ITIS GALILEO GALILEI ROMA

Via Conte Verde 51 00185 Roma (RM)
Segreteria didattica: 06/77071943/5
Segreteria del personale: 06/77071947
E-mail istituto: rmtf090003@istruzione.it
Codice Meccanografico Istituto: RMTF090003

GENERATORE DI FUNZIONI CON CONVERTITORE DIGITALE-ANALOGICO E MICROCONTROLLORE

31 / 05 / 2021 Emanuele Coletta

Indice

Ragioni e motivazioni dietro lo sviluppo di questo progetto....2
Ragioni, motivazioni e scopi....3
Perché un generatore di segnali digitale?....3
Perché due microcontrollori?....4
Costi....4
Il generatore....5
Direct Digital Synthesis: tecniche digitali per la generazione di segnali....5
Direct Digital Synthesis su Teensy 3.5....6
Generazione di forme d’onda arbitrarie....7
Generazione di onde quadre a duty cycle variabile....7
Svantaggi nell’uso dei DAC e filtri....8
Interfaccia utente con Atmega 32u4....8
Protocollo di comunicazione tra i due microcontrollori....9
Schemi....10
Schema a blocchi....10
Schema elettrico....11
Condizionamento del segnale con amplificatori operazionali....11
Offset e amplificazione....12
Condizionamento del range di valori....12
Problemi attuali e possibilità di espansione....12
Fonti....13
Fotografie....14

Ragioni e motivazioni dietro lo sviluppo di questo progetto

Ragioni, motivazioni e scopi

Nel marzo 2020, come misura preventiva del contaggio da Covid-19, le scuole sono state chiuse fino a fine anno scolastico. Anche durante l'anno corrente, le lezioni in presenza sono state rare e repentinamente interrotte per il ritorno alla didattica a distanza. Quest'ultima è indubbiamente stata un tratto caratterisco nelle vite degli studenti e studentesse nell'ultimo anno e mezzo.
Frequentando un istituto tecnico, maggior parte delle ore delle materie di indirizzo sono dedicate ai laboratori pratici. Con l'avvento della didattica a distanza questa possibilità è venuta meno. Anche con il ritorno in presenza, seppur parziale, l'accesso ai laboratori è stato comunque raro e complicato.
Ho così deciso di fare alcune esperienze di laboratorio da solo, a casa, aggiungendolo al normale svolgimento dei compiti. Ho personalmente la fortuna di essere riuscito a prendere in prestito da mio padre parte dei suoi strumenti da laboratorio, come multimetri, saldatori, oscilloscopio e alimentatore stabilizzato. Quello che mancava era un generatore di segnali. Parlando con amici e professori, ho scoperto l'esistenza di generatori di segnali basati su microcontrollori. Ho subito pensato potesse essere un'idea interessante, anche in vista dell'Esame di Stato, progettare e costruire da me un generatore di funzioni digitale. Ho parlato della mia idea ai professori, che l'hanno accettata e condivisa.
Anche la commissione d'esame ha poi gentilmente accettato la mia idea come argomento per questo elaborato.
Il progetto è dunque da considerarsi come un semplice studio di fattibilità, che ha senza subbio avuto esito positivo. Le possibilità di miglioramento e di espansione dello strumento sono enormi, a partire l'introduzione di un controllo completamente digitale della ampiezza e dell'offset , nonché dall'incremento della frequenza massima del segnale in uscita.
Lo scopo del progetto è primariamente quello di fornire a me stesso uno strumento di laboratorio precedentemente mancante nella mia dotazione. Questo ha insito in sè un più ampio scopo didattico. Il generatore di segnali è composto di una serie di elementi che sintetizzano l'esperienza didattica di tutto il triennio del nostro corso di studi: in primis i convertitore digitali-analogici, i filtri attivi e passivi del primo e di ordini superiori, l'uso e la programmazione dei microcontrollori, il condizionamento dei segnali e l'uso, dal più semplice al più complicato, di componenti a semiconduttori, primariamente amplificatori operazionali, poi transistor e diodi. A questo si aggiungono le tecniche digitali per la generazione di segnali, che in parte sfuggono alla programmazione del corso di automazione ma sicuramente espandono l'ambito di quanto imparato in questi tre anni.

Perché un generatore di segnali digitale?

L'avvento negli ultimi anni dei microcontrollori ha indubbiamente aperto le frontiere per una rivoluzione nel mondo dell'elettronica. Il microcontrollore è un elemento le cui azioni possono essere comandate attraverso un programma, scritto in diversi linguaggi di programmazione, compilato e caricato. Questo li rende componenti di una flessibilità incredibile, adattabili virtualmente a ogni situazione. Non si ha più un circuito che esegue una sola azione, come in passato, ma un circuito che, riprogrammando il microcontrollore, può eseguire diverse azioni, senza cambiare la configurazione hardware.
Questo è vero anche per i generatori di segnali. Un singolo microcontrollore, opportunamente programmato, collegato ad un DAC (Digital-Analog Converter / Convertitore Digitale-Analogico) può rimpiazzare e miniaturizzare quelli che prima erano una serie di circuiti distinti, che occupavano più spazio e costano di più.
Ma il fattore principale in realtà è un altro. I generatori di segnali digitali (Generatori DDS in breve) hanno

enormi vantaggi nell'ambito della frequenza: in primo luogo non soffrono la deriva di frequenza tipica dei generatori di segnali analogici, fenomeno che aumenta con l'utilizzo e il surriscaldarsi dei componenti passivi all'interno. Usura e invecchiamento dei componenti passivi nei generatori analogici, più presenti che in quelli digitali, portano anche alla distorsione del segnale in uscita fino anche alla rottura dello strumento. In secondo luogo i generatori DDS impiegano un tempo infinitesimo a passare da un segnale di una determinata frequenza ad uno di una frequenza totalmente diversa rispetto ad un generatore di segnali analogico. I generatori DDS riducono inoltre la quantità di componenti, soprattutto quelli passivi, necessari alla realizzazione degli oscillatori, rimpiazzandoli con componenti a logica sequenziale o con un microcontrollore, seguiti da un DAC. I generatori DDS inoltre, essendo digitali, possono essere riprogrammati anche da remoto, senza la necessità di cambiare la circuiteria interna al variare della frequenza necessitata in uscita. Quello del comando da remoto è un fattore sempre più importante nel campo dell'automazione industriale. Meno importante nel campo, ma comunque un gran punto di forza dei generatori DDS, è la possibilità di generare onde di frequenza minore all'Hertz e con uno sfasamento di precisione minore di un grado.

Perché due microcontrollori?

Agli albori dello sviluppo della tecnologia dei generatori DDS, il convertitore fase-tensione (vedi secondo capitolo) era implementato con componenti in logica sequenziale. Col tempo quest'ultima è stata sostituita dall'utilizzo di microcontrollori. Questo riduce l'ingrombro di spazio e la quantità di componenti utilizzati, rimpiazzando la logica sequenziale con un programma caricato sulla memoria del microcontrollore. Inoltre sostituire la memoria statica che mantiene le informazioni sulla forma d'onda da generare con un array nella memoria del microcontrollore permette di tenerne in memoria più di una e cambiare quella scelta dall'utente programmaticamente.
Per questi motivi è stato scelto di usare dei microcontrollori invece della logica sequenziale.
Il Teensy 3.5 è un potente microcontrollore Arduino-compatibile con processore ARM Cortex M4, con frequenza di clock a 120 MHz, overclockabile fino a 168 MHz, con vari "ottimizzatori" applicabili durante la compilazione del codice per incrementare la velocità di esecuzione del programma. Ogni ciclo di clock (che dura solo nove nanosecondi) di questo microcontrollore è dedicato esclusivamente alla generazione delle onde. Interrompere il ciclo della generazione di onde con altre azioni (come aggiornare il display LCD, oppure leggere i dati in input da altre fonti come l'encoder) porterebbe inevitabilmente all'instabilità della frequenza dell'onda in uscita e talvolta alla deformazione della stessa.
Per evitare questo problema Teensy 3.5 viene affiancato da un Atmega 32u4, che si occupa esclusivamente dell'interfacciamento con l'utente (attraverso l'encoder), della lettura delle informazioni (dopo il condizionamento) sul segnale e mostrare il tutto su un display LCD. Questo toglie carico computazionale al Teensy 3.5, permettendo l'effettiva stabilità in frequenza del segnale generato, evitando deformazioni o deriva.
La scelta di questa coppia di microcontrollori deriva da una mia vasta esperienza nel loro utilizzo nell'ambito del laboratorio di robotica. Inoltre, all'interno del laboratorio di TPSEE, ogni studente della classe ha sviluppato una scheda microcontrollore "general purpose" basata su Atmega 32u4.
Il Teensy 3.5 non è tuttavia il microcontrollore più adatto per questo lavoro, ma ad un modico prezzo offre ben due DAC e un'elevata frequenza di clock. Il circuito stampato è stato comunque progettato ed assemblato per poter sostituire facilmente il Teensy 3.5 con un altro microcontrollore, con DAC integrato o esterno, previa progettazione di una scheda di adattamento che si colleghi ai pin di alimentazioni e interfacciamento con gli altri elementi (azzeramento offset e filtri, collegamento seriale con l'Atmega 32u4) già presenti ed esposti sulla scheda, a cui normalmente è collegato il Teensy.

Costi

I componenti scelti per questo progetto sono componenti di tipo consumer, che rendono il progetto facilmente accessibile e replicabile da chiunque. Di seguito segue una tabella dei principali componenti utilizzati. L'effettivo importo speso per la realizzazione di questo progetto è leggermente più alto, in quanto sono stati scelti rivenditori che potessero offrire una spedizione molto rapida (il Teensy 3.5 per esempio viene spedito dagli Stati Uniti, e scegliendo la spedizione più economica i tempi di spedizione sarebbero arrivati fino a due mesi). Al totale dei componenti e della board si devono aggiungere 9€, costo di spedizione fisso dal sito tme.eu dove sono stati reperiti i componenti. Inoltre, sul questo store online, è possibile l'acquisto di una quantità minima di 100 componenti, quando si tratta di resistenze e condensatori.
Le PCB sono state fatte stampare a Shenzen, in Cina, dall'azienda JLCPCB. L'azienda ha un offerta per la quale le PCB che hanno entrambi i lati più corti di 10cm costano soltanto 2€. La scheda viene stampata in 5 copie. A questi si deve aggiungere un costo minimo di 5€ di spedizione col corriere ePacket.

Componente	Costo/unità [€]	Quantità	Costo totale [€]
Teensy 3.5	20	1	20
OpAmp AD8034	3.50	6	21
Encoder incrementale	3	1	3
Connettore bnc	1.5	1	1.5
Relè di segnale	1.1	2	2.2
Atmega 32u4	3	1	3
Componentistica varia*	\	\	8
Realizzazione PCB	0.40	5	2
Totale	\	\	63.70
Totale con spedizioni (costi minimi possibili)	\	\	79.70

*(resistenze, condensatori, strip, oscillatori al quarzo, potenziometri e trimmer, transistor, diodi)

I limiti minimi di acquisto di TME, l'acquisto di qualche componente di ricambio in caso qualcosa fosse andato storto durante le prove e la scelta dei servizi di spedizione più veloce ha fatto aumentare leggermente il prezzo finale. Tuttavia, se si dovesse scegliere di produrre in massa questo generatore di segnali, i problemi relativi alle spedizioni e all'acquisto massivo di componenti sarebbero facilmente ovviati. Io personalmente non considero l'acquisto di componenti in eccesso una spesa, ma bensì un investimento che allevierà il costo dello sviluppo di progetti futuri.

Il generatore

Direct Digital Synthesis: tecniche digitali per la generazione di segnali

La sintesi diretta digitale (Direct Digital Synthesis in inglese, abbreviato in DDS) è una tecnica per la generazione di segnali che si basa su componenti digitali invece che analogiche. Il segnale viene campionato in un numero *N* di punti, il cui valore di tensione corrispondente viene immagazzinato in una *tavola di lookup*, che sostanzialmente è una memoria ROM. In base alla frequenza che si vuole generare, si utilizzeranno più o meno campionamenti dell'onda in un singolo periodo. Frequenza e campionamenti usati sono inversamente proporzionali. All'aumentare della frequenza, i campionamenti usati in un singolo periodo diminuiscono. Un normale generatore DDS si compone principalmente di tre blocchi hardware:
Il primo è un clock di sample. Dà la frequenza di clock a tutto il circuito per decidere "il ritmo" delle operazioni. La frequenza del segnale in uscita non è però vincolato a questa frequenza.
Il secondo è l'accumulatore di fase. È il blocco che si occupa di scegliere quali e quanti campioni utilizzare. A sua volta si costruisce di due componenti. Un addizionatore e una parola digitale la cui parte intera indica il numero corrispondente al campione da utilizzare nella *tavola di lookup*. Ogni ciclo di clock l'addizionatore aggiunge la parola ad un registro. Quando questo eccederà le dimensioni massime andando in overflow, viene fatto tornare a zero. Insieme sono sostanzialmente un contatore *modulo N*, dove *N* è il numero di elementi presenti nella tavola.
La tavola di lookup, ultimo stadio prima del DAC, viene usato come convertitore fase-ampiezza (spesso definito anche come convertitore fase-tensione). Ogni ciclo di clock, il punto all'interno della *tavola di lookup* corrispondente al valore dell'accumulatore di fase, viene convertito in valore analogico dal DAC. Proseguendo questo procedimento più volte nel tempo, si modula l'uscita del DAC in modo che formi un segnale con andamento sinusoidale nel tempo. Ovviamente la *tavola di lookup* può essere impostata per immagazzinare un segnale di qualsiasi forma, non necessariamente una sinusoide.
In epoca moderna, questi differenti blocchi hardware possono essere facilmente sostituiti da blocchi di codice, e caricati su un microcontrollore. La *tavola di lookup* diventa un *array* calcolato in *runtime*. L'accumulatore di fase diventa una variabile a cui viene addizionato un valore, il contatore *modulo N* sostituito da un controllo *if*. Per la natura intrinseca dei DAC tuttavia è necessario l'uso di un filtro passa-basso che renda il segnale in uscita continuo e non "a gradini" come tipicamente derivamente dall'uso di un DAC.

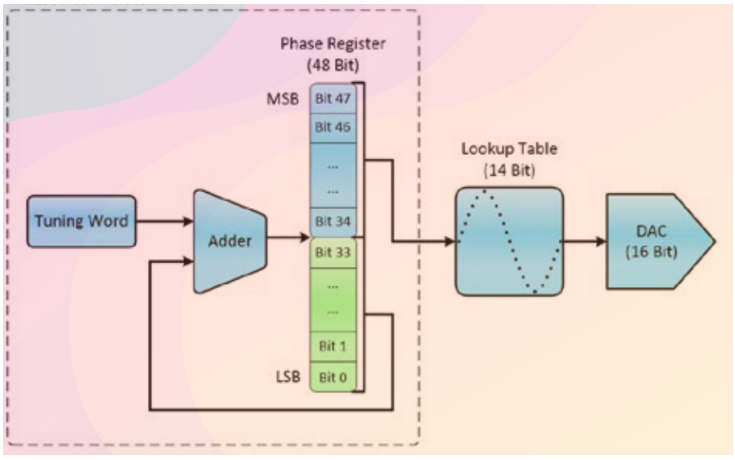


Diagramma dei blocchi hardware di un generatore DDS. Immagine di National Instruments

Direct Digital Synthesis su Teensy 3.5

È stato deciso di programmare il Teensy 3.5 in linguaggio C++, con l'utilizzo del framework Arduino, mediante l'uso dell'IDE Visual Studio Code con plugin PlatformIO. Al contrario di un normale sketch Arduino, che usa una versione abbastanza semplificata del linguaggui C++, l'uso di VSCode e PlatformIO permette l'uso di un C++ standard a cui vengono aggiunte le funzionalità del framework Arduino.

Generazione di forme d'onda arbitrarie

L'implementazione della tecnica DDS è avvenuta nella maniera più semplice possibile: all'avvio del microcontrollore esso calcola la *tavola di lookup* per diverse onda utilizzandone la formula analitica. Quando possibile, ne viene calcolata solo una parte (il primo quarto per le onde sinusoidali, la metà per le onde triangolari) e il resto viene derivato per simmetria. Così facendo, oltre ad avere certezza della simmetria, si riducono anche i tempi di calcolo.
L'accumulatore di fase è implementata come una variabile di tipo float con precisione a 6 cifre decimali. Entrando in un ciclo infinito, viene preso l'indice della *tavola di lookup* corrispondente alla parte intera dell'*accumulatore di fase* e messo in uscita dal DAC. All'inizio del ciclo, l'*accumulatore di fase* viene incrementato di una quantità $\Delta\varphi$ pari a:

$$\Delta\phi = \frac{f}{f_S} G$$

Dove f è la frequenza che si vuole avere in uscita, f_S è la frequeza di campionamento del DAC che si sta utilizzando e G è la risoluzione (grandezza) della *tavola di lookup*. Nel caso il risultato superasse la grandezza della tavola di lookup, essa gli viene sottratta per implementare il contatore modulo N ed evitare overflow.
La stessa tecnica può essere implementata per calcolare altre forme d'onda, come la triangolare, dente di sega o la quadra a duty cycle variabile, semplicemente cambiando la formula analitica usata. Il resto rimane invariato.
L'uso delle funzioni già presenti nel framework Arduino per comandare i DAC risultano avere un'esecuzione abbastanza lenta, richiedendo fino a più di 50 cicli di clock (540 nanosecondi). Queste funzioni eseguono ad ogni chiamata istruzioni che hanno necessità di essere eseguite una sola volta. Per velocizzare l'esecuzione, dunque, sono state prese solamente le istruzioni necessarie eseguite da queste funzioni, che vengono eseguite direttamente nel ciclo di generazione senza bisogno di chiamare altre funzioni presenti nel framework Arduino.

Generazione di onde quadre a duty cycle variabile

Le onde quadre possono essere generate anche utilizzando un pin digitale del microcontrollore e portarlo a livello alto o basso nei tempi giusti. A tale fine, è stato necessario abilitare il registro del Teensy che conta i cicli di clock passati dall'accensione del microcontrollore. Sapendo che ogni ciclo di clock dura approssivamente nove nanosecondi, è possibile derivare per quanti cicli di clock è necessario che il pin digitale venga mantenuto al livello necessario. Ad esempio, per generare un'onda quadra di frequenza 10kHz e duty cycle al 60%, è necessario prima sapere la durata del periodo in nanosecondi:

$$T(ns) = (\frac{1}{f}) 10^9 = \frac{1}{10^4} * 10^9 = 10^{-4} * 10^9 = 10^5 nS$$

Da cui poi si deriva che il livello alto (60% del periodo) dura:

$$H = \frac{60}{100} * T = \frac{3}{5} * 10^5 = 60000\,nS = 60\,mS$$

e il livello basso (40% del periodo) dura:

$$L = \frac{40}{100} * T = \frac{2}{5} * T = T - H = 40000\,nS = 40\,mS$$

È necessario eseguire questo calcolo una sola volta, prima dell'inizio della generazione dell'onda, dopodiché è necessario impostare il pin digitale a livello alto e basso per la durata calcolata.

Svantaggi nell'uso dei DAC e filtri

Esistono vari tipi di DAC. I più comuni sono però tre: quelli a resistenze pesate, quelli PWM e quelli con scala di resistori R-2R.
A prescindere dalla tipologia, è una caratteristica intrinseca dei DAC fornire un'uscita "a gradini".
Derivando da un segnale digitale, un DAC può fornire solamente una quantità discreta e finita di valori in uscita. Ogni gradino è chiamato un "quanto", l'ampiezza di ogni quanto dipende dalla precisione in bit del DAC e dalla tensione di fondo scala utilizzata. Per mascherare la caratteristica visibile dei quanti, ai DAC vengono spesso fatti seguire dei filtri passa-basso, che hanno anche il compito di eliminare rumore e spurie dal segnale in uscita. Per i generatori DDS, si usano spesso filtri attivi di ordini superiori al secondo. Seguendo le fonti riportate in fondo al testo, è stato deciso di utilizzare un filtro Butterworth del sesto ordine per i segnali sinusoidali e le altre forme d'onda arbitrarie.
Per le onde quadre era stato inizialmente deciso di utilizzare, per gli elevati tempi di risposta forniti, un filtro attivo Bessel del sesto ordine per eliminare le oscillazioni presenti sul transitorio del segnale. Tuttavia, simulandolo in ambiente MultiSim, risultava che esso ritardasse troppo la salita o la discesa del transitorio, facendo diventare il segnale quadro simile alla caratteristica di carica e scarica di un condensatore. Per le onde quadre è stato dunque deciso di utilizzare un semplice filtro RC passa-basso del primo ordine, che esegue perfettamente il lavoro. È inoltre predisposto nello schematico e sulla PCB un OpAmp usato come Trigger di Schmitt per filtrare ulteriormente il transitorio, ma al momento non è utilizzato.

Interfaccia utente con Atmega 32u4

L'intefacciamento del generatore con l'utente è affidata ad un microcontrollore secondario, un Atmega 32u4. Collegato al 32u4, un encoder incrementale con pulsante. L'interfaccia viene messa a video su un display LCD 16x2, che comuncia con il 32u4 attraverso il protocollo I2C.
La scelta dell'Atmega 32u4 deriva dalla mia precedente esperienza nel laboratorio di robotica con questo microcontrollore, e dall'esperienza in orario curriculare nel laboratorio di TPSEE.
Esistono svariate schede Arduino basate su Atmega 32u4, come il Micro o il Leonardo, il che permette di utilizzare agilmente il framework Arduino con questo microcontrollore.
Il display ha integrato una scheda di adattamento con un integrato che comunica in I2C con il 32u4 e gestisce il pannello. Questo facilita enormemente lo sbroglio del circuito a fronte di una modica spesa, dovendo collegare solamente quattro piste (due dati e due alimentazione) invece che le sedici normalmente richieste da un pannello LCD di questo tipo.
L'interfaccia grafica si fa un di un menù principale, nella cui riga superiore vengono alternativamente mostrati il tipo e il duty cycle scelto, oppure la frequenza, con un intervallo di due secondi.

Nella riga inferiore vengono mostrate la tensione di picco e la tensione di offset del segnale in uscita, ottenute tramite un circuito di condizionamento che riporta i segnali nel range [0V,5V] leggibili dagli ADC integrati del 32u4.
Premendo una volta il pulsante dell'encoder si accede al menu di *edit*. Viene prima chiesto di decidere il tipo di segnale (sinusoidale, quadro, triangolare…) e successivamente, se si è scelto un segnale quadro, è possibile deciderne il duty cycle.
Si passa successivamente all'impostazione della frequenza del segnale. Con una frequenza massima di 1MHz, è possibile modificare ognuna delle sette cifre del numero. Non è dunque necessario un frequenzimetro come nei generatori di segnali analogici, poiché la tecnica DDS permette di generare segnali che rispettino accuratamente la frequenza decisa a priori.
Una volta decisa la frequenza del segnale si torna alla schermata principale. Le informazioni vengono mandate al Teensy, che si occuperà della generazione del segnale con le caratteristiche richieste.

Protocollo di comunicazione tra i due microcontrollori

Ogni singolo ciclo di clock del Teensy è dedicato alla generazione del segnale. Questo significa che il Teensy non può permettersi di interrompere periodicamente questo compito per controllare se sono state ricevute informazioni dal 32u4.
Il protocollo di comunicazione è basato sull'invio di una stringa dal 32u4 al Teensy, che contiene tutte le informazioni impostate nel menu di *edit*.
La stringa ha un carattere alfabetico 'w' che segna l'inizio del messaggio e uno 'W' che segna la fine. Tra questi, ogni informazione numerica è inserita tra due caratteri alfabetici identificativi.

Ad esempio, se si dovesse generare un segnale sinusoidale di frequenza 10kHz si avrebbe la seguente stringa:

wt0Tf10000FW

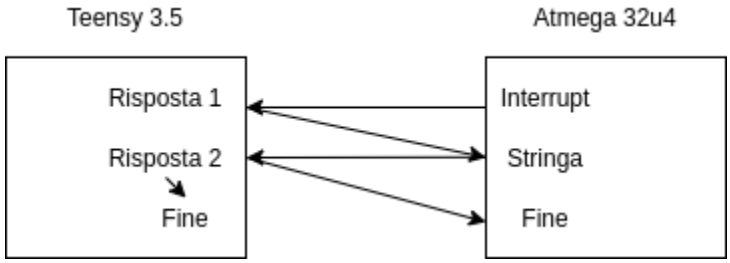
Dove 'w' è il carattere di inizio e 'W' quello di fine. I caratteri 't' e 'T' indicano che al loro interno è contenuto il tipo di segnale da generare, il numero 0 indica un segnale sinusoidale.
Tra i caratteri 'f' e 'F' è contenuta la frequenza del segnale. In questo caso, come deciso, 10000Hz, o 10kHz.
Se invece si dovesse generare un segnale quadro di frequenza 14050Hz e duty cycle 75%, verrebbe generata la stringa:

wt1Td75Df14050FW

Dove, di nuovo, 'w' è il carattere di inzio e 'W' quello di fine. I caratteri 't' e 'T' indicano che al loro interno è contenuto il tipo di segnale da generare. Il numero 1 indica un segnale quadro. Tra i caratteri 'f' e 'F' è contenuta la frequenza del segnale. In questo caso, come deciso, 14050Hz, o 14.05kHz. In aggiunta, tra i caratteri 'd' e 'D' è contenuta la percentuale del duty cycle, in questo caso 75.

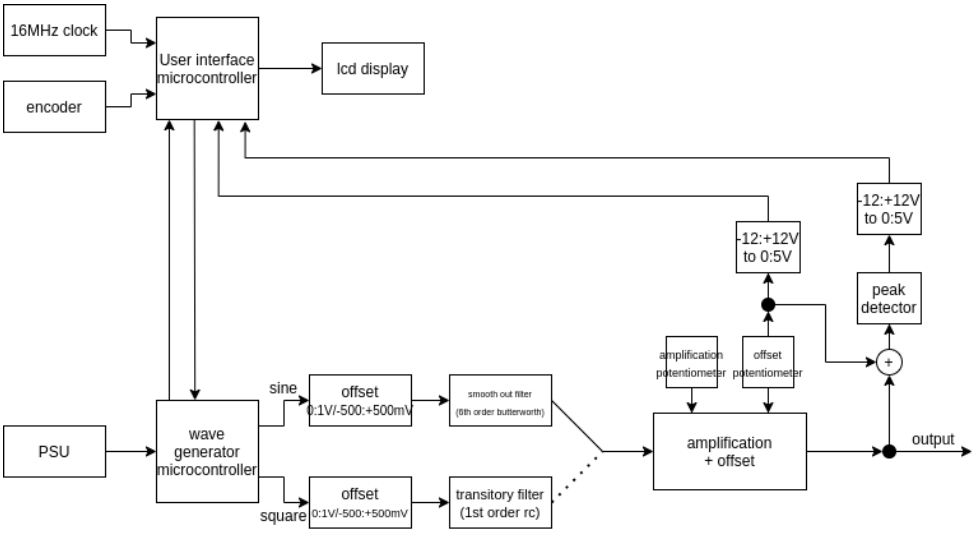
Alla ricezione della stringa, il Teensy ottiene le informazioni necessarie tramite i metodi dell'oggetto String presente nel framework Arduino, che permette di ricavare le sottostringhe presenti tra diversi caratteri senza l'utilizzo di strutture dati aggiuntive come array o altri contatori.
La comunicazione tra i due microcontrollori inizia dunque con l'invio di un interrupt (sul fronte di salita) al Teensy da parte del 32u4. Il 32u4 continua a generare questo interrupt fino alla ricezione di una risposta da parte del Teensy. Quest'ultimo, appena ricevuto l'interrupt, interrompe la generazione del segnale per mettersi in attesa dell'arrivo della stringa. Di nuovo, la stringa viene inviata ripetutamente a intervalli di tempo regolari fino all'arrivo di una risposta positiva da parte del Teensy. A questo punto la comunicazione viene interrotta e i buffer

delle linee seriali di entrambi i microcontrollori vengono svuotati per evitare che il loro contenuto possa interferire con le comunicazioni successive.

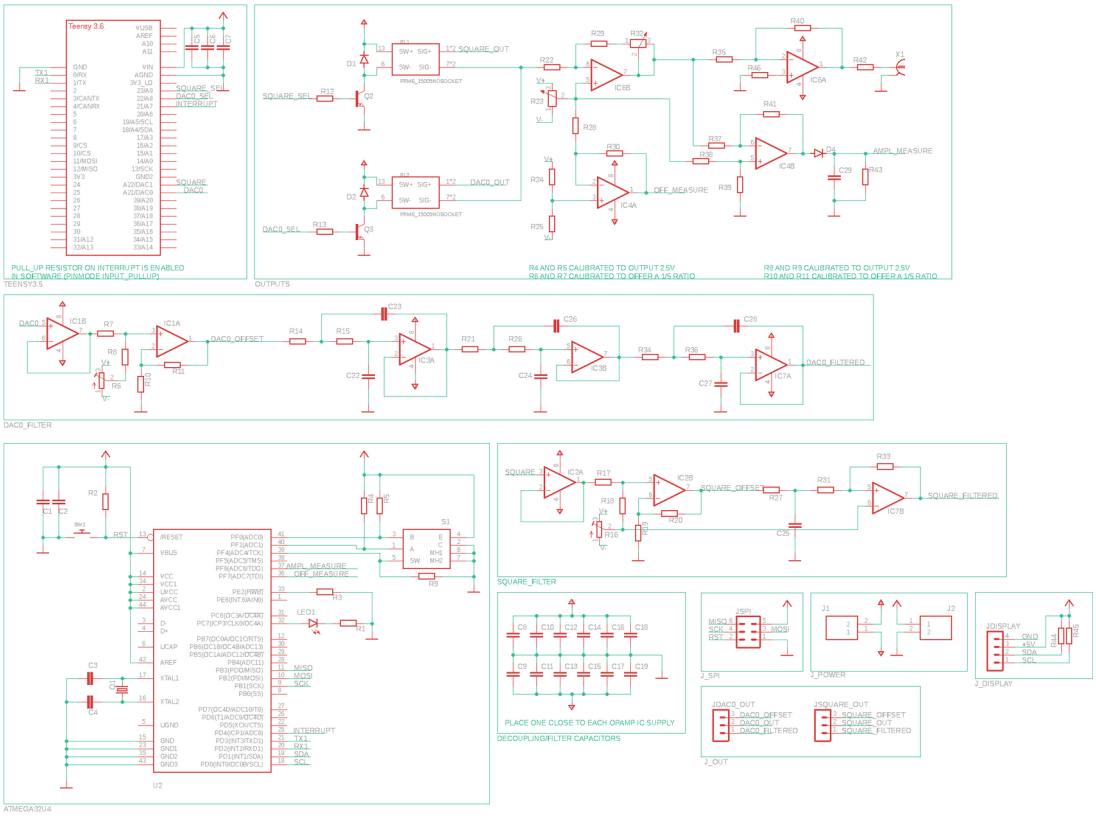


Schemi

Schema a blocchi



Schema elettrico



Condizionamento del segnale con amplificatori operazionali

Offset e amplificazione

Dopo lo stadio di generazione e filtro, il segnale necessita di essere amplificato ed eventualmente che gli venga applicato un offset. Per questa funzione è stato deciso di utilizzare un amplificatore operazionale in configurazione invertente, sul cui ramo di retroazione sono presenti una resistenza da 100Ω e un potenziometro da 10kΩ. La presenza della prima resistenza sul ramo di retroazione fa sì che il segnale non venga mai annullato, ma che possa essere anche attenuato e non solo amplificato. La presenza del potenziometro invece permette un amplificazione variabile.
L'offset a sua volta è implementato mediante l'uso di un potenziometro utilizzato come partitore di tensione tra [-12V,+12V], collegato all'ingresso non invertente dello stesso operazionale che si occupa dell'amplificazione. Alcuni integrati espongono un pin dedicato propriamente alla gestione dell'offset, ma gli AD8034 scelti per questo progetto non li posseggono. Ogni AD8034 contiene due diversi amplificatori operazionali. Nell'ultimo stadio, uno di questi viene utilizzato per amplificazione offset, l'altro viene utilizzato in configurazione invertente a guadango unitario per annullare l'effetto dell'inversione introdotto nello stadio di amplificazione. Se quest'ultimo stadio non ci fosse, i segnali avrebbero duty cycle invertito. È da notare come l'uso di amplificatori operazionali permetta di disaccoppiamento delle impedenze, di fatto impedendo ai circuiti di misura o a qualsiasi carico attaccato al generatore di interferire con lo stadio di uscita.

Condizionamento del range di valori

Non essendo presente un controllo digitale dell'amplificazione dell'offset, ma essendo questi ultimi controllati tramite due potenziometri, deve essere inserito un modo per l'utente di sapere queste due caratteristiche del segnale. È stato scelto di ottenere queste caratteristiche dal segnale stesso e leggerle tramite gli ADC integrati nel 32u4 per essere mostrate sul display LCD
Il potenziometro che regola l'offset può assumere un valore che rientra in [-12V, +12V]. Questi valori sono fuori dalle tolleranze degli ADC del 32u4 e necessitano di essere condizionate in un range [0V, +5V] Per farlo è stato usato un amplificatore operazionale in configurazione invertente, che attenuasse il segnale per portarlo nel range [-2.5V, +2.5V], scegliendo dunque un rapporto di resistori di circa 1/5. Al piedino non invertente di questo operazionale è collegato un partitore di tensione, che offre una tensione di 2.5V in uscita, riportando di fatto il valore di offset nel range [0V, +5V].
Discorso diverso è invece ottenere l'ampiezza dell'onda. Dato che al segnale in uscita può essere applicato un offset, l'uso di un semplice rilevatore di picco non è sufficiente, poiché risentirebbe dell'effetto dell'offset. Per annullare quest'effetto, al segnale di uscita viene sottratto il valore dell'offset tramite un amplificatore operazionale in configurazione amplificatore di differenze, e al risultato viene applicato un rilevatore di picco. L'equazione di uscita di questa configurazione permette inoltre di applicare direttamente un fattore di scala al risultato della sottrazione. Scegliendo un fattore di scala di 5/12, il valore viene riportato nel range [0V, +5V].

Problemi attuali e possibilità di espansione

Allo stadio attuale, il generatore non possiede grandi problemi che ne prescindono il corretto funzionamento o utilizzo. Quello che potrebbe essere considerato un impedimento è la frequenza massima raggiungibile dal segnale generato con i DAC integrati del Teensy. Al momento, dopo una serie di ottimizzazioni lato software e overclock del microcontrollore, è di 1 MHz. Le onde quadre generate con i pin digitali del microcontrollore tuttavia riescono a toccare anche i 2 MHz. Il Teensy risulta dunque non essere esattamente lo strumento più adatto al lavoro richiesto, ed inoltre ha un costo abbastanza consistevole se si tiene conto delle "scarse" performance e dell'ingrombro di spazio di richiede. Al fine di fare esperimenti e prove con altri microcontrollori, la PCB è stata predisposta, in fase di montaggio, con dei connettori strip che permettono lo smontaggio del Teensy dalla scheda. Questo apre la possibilità alla progettazione di altre schede, che utilizzando i pin predisposti per il Teensy 3.5, adattino la PCB già esistente all'uso con altri microcontrollori con DAC interni o esterni. Sarebbe così possibile esplorare un abbassamento dei costi utilizzando il nuovissimo Raspberry Pi Pico con un DAC esterno a basso costo. Oppure, al fronte di un leggero incremento dei costi, sarebbe interessante provare l'implementazione di un DAC esterno come l'AD7542, che, stando al datasheet, promette tempi di risposta incredibilmente bassi e quindi una frequenza massima del segnale di uscita molto più elevata. Oppure cambiando totalmente famiglia di microcontrollori, sarebbe possibile sperimentare l'uso di microcontrollori della famiglia STM, basati su architettura ARM Cortex, che offrirebbe inoltre l'occasione didattica per l'utilizzo di microcontrollori diversi da quelli della famiglia Atmega dato che richiedono un framework diverso da quello Arduino (molti STM usano il framework mbed, anche se supportano quello Arduino. Altri non supportano nessuno dei due e richiedono l'utilizzo diretto dei registri del microcontrollore).
Inoltre, per avvicinare il progetto più ad un prodotto finito che ad uno palesemente hobbystico ed autocostruito, si potrebbero utilizzare dei resistori comandati in tensione o dei resistori programmabili per avere un controllo totalmente digitale di amplificazione offset, senza bisogno di circuiti di feedback che ricavino le caratteristiche del segnale dal segnale stesso. Implementare l'uso di un display OLED invece che un LCD 16x2 inoltre permetterebbbe un'interfaccia grafica più complessa.
Dopo questi sviluppi, arrivati a una soluzione soddisfacente e che offra un buon rapporto qualità/prezzo sarebbe possibile iniziare a pensare ad una soluzione modulare, che implementi una singola base (con un microcontrollore che abbia lo stesso compito dell'Atmega 32u4 nel generatore attuale) a cui modularmente vengano attaccati altre schede che si occupano esclusivamente della generazione di segnali, aprendo la possibilità all'uso di diversi canali liberamente aggiungibili e removibili.
Nonostante io sia un grande ammiratore della filosofia open-source, in un futuro vorrei esplorare la possibilità di mettere in commercio questo progetto. Per coloro che dovessero essere capaci, il progetto è disponibile open-source sul mio profilo GitHub sotto licenza GPL3. La licenza permette l'uso di questi file a scopi di lucro, e sarebbe interessante mettere in commercio il progetto sotto forma di kit già montato o da montare, con o senza chassis o circuito di alimentazione. Questo lascia la possibilità ai capaci di usare liberamente i file, ai meno capaci di prendere comunque vantaggio del progetto ad un modico costo.

Fonti

- Teoria della tecnica DDS:
  - https://www.analog.com/en/analog-dialogue/articles/all-about-direct-digital-synthesis.html
  - https://https://www.analog.com/media/en/training-seminars/design-handbooks/Technical-Tutorial-DDS/Section8.pdf
  - https://www.ni.com/it-it/innovations/white-papers/06/understanding-direct-digital-synthesis--dds-.html
  - https://en.wikipedia.org/wiki/Direct_digital_synthesis
  - http://lib.tkk.fi/Diss/2000/isbn9512253186/isbn9512253186.pdf

- Pratica della DDS:
  - https://zipcpu.com/dsp/2017/08/26/quarterwave.html
  - https://zipcpu.com/dsp/2017/07/11/simplest-sinewave-generator.html
  - https://stackoverflow.com/questions/13466623/how-to-look-up-sine-of-different-frequencies-from-a-fixed-sized-lookup-table
  - https://stackoverflow.com/questions/16889426/fm-synthesis-using-phase-accumulator

Tutto il progetto (i file CAD sia per il case che per la PCB, il codice e la documentazione, incluso questo elaborato) è disponibile open source sotto licenza GPLv3 sul GitHub: https://github.com/EmaMaker/SignalGenerator-Teensy

Fotografie