggio produce valori intermedi e approssimativamente proporzionali al rapporto di tempo tra Acceso e Spento

Adc_8 e Adc_16



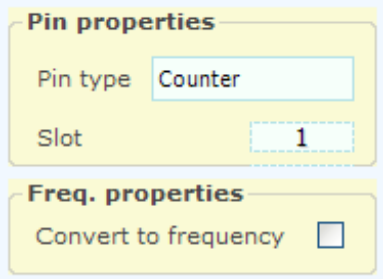
Questo tipo di Pin fornisce un ingresso analogico.

Il valore di tensione da 0V a 5V viene trasformato in un numero tra "Min value" e "Max value". Il valore viene infine filtrato con "Response speed" e poi scritto nello Slot. Il filtraggio riduce il rumore presente nel segnale di ingresso, ma rallenta la risposta. Il valore 30 rappresenta un buon compromesso tra velocità e rumore.

I tipi di Pin in "Input" <-- Counter

Questo tipo non è attualmente implementato nel firmware dell'Arduino.

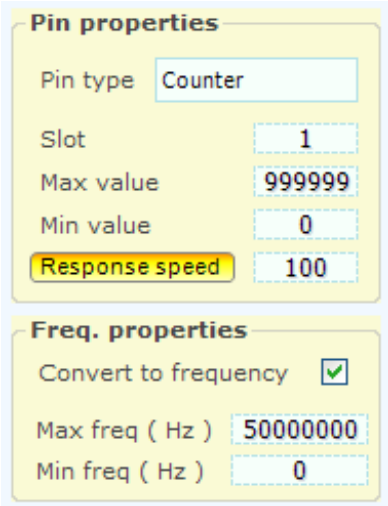
Counter e Counter_pu



Tutti i Pin possono essere programmati come Counter o Counter_pu ma la velocità di conteggio massima è abbastanza limitata, intorno a qualche KHz, dipendente dal carico sul microcontrollore e dal duty-cycle del segnale.

Se si necessita di una velocità superiore si devono usare i FastCounter del Theremino Master.

Counter e Counter_pu con l'opzione "Freq"



I Pin programmati come Counter o Counter_pu possono essere trasformati da contatori a frequenzimetri.

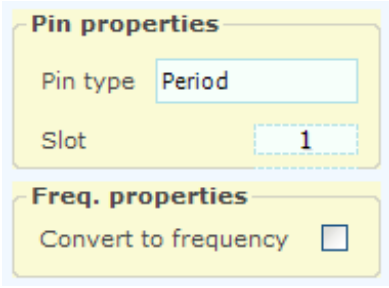
Il valore di frequenza limitato tra "Min freq" e "Max freq", viene poi rapportato tra "Min vale" e "Max value", filtrato con "Response speed" e infine inviato allo Slot.

I Pin di tipo “Counter” e “Counter_Pu” usano 16 bit per la trasmissione dei dati.

I tipi di Pin in "Input" <-- Period

Questo tipo non è attualmente implementato nel firmware dell'Arduino.

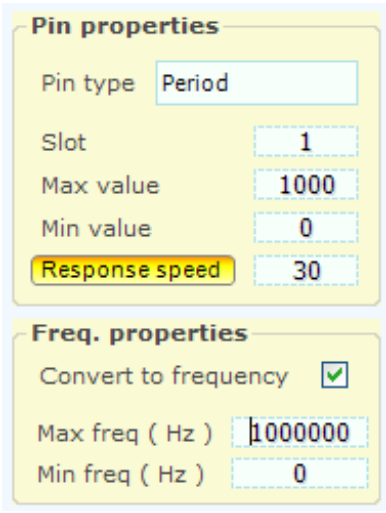
Period e Period_pu



Questo tipo di Pin misura il periodo di una forma d'onda ripetitiva, da salita a salita, fino ad un periodo massimo di circa 260 secondi.

La risoluzione è di mezzo microsecondo e la precisione è del +/- 1% in un range di temperatura ambiente da 0C a 50C

Period e Period_pu con l'opzione "Freq"



I Pin programmati come Period o Period_pu possono essere trasformati da contatori a frequenzimetri.

Questa tecnica permette di misurare frequenze molto basse (fino a circa un decimo di Hertz) con altissima risoluzione.

Il valore di frequenza limitato tra "Min freq" e "Max freq", viene poi rapportato tra "Min value" e "Max value", filtrato con "Response speed" e infine inviato allo Slot.

I Pin di tipo "Period" e "Period_pu" usano 32 bit per la trasmissione dei dati.

I tipi di Pin “Generici”

Quello che spiegheremo in questa pagina vale solo per leggere o scrivere sensori o attuatori speciali, ad esempio sensori o motori che comunicano in I2C. Oppure per pre-condizionare i dati con un proprio firmware prima di inviarli al PC, o dal PC verso gli attuatori.

Per leggere e scrivere i normali Ingressi e Uscite non è necessario utilizzare le funzioni spiegate in questa pagina, basta configurare i Pin con l'ArduHAL e tutto funziona immediatamente.

- - - - - -

Nelle prossime righe sono mostrati i sei tipi fondamentali.

- La prima coppia rimanda all'ArduHAL (sul Pin 1) il valore a 8 bit che arriva dal Pin 0
- Le altre due coppie fanno lo stesso a 16 e a 24 bit e utilizzando i Pin A0, A1 A2 e A3

```
uint32_t n;

n = Theremino.genericRead8(0);
Theremino.genericWrite8(1, n);

n = Theremino.genericRead16(A0);
Theremino.genericWrite16(A1, n);

n = Theremino.genericRead24(A2);
Theremino.genericWrite24(A3, n);
```

Una volta scritte queste righe nella funzione Loop() di Arduino, le si possono provare con l'ArduHal, modificando il valore della colonna Value con il mouse o aiutandosi con lo SlotViewer.

- - - - - -

Nelle funzioni GenericWrite e GenericRead si deve indicare un Pin da utilizzare e quello stesso Pin deve essere configurato nell'ArduHAL con il tipo Gen_in o Gen_out corrispondente. Se si fanno errori si avranno risultati imprevedibili.

I Pin sono tutti utilizzabili per comunicare dati “generici”. Come nome dei Pin si può utilizzare un numero da 0 a 21. Per gli ultimi otto Pin si possono anche utilizzare i nomi da A0 ad A7.

Si consiglia di utilizzare per primi D0 e D1, che non sono utilizzabili come Pin fisici.

Quando nell'ArduHAL si imposta un Pin come generico, il Pin fisico corrispondente non viene più letto (o scritto) automaticamente dal nostro firmware, ma si dovrà scrivere il firmware necessario. Quindi per utilizzare queste funzioni si deve saper pianificare e organizzare senza errori, e ci vuole una buona esperienza nella programmazione.

Il documento “Leggere un sensore I2C”, che si scarica da ***questa pagina***, è un buon esempio di utilizzo dei Pin Generici.

Utilizzare i tipi di Pin “Generici”

Inviare dati dai sensori verso l'ArduHAL e infine alle applicazioni sul PC

- Si sceglie un Pin da utilizzare
- Nel Loop di Arduino si legge il sensore e si interpretano i suoi dati
- Nel Loop di Arduino si invia il dato con “GenericWrite” (8, 16 o 24)
- Sull'ArduHal si legge il dato configurando il Pin come Gen_in (8, 16 o 24)
- Sull'ArduHal si imposta il Pin scelto con uno Slot attraverso il quale il dato del sensore andrà alle applicazioni che lo utilizzano.

Inviare dati dalle applicazioni sul PC all'ArduHAL e infine verso l'hardware

- Sull'ArduHAL si sceglie un Pin da utilizzare
- Sull'ArduHal si imposta il Pin scelto con uno Slot attraverso il quale inviare i dati numerici
- Sull'ArduHal si inviano i dati ad Arduino configurando il Pin come Gen_out (8, 16 o 24)
- Nel Loop di Arduino si legge il dato con GenericRead (8, 16 o 24)
- Nel Loop di Arduino si interpreta e traduce il dato numerico e lo si invia all'hardware.

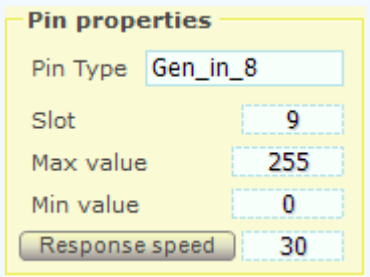
Trasferire un valore intero da Arduino verso uno Slot del PC

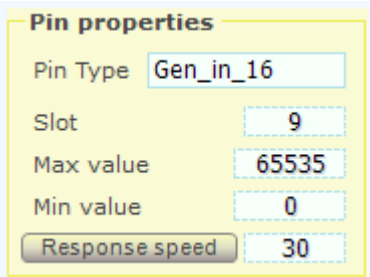
I valori in arrivo da Arduino sono interi che stanno in 16 bit (da 0 a 65535). Ma in alcuni casi potrebbero essere valori piccoli e quindi stare in 8 bit (da 0 a 255), oppure valori molto grandi e quindi aver bisogno di 24 bit (da 0 a 16777215).

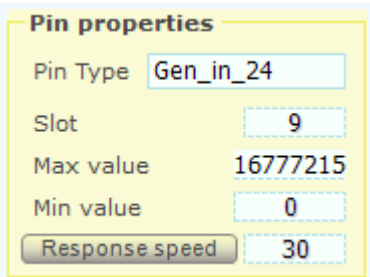
Lo standard del sistema Theremino è rapportare tutti questi valori ad una escursione “normalizzata” da 0 a 1000.

In alcuni casi, come ad esempio la lettura di una distanza da un sensore che fornisce già il valore in millimetri, può essere utile trasferire il valore prodotto dal sensore verso uno Slot senza modificarlo.

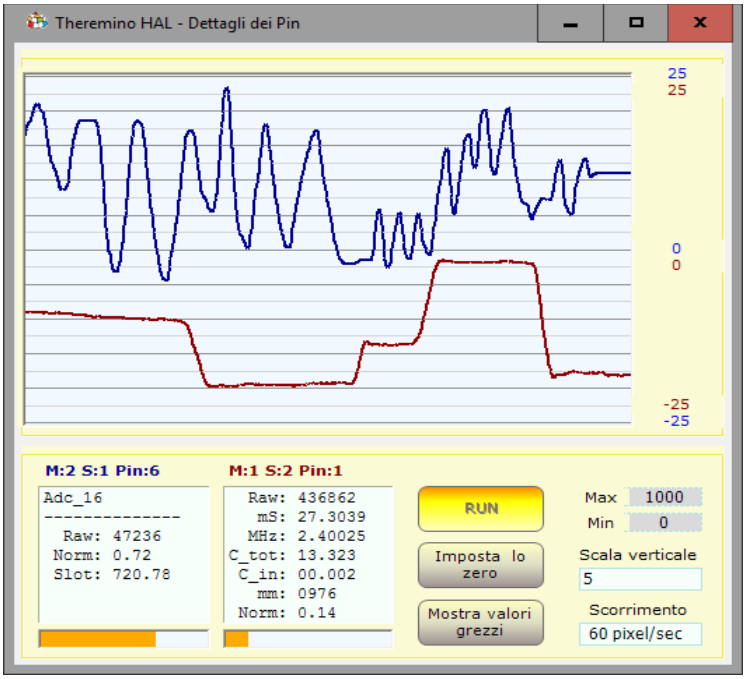
Per ottenere questo comportamento modificare la casella “Max value” con un valore adeguato al numero di bit che si usano, come si vede in queste immagini.







Il visualizzatore dei dettagli dei Pin



Con doppio click sulla linea di un Pin attivo, si apre questo strumento. Per due segnali, cliccare prima su un Pin e poi sul secondo, con un click singolo.

La scala verticale può essere impostata a “Scale Min-Max”, che corrisponde al valore delle caselle Min e Max.

Oppure la si può impostare in 24 livelli da 0.01 a 50000 punti, per ogni divisione verticale (dieci linee scure). Se si utilizzano queste impostazioni, per centrare le tracce si preme "Reset zero".

In alcuni casi può essere utile visualizzare i valori grezzi ("Raw" in inglese) Per visualizzare i valori "Raw" si abilita il tasto "Show raw count".

Il controllo "Scorrimento" regola la velocità di scorrimento del grafico da 0.1 Pixel al secondo fino a 60 Pixel al secondo.

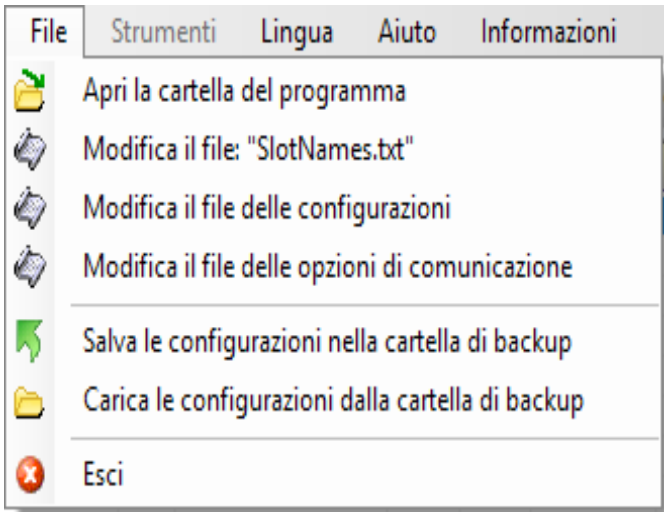
I due riquadri di testo mostrano i dettagli interni dei Pin, il titolo indica di quale modulo e Pin si tratta. In questa immagine il testo "M:1 Pin:2" significa "Modulo 1, Pin 2".

I dettagli dei Pin possono aiutare nel controllo e la regolazione dei dispositivi di Ingresso Uscita (Sensori e Attuatori).

Alcuni tipi di Pin sono più complessi e presentano più valori intermedi. In genere esiste un valore "Raw" con valori molto variabili a seconda del tipo di Pin, un valore "Normalizzato" che va sempre da 0 a 1, e un valore "Slot" che normalmente va da 0 a 1000 e che è il valore "semplificato" disponibile sugli Slot e usabile facilmente da tutti i software ad alto livello.

- **Raw** Valore "grezzo" che può essere un conteggio, un tempo, una tensione o altro.
- **mS** Tempo in millisecondi
- **uSec** Tempo in microsecondi
- **Smoot** Valore che è stato passato in un filtro FIR (usato solo nei Cap8 e Cap16)
- **Mean** Valore medio (usato nei tipo Cap8 e Cap16 come calibrazione dello zero)
- **Norm** Valore normalizzato tra zero e uno
- **Slot** Valore scritto o letto dallo Slot associato al Pin (normalmente da 1 a 1000)
- **Out** Valore digitalizzato che può valere solo "0" o "1" (usato solo da DigOut)

I comandi dei menu



Aprire la cartella di lavoro può essere utile per modificare i file di documentazione e delle lingue.

Modificare il file: "SlotNames" i commenti (o nomi degli slot) sono ***spiegati qui***.

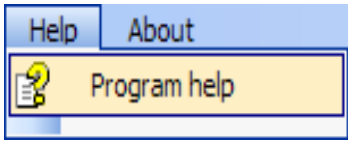
Editare le configurazioni può essere comodo in certi casi. Per maggiori informazioni leggere anche "Domande e risposte" nell'ultima pagina di questo documento.

Modificare le opzioni di comunicazione *Viene aperto il file delle opzioni di comunicazione per modificare le porte (Com1, Com2 ecc...), le velocità (Baudrate) e i nomi dei moduli da riconoscere. Nel file delle opzioni di comunicazione sono spiegate le possibilità di scelta.*

Salvare le configurazioni come backup *permette di fare delle copie di sicurezza delle configurazioni. Se i file di configurazione vengono modificati per errore sarà poi possibile ricaricarli da una versione precedente. Le versioni precedenti mostrano la data e l'ora in cui sono state salvate.*

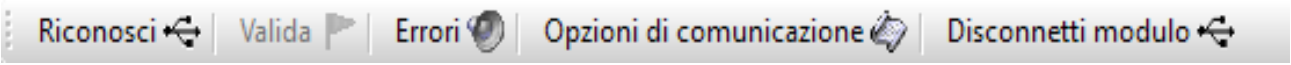
I file delle lingue si trovano nella cartella "Docs" vicino alla applicazione ThereminoHAL.exe.

Per fare nuovi file delle lingue basta copiare il file Language_ENG.txt, cambiare "ENG" con "FRA", "ESP", "DEU" o "JPN" e modificare il testo con BloccoNote.



Questo comando apre il file di documentazione.

I comandi della Toolbar



Riconosci
Serve per riconoscere i moduli Arduino connessi. I moduli vengono cercati su tutte le Porte Comm, i Baud Rate e i Nomi specificati nelle Communication Options.

Valida
Quando si aggiungono o tolgono moduli, si viene avvertiti che la configurazione è cambiata con righe rosse nella lista. Se si sceglie di perdere la vecchia configurazione e adeguarsi all'hardware attuale, con questo pulsante si rende valida la nuova configurazione.

Errori
Se premuto gli errori di comunicazione vengono evidenziati con un suono.

Opzioni di comunicazione
Viene aperto il file delle opzioni di comunicazione per modificare le porte (Com1, Com2 ecc...), le velocità (Baudrate) e i nomi dei moduli da riconoscere. Nel file delle opzioni di comunicazione sono spiegate le possibilità di scelta.

Disconnetti Modulo
Elimina il Modulo selezionato dalla lista. In questo modo si possono eliminare i Moduli indesiderati senza doverli fisicamente disconnettere dalla USB.

Applicazioni isolate

Alcune applicazioni del sistema Theremino lanciano automaticamente un proprio ArduHAL. Questo accade se esiste un file Theremino_ArduHAL.exe, nella cartella ThereminoArduHAL, situata accanto al file EXE della applicazione. Si potrebbe anche collocare il solo file Theremino_ArduHAL.exe accanto al file exe della applicazione, ma è meglio che l'ArduHAL abbia una sua cartella, con la sotto-cartella Docs contenente i file di documentazione e delle lingue.

Questi ArduHAL usano una propria configurazione privata e si connettono solo ai Moduli i cui nomi sono nel file CommOptions. Una applicazione composta in questo modo, continuerà a funzionare anche se copiata su un diverso computer e anche se altre applicazioni del sistema Theremino stanno comunicando con i propri Moduli sulla stessa rete.

Le applicazioni che più si avvantaggiano da queste possibilità, sono le applicazioni con un compito preciso, come ad esempio: Theremino Geiger, Theremino OilMeter, Theremino Meteo, Theremino Theremin, Theremino Arm, Theremino Geo e Theremino EmotionMeter.

Questo non vuol dire che le applicazioni isolate non possano comunicare con le altre. La comunicazione modulare è sempre possibile e avviene attraverso gli Slot, che sono in comune per tutte le applicazioni.

Per evitare di usare gli stessi Slot per compiti diversi abbiamo definito uno schema di massima.

```
Experimental 100 slots     000 - 099
   - - -
Theremino_Theremin         100 - 199
Theremino_SlotsToMidi      200 - 299
Theremino_MusicKeys        300 - 329
   - - -
   469 free slots          330 - 799
   - - -
Theremino_OilMeter         800 - 809
Theremino_EEG              810 - 819
Theremino_Meteo            820 - 839
Theremino_Arm              840 - 849
   10 free slots           850 - 859
   10 free slots           860 - 869
   10 free slots           870 - 879
Theremino_EmotionMeter     880 - 889
Theremino_Geiger           900 - 909
Theremino_Bridge           900 - 909
Theremino_GEO              910 - 919
Theremino_GeoPreampTester  920 - 929
Theremino_Radar            930 - 939
   10 free slots           940 - 949
   10 free slots           950 - 959
   10 free slots           960 - 969
   10 free slots           970 - 979
   10 free slots           980 - 989
   10 free slots           990 - 999
```

Questo schema è solo indicativo. Si possono usare gli Slot a piacere, basta che nello stesso PC non si usi contemporaneamente lo stesso Slot, per due compiti diversi. Se si sbaglia non si rompe nulla, ma i dati si sovrappongono con risultati indefiniti.

Regolazione delle caselle numeriche

Le caselle numeriche dell'HAL (e di tutte le altre applicazioni del sistema Theremino) sono state sviluppate da noi (Nota1) per essere più comode e flessibili delle TextBox originali di Microsoft.

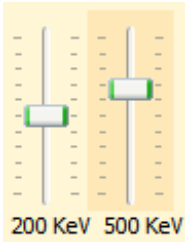
I valori numerici sono regolabili in molti modi
- Cliccando, e tenendo premuto, il bottone sinistro del mouse e muovendo il mouse su e giù
- Con la rotella del mouse
- Con i tasti freccia-su e freccia-giù della tastiera
- Con i normali metodi che si usano per scrivere numeri con la tastiera
- Con i normali metodi di selezione e di copia-incolla

- Premendo SHIFT la velocità di variazione viene moltiplicata per cento
- Premendo CTRL la velocità di variazione viene moltiplicata per dieci
- Premendo ALT la velocità di variazione viene divisa per dieci

Muovere il mouse su e giù permette ampie e veloci regolazioni
La rotella del mouse permette una regolazione comoda e immediata
I tasti freccia permettono regolazioni fini senza dover distogliere lo sguardo da ciò che si sta regolando

(Nota1) *Come tutto il nostro software i loro file sorgente sono disponibili (Freeware e OpenSource) e sono scaricabili da qui: **www.theremino.com/downloads/uncategorized** (sezione "Custom controls") Questi controlli possono essere usati a piacere in ogni progetto anche senza nominarne la fonte. I sorgenti "Open" servono anche come garanzia che non vi abbiamo incluso malware.*

Regolazione dei cursori



Questi sono i cursori originali di Microsoft, sono abbastanza comodi per cui abbiamo solo aggiunto il colore arancio e la possibilità di azzerarli.

<<< I cursori non a zero sono evidenziati con un colore arancio, per azzerarli basta fare click con il bottone destro del mouse (non tutti i cursori hanno uno zero e in tal caso non si colorano e non sono azzerabili con il mouse)

I cursori sono regolabili nei modi seguenti
- Cliccando sul cursore con il bottone destro del mouse per "azzerarli"
- Cliccando sul cursore con il bottone sinistro del mouse e muovendo il mouse su e giu
- Con la rotella del mouse
- Con i tasti freccia-sinistra e freccia-destra della tastiera
- Con i tasti freccia-su e freccia-giù della tastiera

Il metodo di muovere il mouse su e giù permette ampie e veloci regolazioni.
La rotella del mouse permette una regolazione comoda e immediata.
I tasti freccia permettono regolazioni fini senza distogliere lo sguardo da ciò che si sta regolando.
I tasti freccia sinistra/destra o su/giù hanno lo stesso effetto, ma può essere più intuitivo usare i primi per i cursori orizzontali e i secondi per i cursori verticali.

Domande e Risposte

Posso modificare il testo dei pannelli del programma nelle varie lingue?

Certamente, basta modificare i file: "..\Docs\Language_Eng.txt" e "..\Docs\Language_Ita.txt"
Per le lingue Tedesco, Francese e Spagnolo basta copiare il file inglese tre volte con i nomi seguenti: "..\Docs\Language_Deu.txt", "..\Docs\Language_Fra.txt", "..\Docs\Language_Esp.txt" e poi modificarli.

Posso editare il file di configurazione?

Normalmente la associazione tra configurazioni e moduli viene mantenuta allineata dall'HAL, che usa i nomi dei moduli per stabilire le giuste configurazioni da adottare. Normalmente l'HAL riesce a usare la giusta configurazione anche se si scollegano e sostituiscono moduli.

In alcuni casi, se si cambia nome ai moduli con un HAL che si trova su un diverso computer, o in una cartella diversa, allora l'allineamento tra configurazione e hardware si perde. In questi casi si può cliccare sulla tendina a discesa del nome del modulo e ripristinare l'allineamento scegliendo la giusta configurazione per ogni Modulo.

Per fare modifiche più complesse, si può aprire il file "Theremino_HAL_ConfigDatabase.txt" con un editor di testo come il "Notepad" e editare manualmente le configurazioni che sono abbastanza semplici.

Come ridurre il lavoro della CPU?

- Chiudere o minimizzare la finestra "Component details".
- Minimizzare la finestra principale.
- Ridurre la “Velocità”, come spiegato nelle prime pagine di questo documento.