1. INTRODUZIONE E OBIETTIVI DEL PROGETTO

1.1 Obiettivi del progetto e sua realizzazione

L'azionamento con motore sincrono a magneti permanenti, per attuare il controllo di corrente (coppia), necessita della conoscenza della posizione assoluta del rotore. Tale angolo costituisce la posizione dell'asse diretto del sistema di riferimento al quale sono riferite le correnti da imporre allo statore per avere la desiderata coppia.

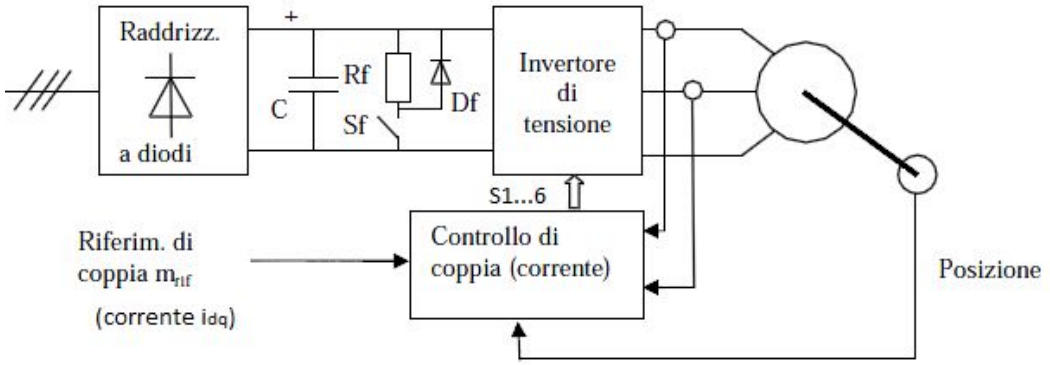


Illustrazione 1: Schema generale di un azionamento in alternata con controllo di coppia

Questa è una caratteristica degli azionamenti con macchina sincrona e consente di ottenere prestazioni dinamiche altrimenti non possibili, grazie all'accurato controllo vettoriale di corrente che si può realizzare. Al tempo stesso però la necessità di un **trasduttore di coppia assoluta** costituisce uno svantaggio in termini di costo, ingombro, affidabilità; la sua eliminazione ricorrendo all'utilizzo di una coppia stimata invece che misurata (*azionamenti sensorless*) ha assunto un interesse pratico crescente e stimola lo studio di soluzioni anche innovative ed avanzate.

Per poter realizzare questa stima si ricorre a degli algoritmi avanzati che necessitano, tra le altre, della **misura real-time delle reali tensioni concatenate che alimentano il motore**.

In particolare si deve misurare, ad ogni periodo di modulazione PWM, la **componente fondamentale** delle tensioni concatenate del motore. La misura deve essere il più precisa possibile per una corretta stima di coppia, e quindi può essere fatta solamente per via analogica, o con un sistema digitale che possa campionare i segnali acquisiti ad altissime frequenze, cosa che non è possibile con il microcontrollore usato nell'azionamento.

L'obiettivo principale di questo progetto è quello di poter misurare e rendere disponibile al sistema di controllo dell'azionamento, ad ogni periodo di PWM, le reali tensioni concatenate che arrivano al motore, in particolare la loro componente fondamentale, poiché esse vengono prelevate in uscita dall'inverter trifase che genera delle onde quadre modulate tra un'alimentazione positiva e una negativa. Nella figura 2 sono illustrate la portante del modulatore PWM, le modulanti v_1*, v_2* e v_3* e le uscite del modulatore v_1, v_2 e v_3, con evidenziate le loro componenti fondamentali $v_1$med, $v_2$med, $v_3$med.

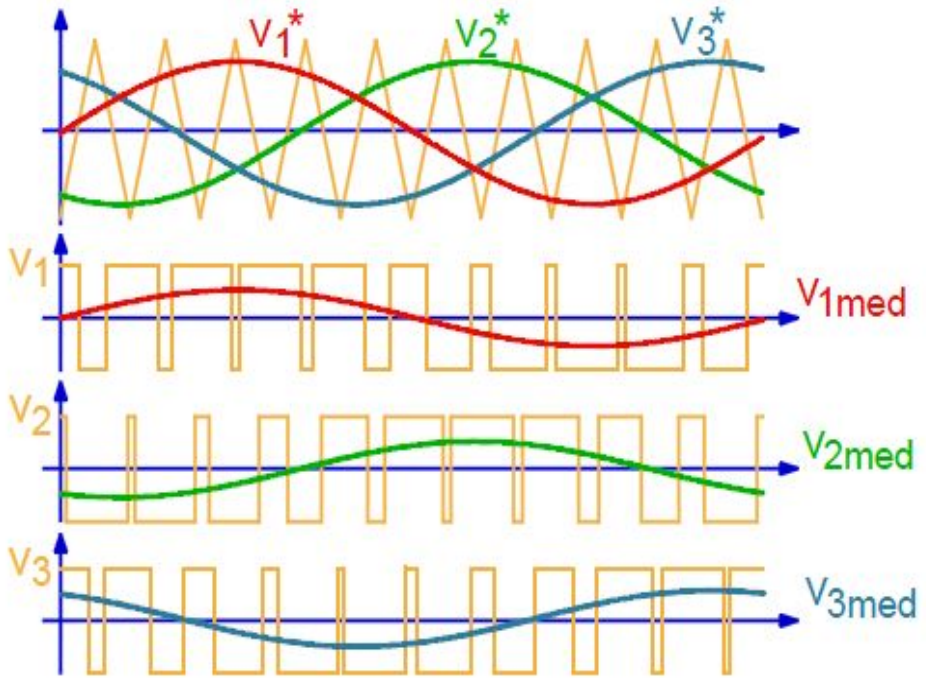


Illustrazione 2: v_1, v_2, v_3 = tensioni di fase generate dall'inverter (arancioni)

Nell'illustrazione 3 si possono vedere due tensioni di fase $v_a(t)$ e $v_b(t)$ e la tensione concatenata $v_{ab}(t)$ ricavata da esse.

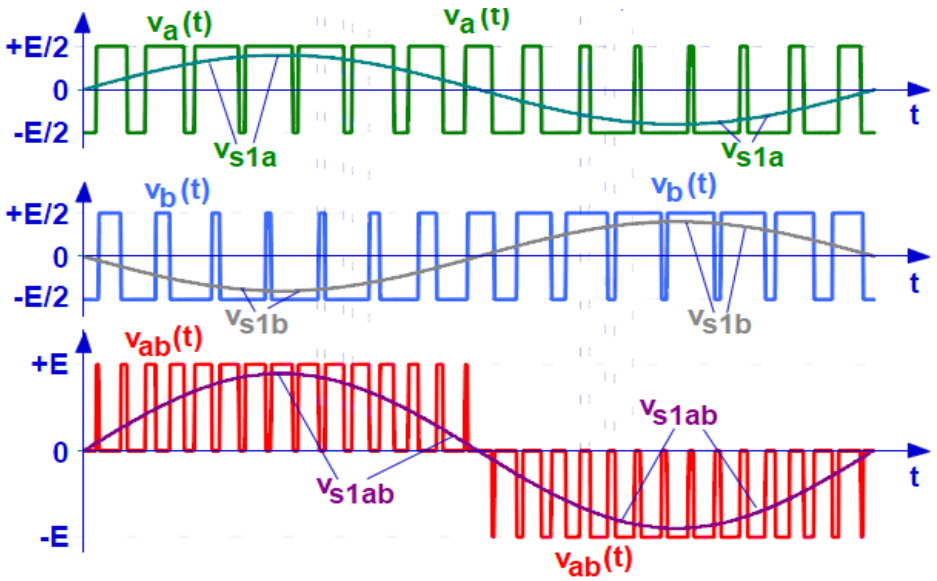


Illustrazione 3: Esempio di tensione concatenata generata dall'inverter (grafico rosso) e suo valore medio (grafico viola)

Nella prima fase di questo lavoro sperimentale si studia una scheda analogica costruita e progettata dalla ditta REEL s.r.l che acquisisce le tensioni di fase del motore e calcola la media delle concatenate, restituendo in uscita un segnale analogico in corrente proporzionale ad esso (vedi illustrazione 3).

Questo circuito deve essere sincronizzato con il resto dell'azionamento, che in questo caso è realizzato attraverso un ambiente di sviluppo integrato **FPCS** (*Fast Control Prototyping System*).

Il sistema di misura delle tensioni del motore deve conoscere, attraverso un segnale di sincronismo che arriva dal sistema di controllo dell'azionamento, l'istante in cui inizia il periodo di PWM e quando comincia il periodo successivo, per determinare l'intervallo di tempo su cui viene calcolata la media delle concatenate.

La scheda di misura analogica è formata da un **doppio integratore** controllato dai segnali di sincronismo generati dallo stadio precedente e che restituisce in uscita il valore medio desiderato.

Successivamente si sono fatte delle misure delle uscite del sistema così ottenuto per un motore sincrono a magneti permanenti, azionato dal FPCS, ottenendo un segnale analogico proporzionale alla media delle tensioni concatenate. Questi valori vengono poi analizzati e confrontati con le uscite di un altro sistema che ha

gli stessi obiettivi del precedente, ma è stato realizzato per via digitale attraverso l'uso di una **FPGA**.

Si cerca di utilizzare un sistema digitale principalmente per motivi di flessibilità, di maggior semplicità nella fase di realizzazione e minore probabilità di guasti o difetti della scheda.

Inoltre l'uso di tecnologie basate su FPGA in ambito industriale è un campo sempre in maggiore espansione e che merita di essere sviluppato, visto i vantaggi che possono offrire questi dispositivi rispetto ai normali microcontrollori o DSP, soprattutto in termini di velocità di calcolo e campionatura dei segnali.

Per realizzare questa soluzione digitale è stata utilizzata la scheda ***CYCLONE III FPGA STARTER BOARD*** che monta una FPGA CYCLONE III della Altera e un'altra scheda, realizzata dalla Terasic S.p.A, azienda partner dell'Altera per lo sviluppo di sistemi elettronici legati all'utilizzo delle loro FPGA, che realizza la conversione analogico/digitale per acquisire le tensioni di fase del motore e la conversione digitale/analogico per restituire la componente fondamentale delle tensioni concatenate e poterle analizzare con un'oscilloscopio.

Sono state effettuate alcune modifiche a quest'ultimo circuito per poterlo adattare alle esigenze di questo particolare progetto ed è stata realizzata un'ulteriore scheda per interfacciare le tensioni di fase del motore in uscita dall'inverter del FPCS con i convertitori A/D.

Il software per programmare l'FPGA è stato realizzato attraverso l'ambiente di sviluppo ***QUARTUS II***, con i linguaggi di programmazione Verilog, VHDL e il tool grafico della Altera.

Una volta connesso questo sistema digitale al resto dell'azionamento si sono ottenuti dei risultati confrontabili con la soluzione analogica e che potranno essere utilizzati dal sistema di controllo per stimare, ad ogni periodo di modulazione PWM, la coppia del motore in alternata.

1.2 Schema generale del sistema

La connessione dell'intero sistema può essere schematizzata come illustrato nella figura 4:

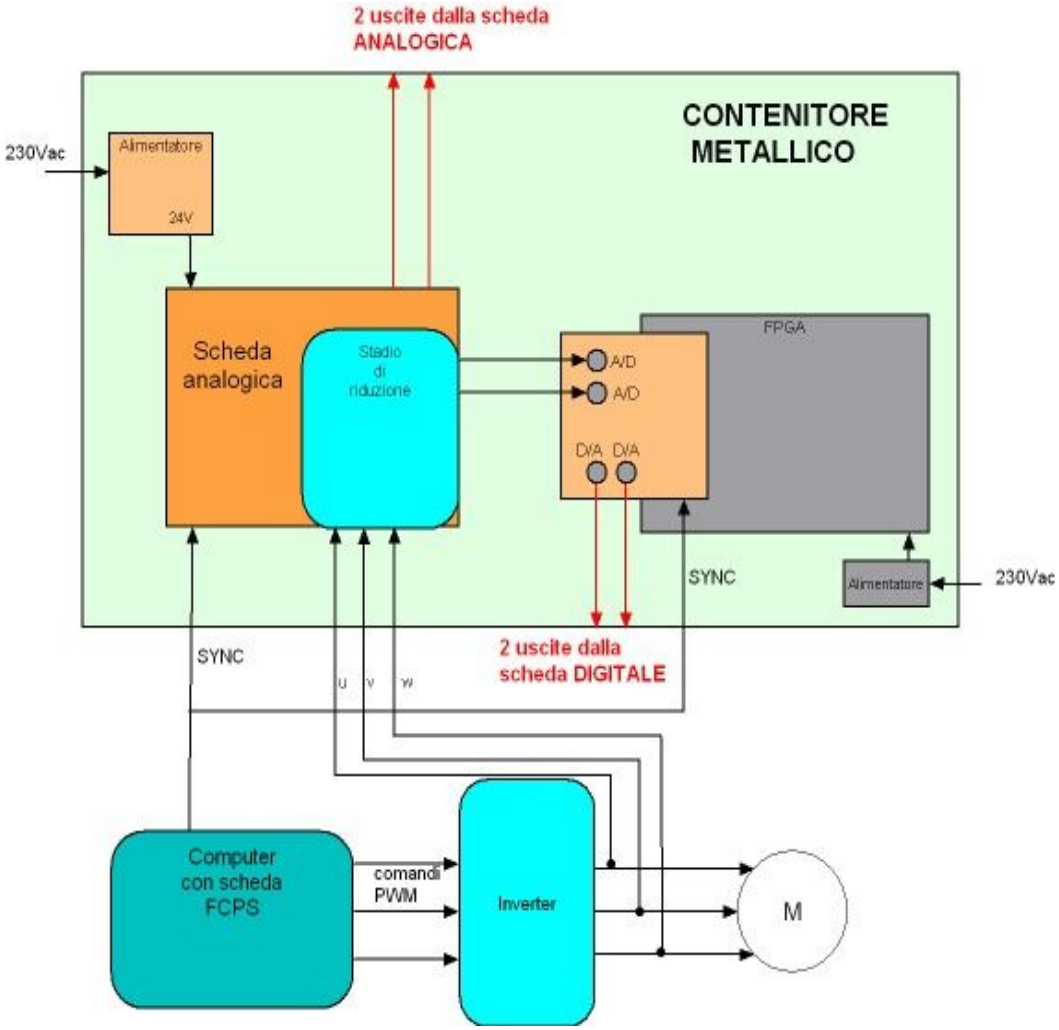


Illustrazione 4: schema a blocchi della versione preliminare del contenitore metallico

Per comandare il motore sincrono a magneti permanenti, da cui poi verranno prelevate le tensioni concatenate dal sistema di acquisizione, è stato implementato un controllo **FOC** (field oriented control = CONTROLLO A ORIENTAMENTO DI CAMPO) (vedi riferimento bibliografico [**1**]). Per implementare il controllo è stata impiegato un sistema **FPCS**, in particolare il

modello DS1104 (figura 5): una scheda add-on su bus PCI da installare su un normale PC.

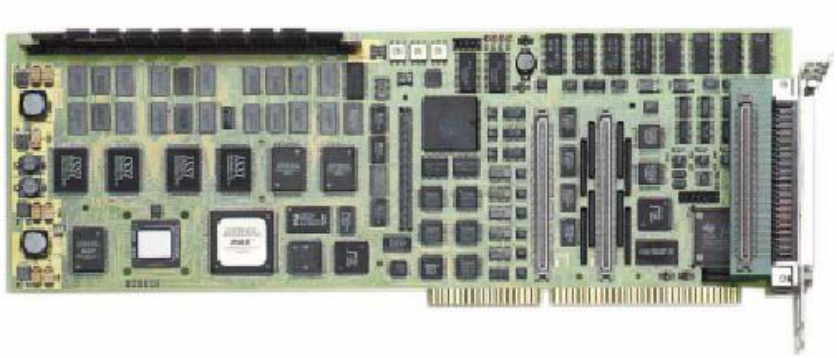


Illustrazione 5: Scheda di controllo DS1104 R&D

La DS1104 è un sistema "real-time" equipaggiato con un processore floatingpoint MPC8240 della Motorola, con architettura PowerPC 603e a 250MHz, che funge da dispositivo master. Al suo interno trova posto anche una DSP TMS320F240 della Texas Instrument (slave) che si occupa dell'interfacciamento con il sistema da controllare. Nell'illustrazione 6 è rappresentato uno schema blocchi che riassume l'intera architettura della DS1104:

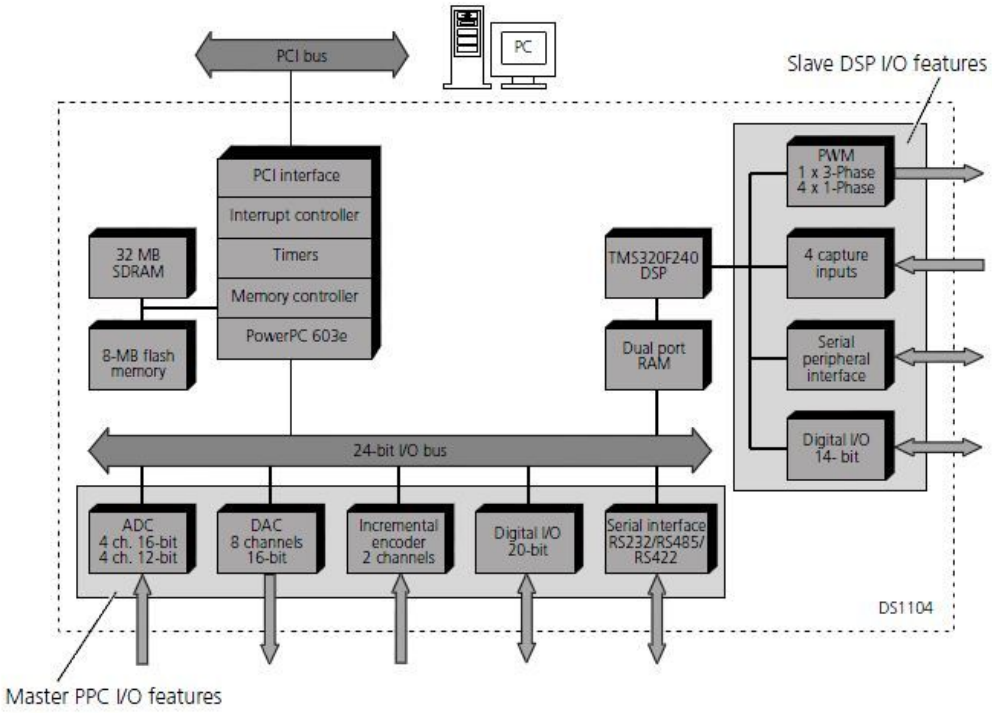


Illustrazione 6: Schema a blocchi del sistema FPCS DS1104

I sistemi FPCS sono dispositivi "**real-time**" orientati al "fast prototyping" di sistemi di controllo. Fra le loro potenzialità, c'è quella di una facile programmazione mediante schema Simulink. Il software fornito dal FPCS, interagendo con il Toolbox Real-Time Workshop di Matlab/Simulink, permette di convertire lo schema Simulink in codice C. Inoltre, grazie al programma *ControlDesk*, fornito sempre nel FPCS, è possibile realizzare in maniera veloce è intuitiva interfacce grafiche per monitorare o controllare tutte le grandezze del sistema. Infatti, *ControlDesk* permette di prelevare o intervenire su tutte le variabili presenti nello schema Simulink, modificarle in tempo reale durante il funzionamento del sistema e visualizzarle su sinottici grafici.

Ovviamente il programma di controllo dell'azionamento può essere implementato anche in C, come si fa normalmente per le schede "embedded" a microcontrollore.

Per l'interfacciamento fisico della DS1104 con il mondo esterno, è stata una usata la scheda connessioni CP1104+CLP1104, illustrata nella figura 7:

Illustrazione 7: Pannello di controllo CP1104+CLP1104

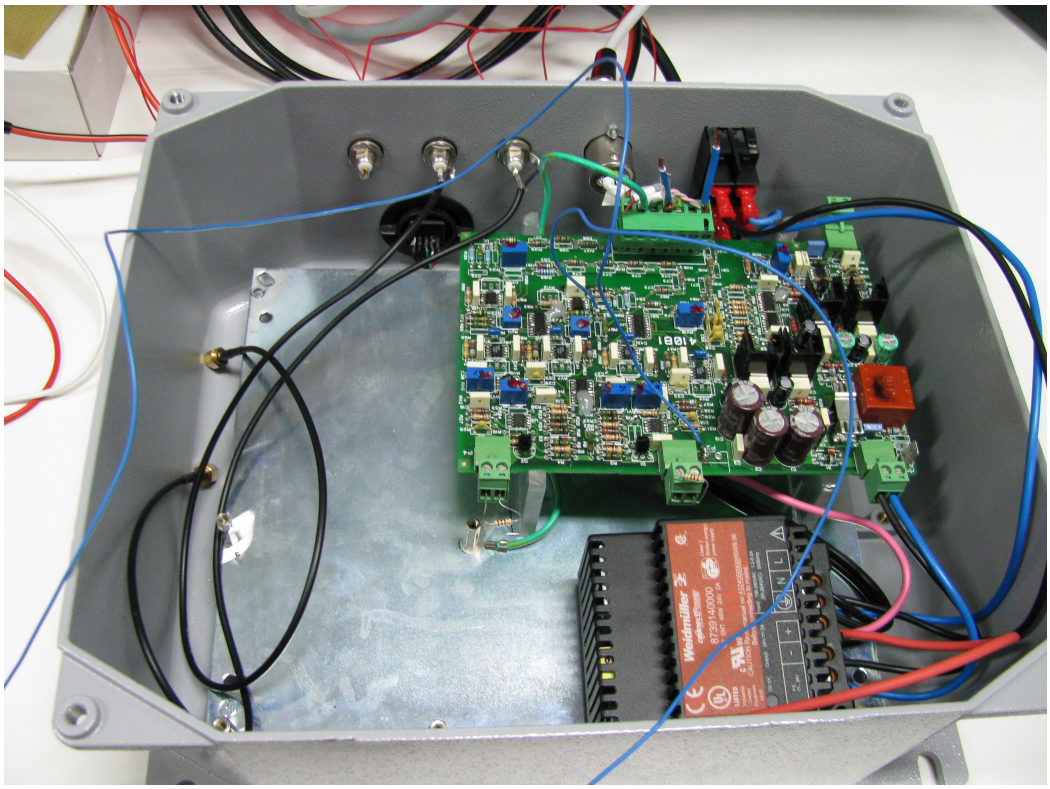
Il pannello connettore CP1104 consente semplici connessioni tra la scheda DS1104 R&D e i dispositivi ad essa connessi (ha connettori A/D, D/A, seriali, digitali I/O, di interfaccia per encoder incrementale) . In aggiunta al CP1104, il CLP1104 (*Connector/LED Combi Panel*) fornisce una serie di 54 LED indicanti lo stato dei segnali digitali.

E' stato previsto di predisporre entrambi i sistemi di acquisizione, sia quello analogico che quello digitale, all'interno di una **contenitore metallico** chiuso verso l'esterno e che rende disponibili, attraverso opportuni connettori, gli ingressi e le uscite delle schede per interfacciarsi con l'azionamento e con la rete di alimentazione.

Il motivo principale per cui è stato previsto questo contenitore metallico è quello di minimizzare i rumori, sempre molto presenti negli apparati con motori elettrici, che possono disturbare i segnali utili generati dalle schede, i quali devono essere precisi e senza rumore sovrapposto.

Per rendere disponibili i 24Vdc alla scheda analogica si è utilizzato un **alimentatore KERT KAL2402DIN**, il quale, dalla rete di alimentazione, genera un segnale continuo a 24V in grado di erogare fino a 2A.

Di seguito si può vedere un particolare dell'interno della scatola metallica con questi elementi:



Il segnale di sincronizzazione con il periodo di PWM dovrà necessariamente arrivare dal sistema di controllo del motore, cioè dalla scheda FPCS montata sul PC da dove si esegue l'algoritmo di controllo del motore. Vengono poi prelevate le tensioni di fase del motore in uscita dall'inverter che verranno elaborate dai 2 sistemi parallelamente.

Ai 2 A/D interfacciati con l'FPGA devono arrivare le tensioni concatenate del motore opportunamente ridotte per renderle compatibili con il fondo scala dei convertitori. Per fare questo si sfrutta lo stadio di riduzione della scheda analogica, che calcola le concatenate e le riduce di un fattore 1/200.

In questo modo si risparmia la predisposizione di un ulteriore circuito, poiché gli stessi segnali sono utilizzati come ingressi da entrambi i sistemi in parallelo.

Anche la scheda FPGA avrà bisogno, come ingresso, del segnale di sincronismo dal FPCS. Per alimentare il sistema digitale si usa l'alimentatore fornito dal costruttore della scheda. Infine entra nella scatola anche il connettore USB (non disegnato nelle illustrazioni) che, collegato ad un PC, permette la ri-programmazione dell'FPGA.

In uscita dal contenitore metallico si avranno quindi:

- 2 segnali proporzionali alla media sul periodo di modulazione PWM delle tensioni concatenate del motore elaborate dal sistema **ANALOGICO**.
- 2 segnali, in uscita da 2 distinti convertitori D/A, proporzionali alla media sul periodo PWM delle tensioni concatenate del motore elaborate dal sistema **DIGITALE**.

In seguito si potranno analizzare questi segnali per il confronto fra le 2 diverse elaborazioni.

Dopo l'analisi della configurazione di tutti gli apparati che compongono l'intera realizzazione e lo studio dei segnali che vengono scambiati dai vari sistemi, si sono riscontrate 2 problematiche principali:

- la scheda di misura analogica, per poter funzionare correttamente, ha bisogno che il segnale di sincronismo con il periodo di PWM sia in fase con esso e con duty-cycle al **50%**. Invece il segnale digitale che rende disponibile la scheda DSPACE attraverso i suoi pin I/O è normalmente a 5V, e con un impulso (di durata molto breve rispetto all'intero periodo) a 0V all'inizio del periodo di PWM. Questo è un output digitale che segnala quando viene lanciata l'interrupt del programma di controllo del motore per eseguire per la generazione della routine di servizio della PWM.

Quindi è necessario trasformare questo segnale digitale in uno in fase con esso ma con duty-cycle al 50%, come illustrato nella figura 8:

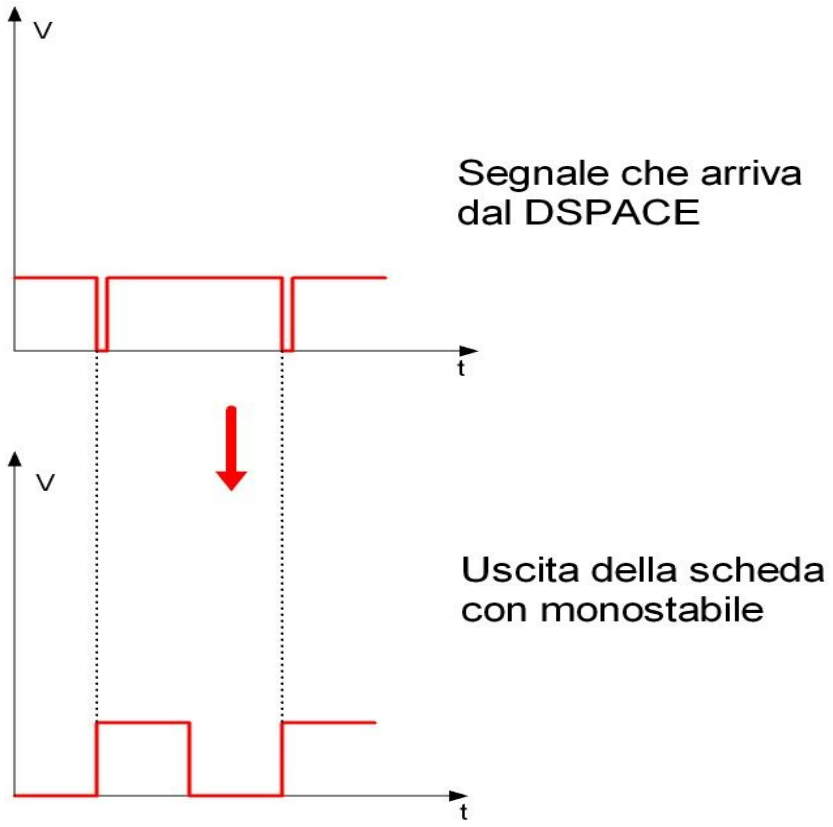


Illustrazione 8: elaborazione del segnale di sincronismo

- la schedina di interfaccia con l'FPGA ha, prima di entrare negli A/D, un trasformatore RF con banda di lavoro che va dai 0,4 ai 800 Mhz. Con questi componenti gli A/D lavorano solamente con segnali differenziali che rientrano nella banda dei trasformatori, mentre il resto viene tutto filtrato. Le tensioni concatenate che arrivano dallo stadio di riduzione della scheda analogica hanno però un'estensione di banda molto diversa da quella descritta sopra, e che contiene anche la componente continua. Inoltre sono segnali single-ended e con un'ampiezza massima che supera il valore di fondo scala dei convertitori A/D.

Per risolvere queste incompatibilità si sono costruiti altri 2 semplici circuiti che rendono tutti i sistemi interfacciabili fra loro.

La prima schedina realizzata, attraverso l'uso di un integrato **monostabile**, è posta tra il PC che contiene il sistema FPCS e l'ingresso per il sincronismo della scheda analogica. Nell'illustrazione 9 si può vedere il risultato sperimentale della

realizzazione di questa schedina attraverso l'analisi del suo ingresso (segnale verde che arriva dal FPCS) e della sua uscita (segnale giallo che rappresenta il segnale di sincronismo per la scheda analogica):

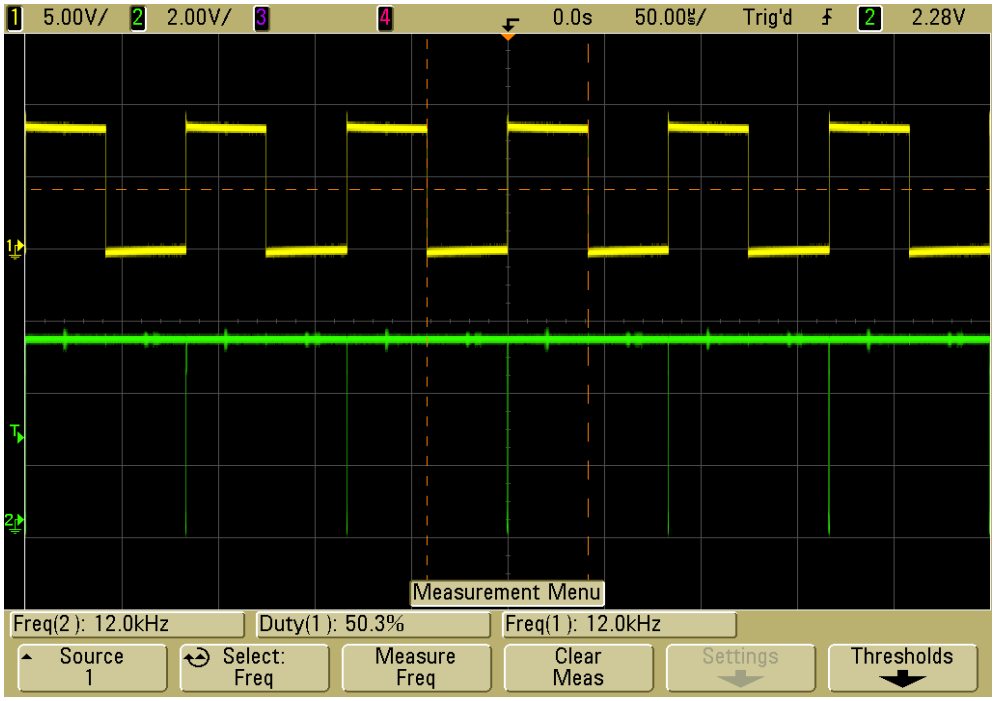


Illustrazione 9: GIALLO= uscita della scheda con monostabile VERDE= ingresso della scheda con monostabile

La problematica relativa ai trasformatori RF presenti nella scheda di interfaccia con gli A/D, è stata risolta nel seguente modo: sono stati tolti tutti i trasformatori RF e costruita un'ulteriore scheda con dei nuovi integrati, gli AD8138. Questi sono dei driver differenziali per i convertitori A/D che trasformano un segnale single-ended in un segnale differenziale, con un opportuno fattore di attenuazione e senza limiti di banda.

La nuova schedina acquisisce quindi le 2 tensioni concatenate del motore dallo stadio di riduzione della scheda analogica (single-ended), e le restituisce come segnali differenziali e con un ulteriore fattore di attenuazione.

L'intero apparato, dopo le modifiche apportate, si presenta come mostrato nell'illustrazione 10:

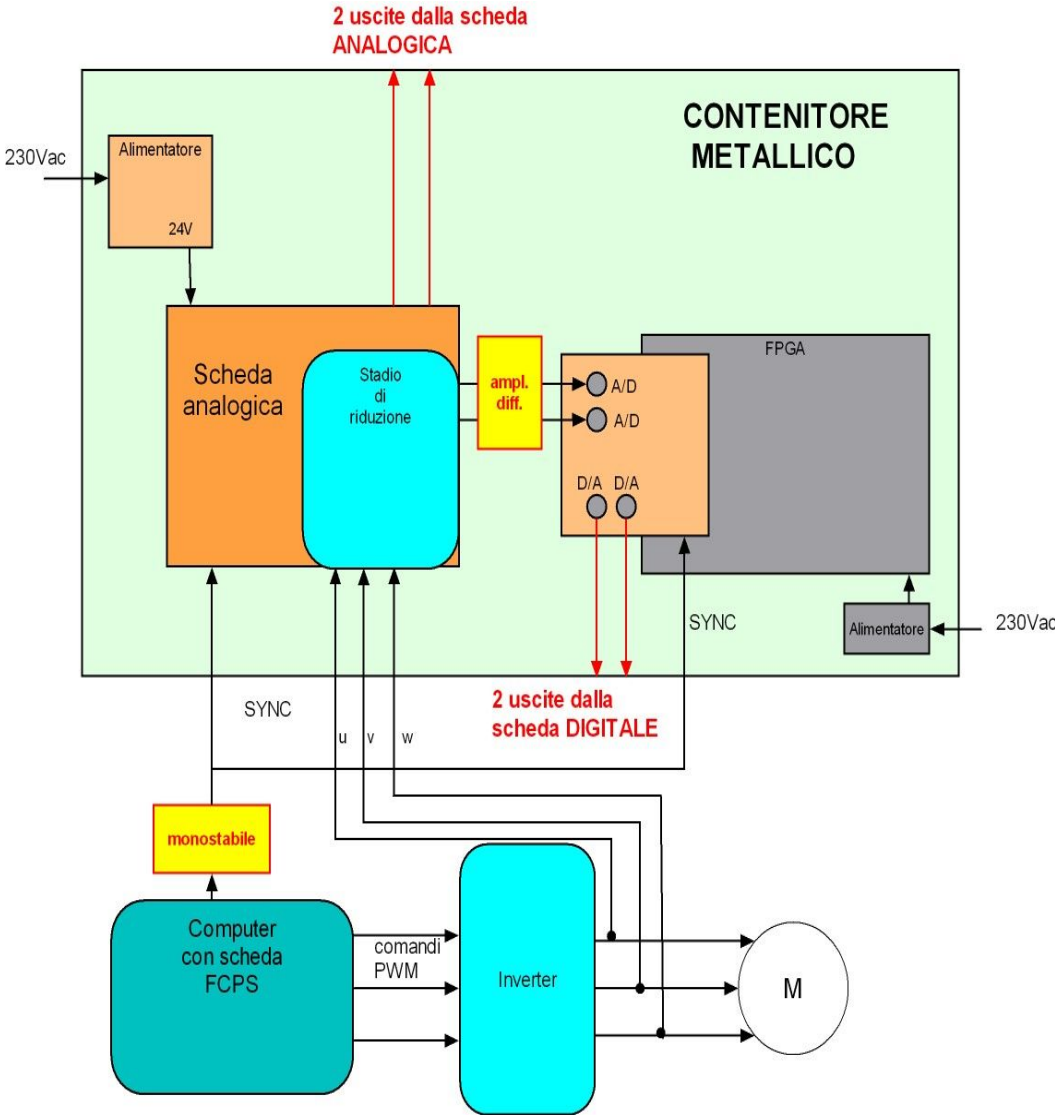
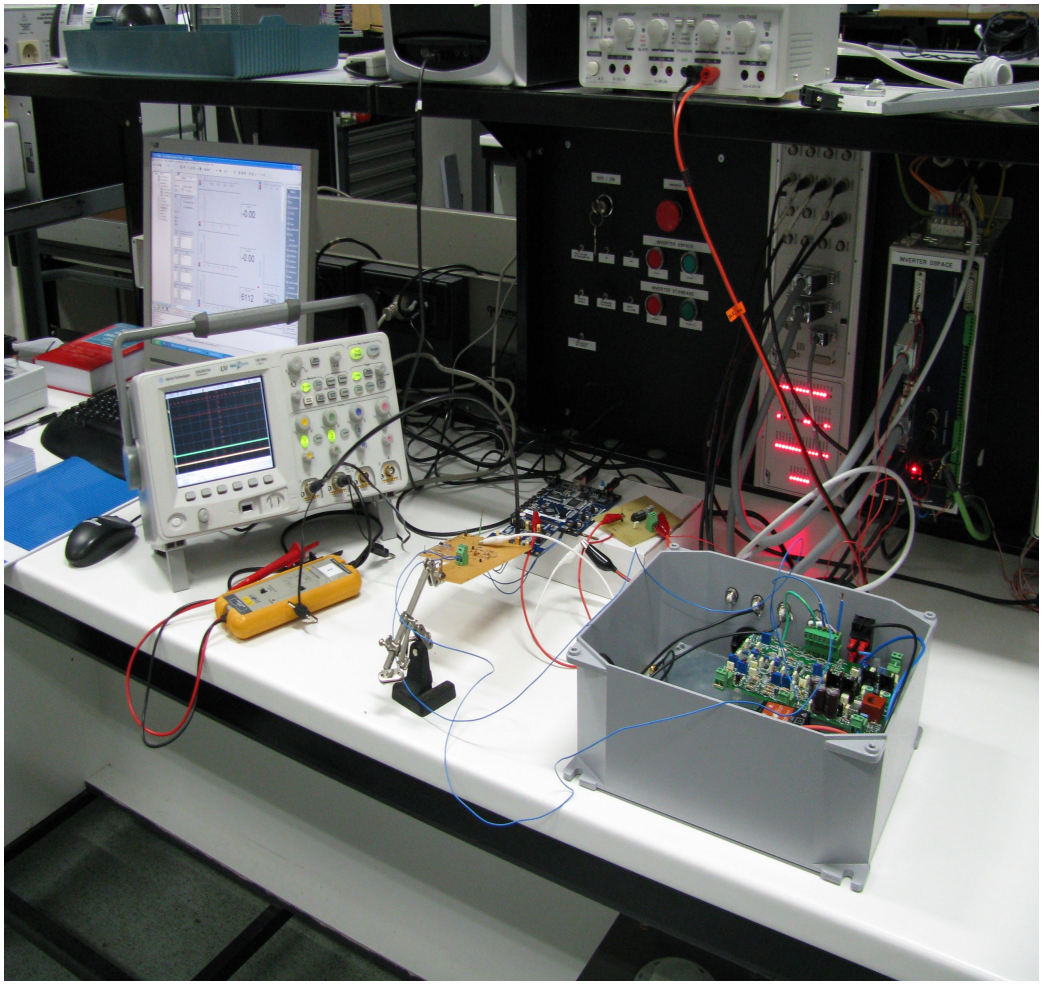


Illustrazione 10: Schema a blocchi del contenitore metallico modificato

La **massa** dell'intero apparato è unica: tutte le masse delle schede sono state accomunate, compresa quella del sistema FCPS (nei disegni non sono presenti le connessioni delle masse per maggiore chiarezza dell'illustrazione). Il neutro del motore non è disponibile, ma non serve perché qui vengono elaborate le concatenate (=segnali differenziali).

Di seguito viene riportata una fotografia di tutti i dispositivi collegati fra loro che sono stati descritti precedentemente, a cui si aggiunge un oscilloscopio digitale per poter effettuare le misure:



2. MISURA DELLE TENSIONI: LA SOLUZIONE ANALOGICA

2.1 Struttura generale della scheda e sua caratterizzazione

La scheda analogica 4108 è un prototipo prodotto dalla ditta REEL s.r.l. impiegata per rilevare e convertire in segnale analogico di corrente il valore medio delle tensioni concatenate in uscita ad un modulatore PWM.
Questa conversione in corrente è stata effettuata per rendere il segnale in uscita dalla scheda meno sensibile ai rumori creati dall'azionamento che potrebbero alterare sensibilmente un segnale in tensione. La ditta produttrice ha fornito il datasheet della scheda e le istruzioni per il suo collaudo.
I connettori che rappresentano gli ingressi della scheda sono 3: un connettore bipolare per il segnale di sincronismo, l'alimentazione a 24V e un connettore a tre vie per le 3 tensioni di fase del motore.
Il segnale di sincronismo deve avere le seguenti caratteristiche:

- segnale ad onda quadra con livello logico basso a 0V e livello logico alto a 5V
- essere in fase con il periodo di PWM dell'azionamento e con la stessa frequenza
- avere un duty-cycle del 50%

Le uscite invece sono 2, una per ogni segnale che rappresenta la componente fondamentale della tensione concatenata del motore. Ognuna viene fornita da un connettore bipolare, poiché le uscite sono single-ended.
La scheda è una multi-strato con un piano di massa e tutti i componenti sono in PTH. Le sue dimensioni sono di 16,9cm di lunghezza e 11,1 cm in larghezza.
La tensione in ingresso può arrivare fino a 1414 V_{dc}, pari a 1000 V_{rms}, visto lo stadio di riduzione in ingresso che abbassa di 200 volte le tensioni di fase del motore per rendere i segnali trattabili con gli integrati utilizzati.
Essa può essere suddivisa in 5 sottosistemi che possono essere studiati ed analizzati separatamente:

- Stadio di riduzione del segnale di ingresso: le tensioni di fase che arrivano dal motore vengono opportunamente ridotte e filtrate per permettere a questi segnali di poter essere trattati con amplificatori differenziali che elaborano le tensioni concatenate.
- Stadio di alimentazione: riceve in ingresso 24 V_{dc} e genera, attraverso un convertitore statico isolato, le tensioni continue necessarie ad alimentare i componenti e gli integrati della scheda.
- Stadio di sincronismo: attraverso un segnale di sincronismo in ingresso, più precisamente un'onda quadra a 12 kHz (stessa frequenza del periodo di PWM dell'azionamento) con duty-cycle al 50%, si crea una ben precisa sequenza di uscite opportunamente sfasate, che andranno a comandare interruttori o a resettare gli integratori dello stadio successivo.
- Stadio di integrazione: in questo circuito si calcolano le medie delle tensioni concatenate, attraverso un'integrazione dei segnali che gli arrivano in ingresso.
- Stadio di uscita: qui si effettua la conversione in segnale analogico di corrente attraverso un generatore analogico di corrente comandato in tensione.

Si analizzano ora nel dettaglio gli stadi di sincronismo e di integrazione, sia perché essi costituiscono il "cuore" della scheda dove si svolgono le funzioni più interessanti al fine di capire il suo funzionamento, sia perché essi dovranno successivamente essere realizzati in maniera digitale attraverso l'utilizzo di una FPGA.

Per lo studio dei circuiti e la verifica delle loro effettive funzionalità si è fatto uso di un alimentatore 24V/0,5 A, un oscilloscopio, un generatore di funzioni ed un multimetro.

2.2 Stadio di sincronismo

L'ingresso di questo circuito è un tipico segnale di sincronismo con le seguenti caratteristiche:

- segnale ad onda quadra da 0 a 5V
- duty cycle al 50%
- frequenza di 12kHz (la stessa della PWM dell'azionamento)
- in fase con il periodo della PWM

Questo stadio genera 6 segnali ad onda quadra da 0 a 12V, a partire da quello descritto precedentemente, con determinate caratteristiche (si veda l'illustrazione 11), per poter comandare nella maniera corretta degli interruttori (MOSFET) posti nello stadio di integrazione. Questi MOSFET servono ai doppi integratori per deviare su uno solo di essi il segnale da integrare e permettere di resettarsi alla fine di ogni periodo di PWM.

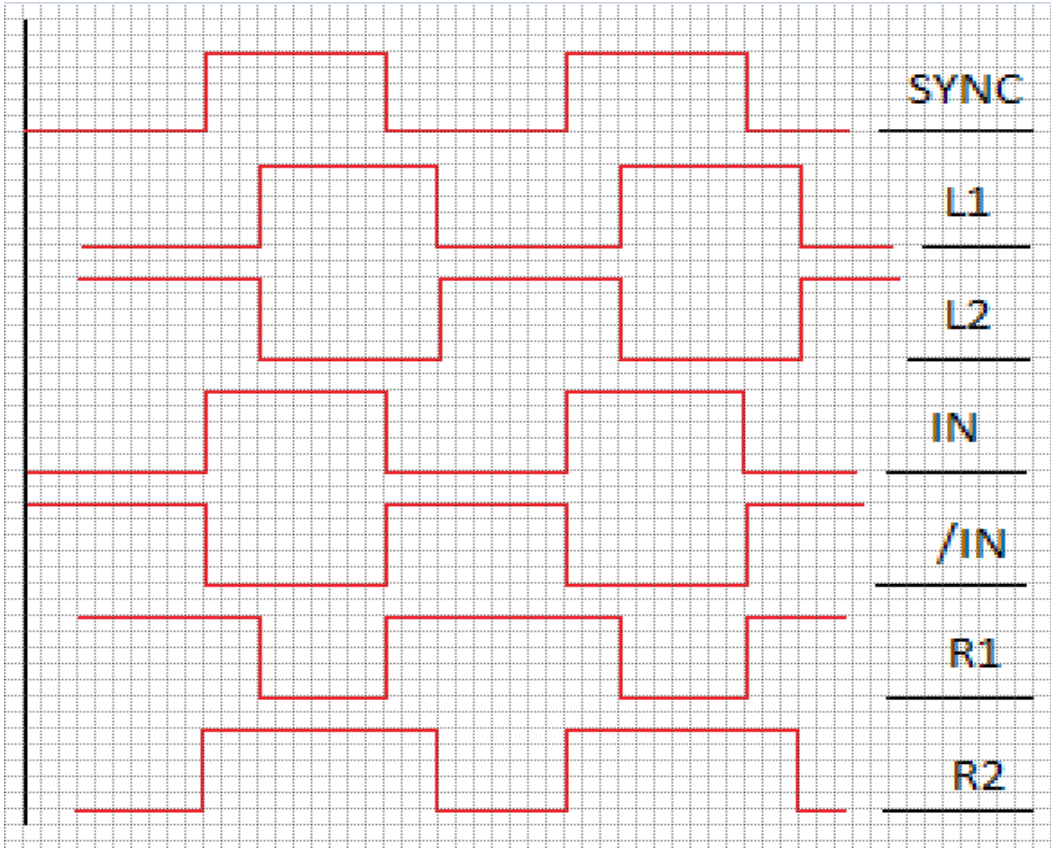


Illustrazione 11: segnali di sincronismo

I segnali nella figura precedente hanno le seguenti denominazioni:

- **SYNC** = è il segnale d'ingresso del circuito descritto precedentemente

- **L1** = 1° uscita del sistema. E' un'onda quadra tra 0 e 12V sfasata rispetto al SYNC di ¼ del periodo di PWM. Con un periodo di 12kHz c'è un ritardo di circa 21μs, impostabile con un trimmer in fase di collaudo della scheda esattamente a ¼ del periodo di PWM

- **L2** = 2° uscita del sistema. E' un'onda quadra tra 0 e 12V in controfase rispetto a L1 (sootolineatura=negato):

  $$L2 = \underline{L1}$$

- **IN** = 3° uscita del sistema. E' un'onda quadra tra 0 e 12V in fase con SYNC

- **/IN** = 4° uscita del sistema. E' un'onda quadra tra 0 e 12V in controfase rispetto a IN:

  $$/IN = \underline{IN}$$

- **R1** = 5° uscita del sistema. E' un'onda quadra tra 0 e 12V generata dalla seguente funzione logica:

  $$R1 = \underline{IN \cdot L1}$$

  Essa va a comandare il reset di un dei 2 integratori (attivo basso).

- **R2** = 6° uscita del sistema. E' un'onda quadra tra 0 e 12V generata dalla seguente funzione logica:

  $$R2 = \underline{/IN \cdot L2}$$

  Essa va a comandare il reset dell'altro integratore.

Si analizza ora com'è stato realizzato il circuito di sincronismo che permette di ottenere i segnali descritti sopra.
L'ingresso ad onda quadra SYNC viene invertito e posto tra -12V e +12V da un comparatore invertente. Questo comparatore è realizzato attraverso un

amplificatore operazionale (LM393 della National Semiconductor) che confronta il segnale SYNC con un riferimento (=2,17V) e restituisce in uscita un segnale ad onda quadra basso (-12V) se SYNC>2,17V, altrimenti alto (+12V). Nella figura 12 sono illustrate le misure effettuate sul segnale appena descritto (segnale ROSA=SYNC, segnale VERDE=onda quadra +/-12V in uscita dal comparatore invertente). Attraverso uno stadio con 2 BJT, uno npn e uno pnp, alimentati a +12V e -12V, il segnale siffatto viene ulteriormente definito e portato ad un valore maggiore di corrente per comandare i successivi stadi.

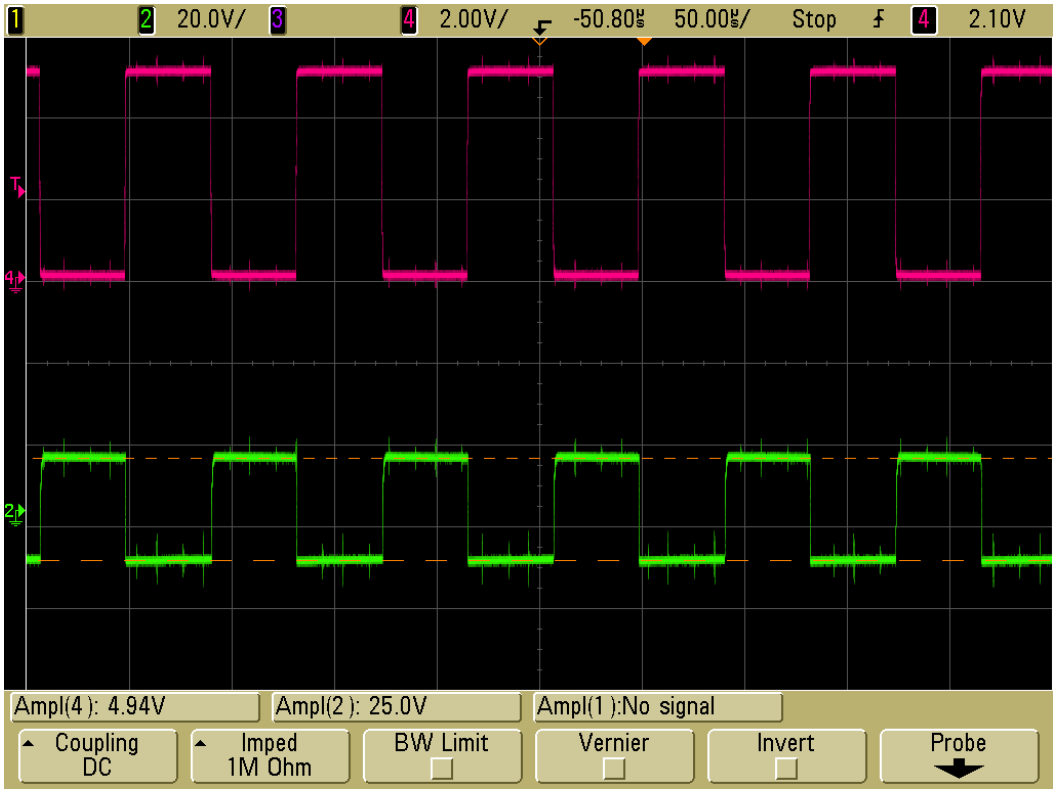


Illustrazione 12: Segnale ROSA = SYNC Segnale VERDE = onda quadra +/-12V in uscita dal comparatore invertente

§

Per generare il segnale L1 sfasato di 21µs rispetto all'ingresso si è utilizzato un filtro RC seguito da un trigger di Schimdtt: quest'ultimo è un particolare tipo di squadratore, ovvero un circuito che consente di trasformare un segnale analogico in un'uscita che varia soltanto tra due valori di tensione a seconda che l'ingresso superi una certa soglia o sia inferiore ad una seconda soglia (più bassa).

Pertanto l''uscita può assumere un valore di tensione o basso o alto. In ingresso il trigger ha due soglie, una alta e una bassa non coincidenti: in un circuito invertente (come quello progettato nella scheda analogica) quando l'entrata è al di sotto della soglia bassa, l'uscita assume il valore alto; quando l'entrata si trova al di sopra della soglia alta (più elevata), l'uscita assume il valore basso. Quando il valore in ingresso si trova compreso tra le due soglie, l'uscita conserva il valore precedente finché l'entrata non sia variata sufficientemente da farne scattare il cambio (isteresi).

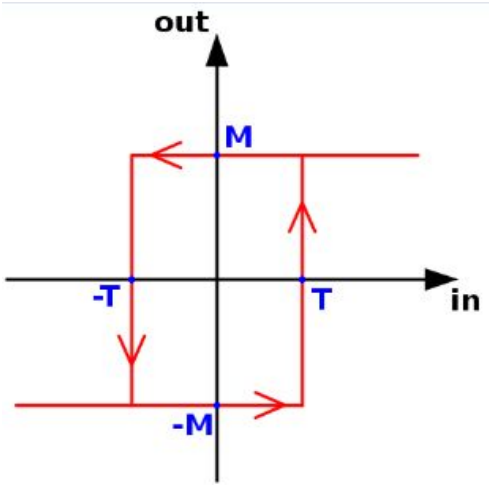


Illustrazione 13: Curva tipica di ISTERESI

Nella scheda analogica 4108 il trigger è realizzato attraverso un amplificatore operazionale: l'onda quadra +/-12V viene trasformata in un segnale a rampa dal filtro RC. L'amplificatore operazionale funge, anche in questo caso, da comparatore fornendo +12V (la sua tensione di saturazione) quando la tensione d'ingresso (V_{IN}) al morsetto invertente è minore della tensione al morsetto non invertente, che rappresenta la tensione di riferimento (V_{REF}). Al contrario, quando $V_{REF}<V_{IN}$, la tensione d'uscita commuta a 0V (tensione nulla) perché non può assumere valori negativi per la presenza del diodo in uscita.
In questo modo il segnale d'ingresso viene invertito e sfasato del tempo (t_A-t_B) (si veda l'illustrazione 12).

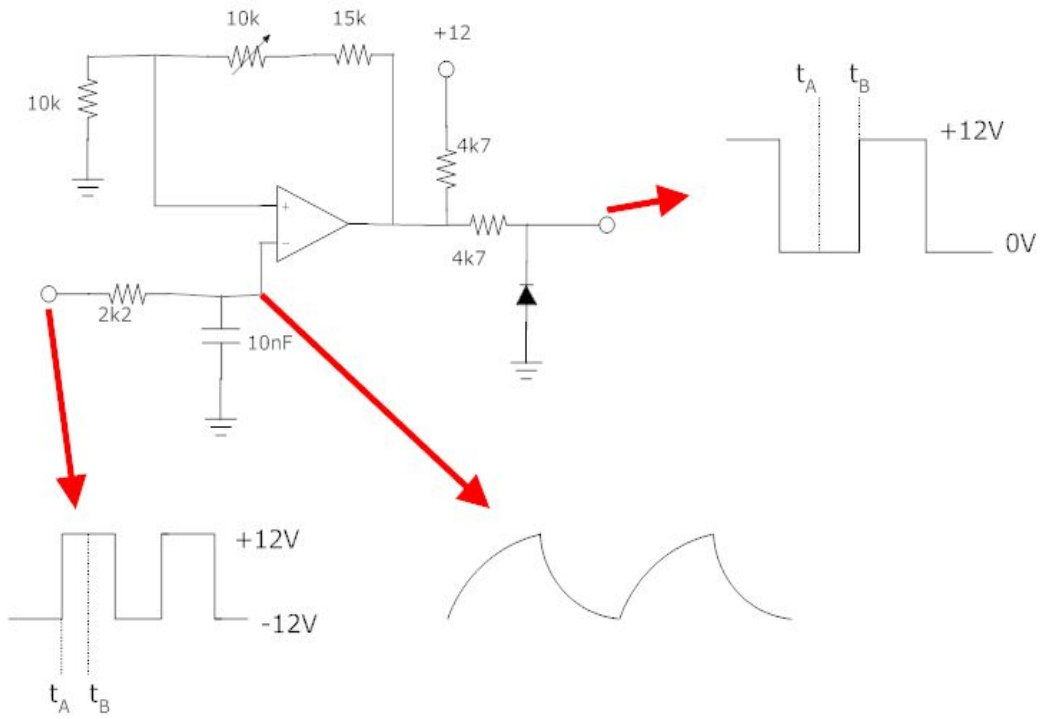


Illustrazione 14: filtro RC e trigger di Schimdtt

In questo caso la V_{REF} è data dal partitore resistivo:

$$Vref = \frac{12V * 10k\,\Omega}{4{,}7k\,\Omega + 10k\,\Omega + 15k\,\Omega + 10k\,\Omega} = 3{,}02\,V$$

Quando la V_{IN} è compresa fra 0V e 3,02V l'uscita non commuta e rimane fissata al valore logico precedentemente assunto (effetto di isteresi).

Riprendendo la nomenclatura dell'illustrazione 11 si ottiene:

M = 12V

-M = 0V

T = 3,02V

-T = 0V

Il trimmer da 10kΩ è stato inserito per poter tarare, in fase di collaudo della scheda, il ritardo esattamente ad ¼ del periodo di PWM.

Attraverso degli integrati che svolgono delle semplici funzioni logiche binarie (in questo caso AND o invertitori) come descritte precedentemente vengono generati anche L2, IN, /IN, R1, R2.

Ogni segnale è in uscita da un invertitore (o la cascata di 2 invertitori) per avere una adeguata resistenza di uscita e da un filtro passa basso RC per limitare le alte frequenze dei fronti di salita dei segnali.

Nelle immagini 15, 16, 17, 18, 19, sono illustrate le misure effettuate su questo stadio della scheda analogica di misura per verificare le forme d'onda di tutti i segnali di uscita di questo circuito. Esse rispondono perfettamente alla loro funzione logica. Osservando le misurazioni dei segnali L1, L2, /IN, R1, R2 si può notare come esse abbiano un disturbo sovrapposto di frequenza maggiore rispetto ai 12kHz dei segnali di sincronismo. Questo è dovuto allo stadio di alimentazione in configurazione flyback della scheda, che lavora a frequenze nell'ordine delle decine di kHz.

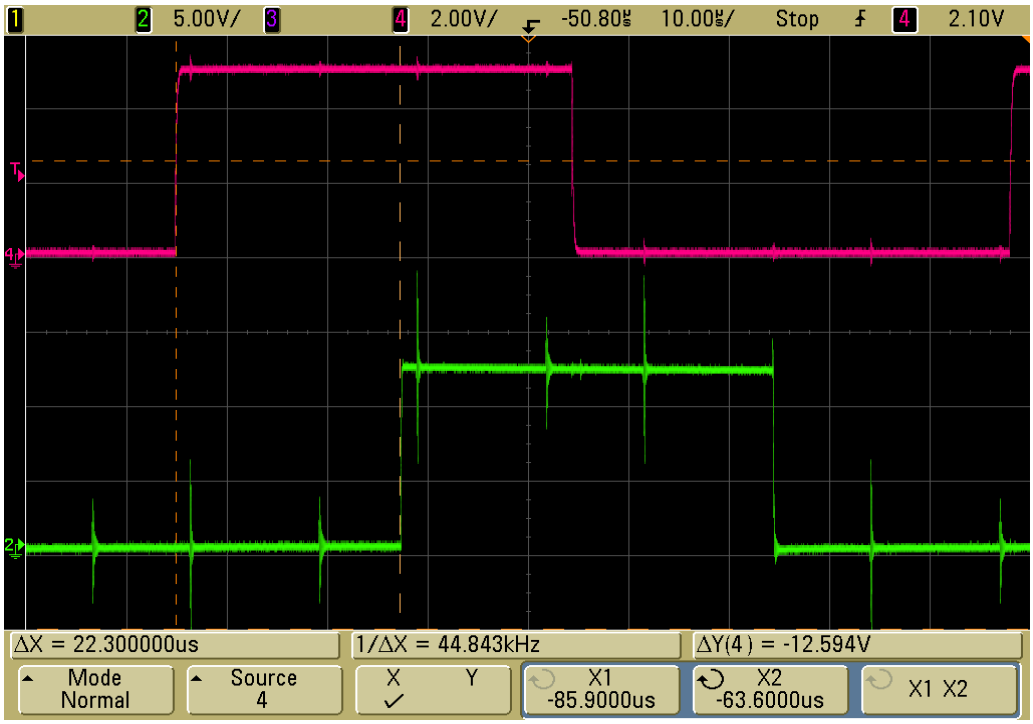


Illustrazione 15: Segnale ROSA = IN Segnale Verde = L1

Testo 1: In evidenza lo sfasamento fra i segnali di ***22,3us***

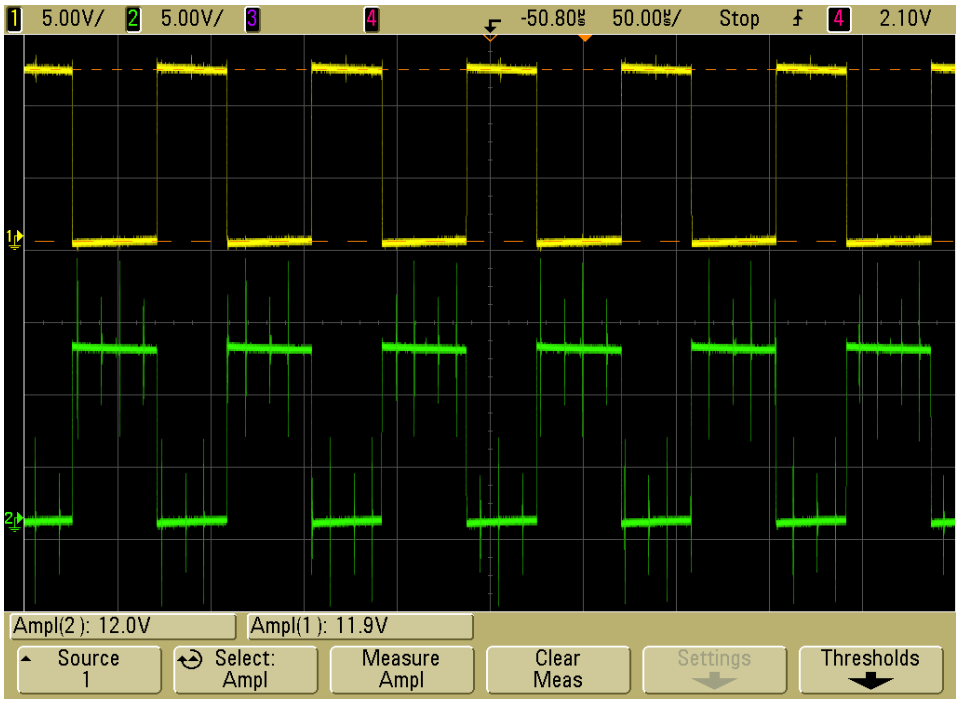


Illustrazione 16: Segnale GIALLO = IN Segnale VERDE = /IN

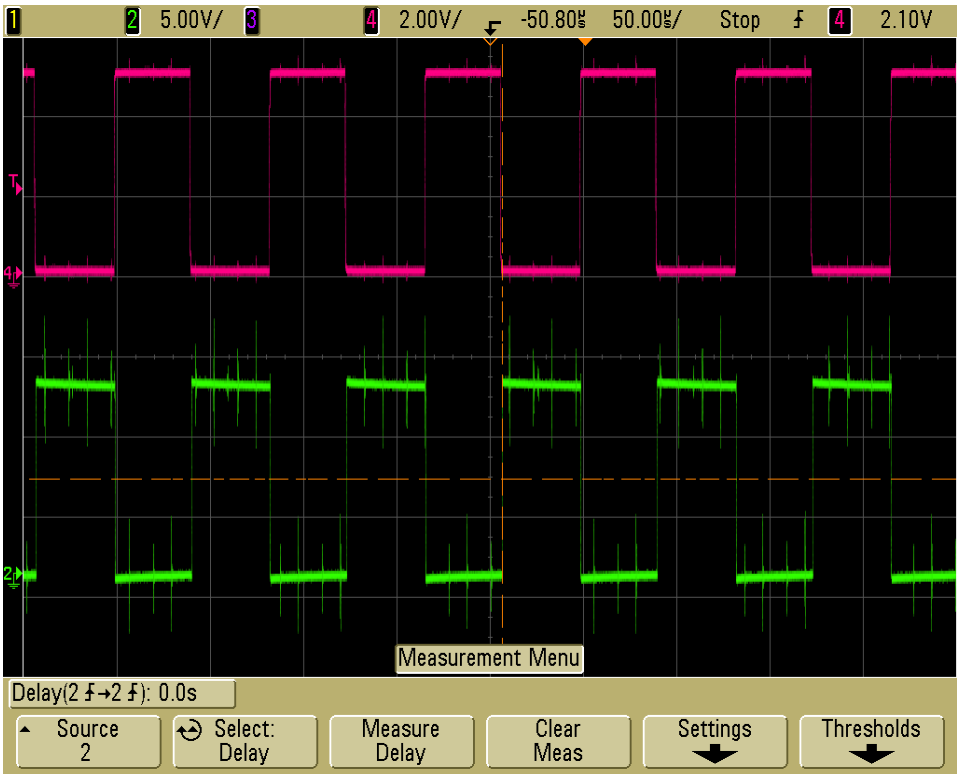


Illustrazione 17: Segnale ROSA = SYNC Segnale VERDE = IN

Testo 2: In evidenza lo sfasamento (nullo) fra i due segnali

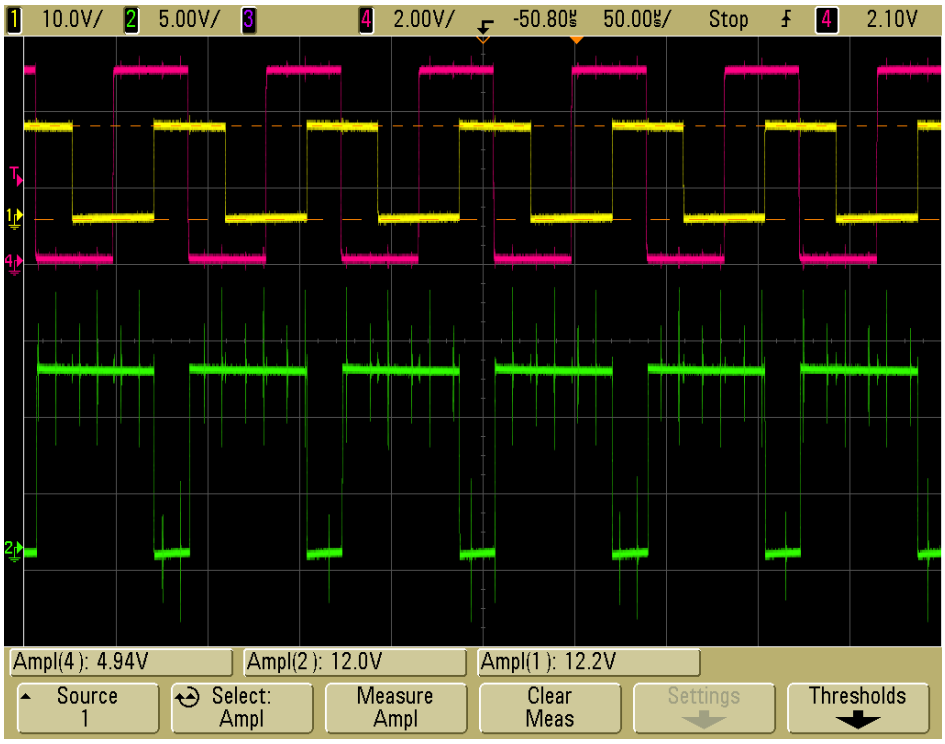


Illustrazione 18: Segnale ROSA = SYNC Segnale GIALLO = L1 Segnale VERDE = R1

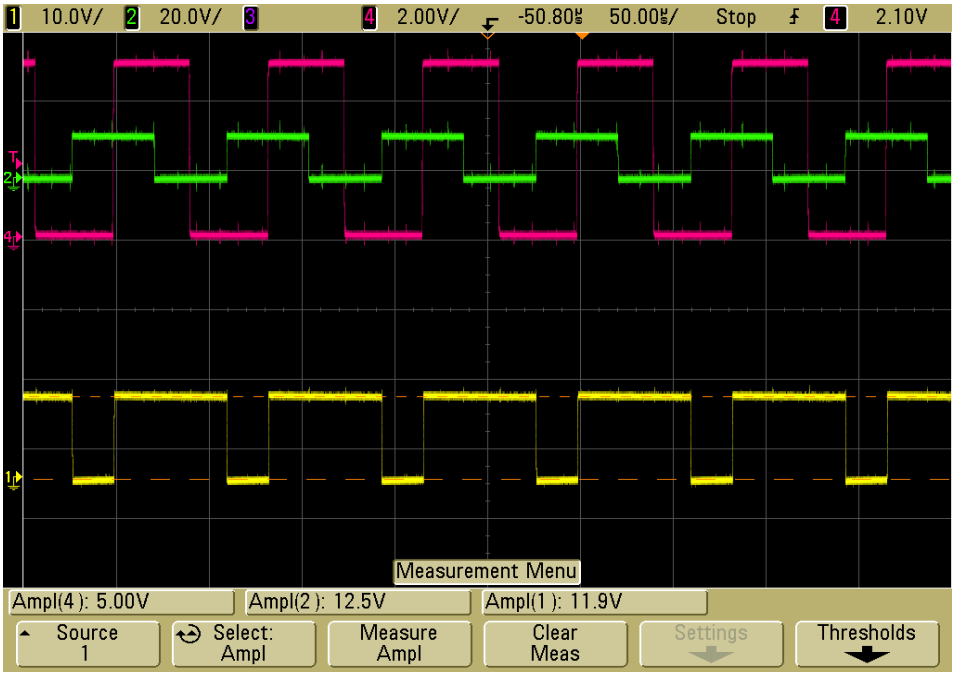


Illustrazione 19: Segnale ROSA = SYNC Segnale VERDE = L2 Segnale GIALLO = R2

2.3 Stadio di integrazione

Questo stadio rappresenta il "cuore" della scheda in quanto è qui che vengono calcolate effettivamente le medie dei segnali che arrivano dallo stadio di riduzione.

I segnali d'ingresso di questa parte della scheda sono le tensioni concatenate del motore (opportunamente ridotte per renderle compatibili con gli integrati utilizzati) e i segnali di sincronismo descritti nel paragrafo precedente.

Per calcolare le medie si utilizzano degli integratori analogici, realizzati secondo lo schema di figura 20 (si veda per approfondimenti il riferimento bibliografico [10]).

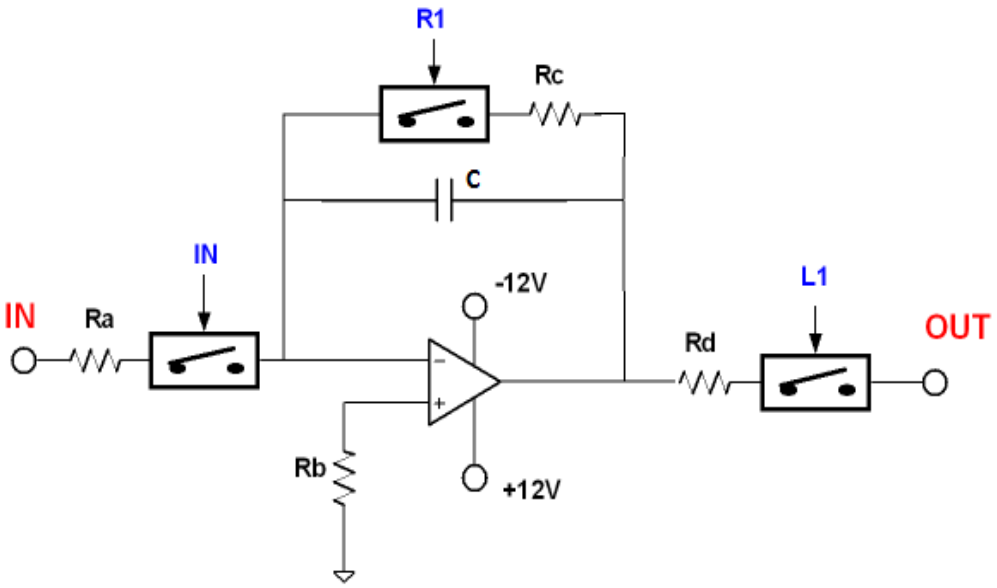


Illustrazione 20: Configurazione dell'integratore comandato

Si tratta della classica configurazione di un integratore analogico realizzato con un amplificatore differenziale in configurazione invertente:

$$V_o = -\frac{1}{R_a C}\int_0^{T/2} V_{in}\,dt$$

Il suo guadagno (teorico a causa delle tolleranze dei componenti), con i valori di resistenze e condensatori presenti sulla scheda, è:

$$\frac{-1}{Ra*C}=\frac{-1}{24{,}3\,k\,\Omega*2{,}2\,nF}=-18705{,}57$$

R_b (=10kΩ) serve per minimizzare l'offset dovuto alle correnti di polarizzazione dell'amplificatore operazionale.

Per ogni tensione concatenata è previsto un **DOPPIO CANALE DI INTEGRAZIONE** (si veda l'illustrazione 21): la doppia integrazione è resa necessaria a causa della configurazione analogica degli integratori che devono per forza avere il tempo di resettarsi all'interno del periodo di PWM, altrimenti saturerebbero.

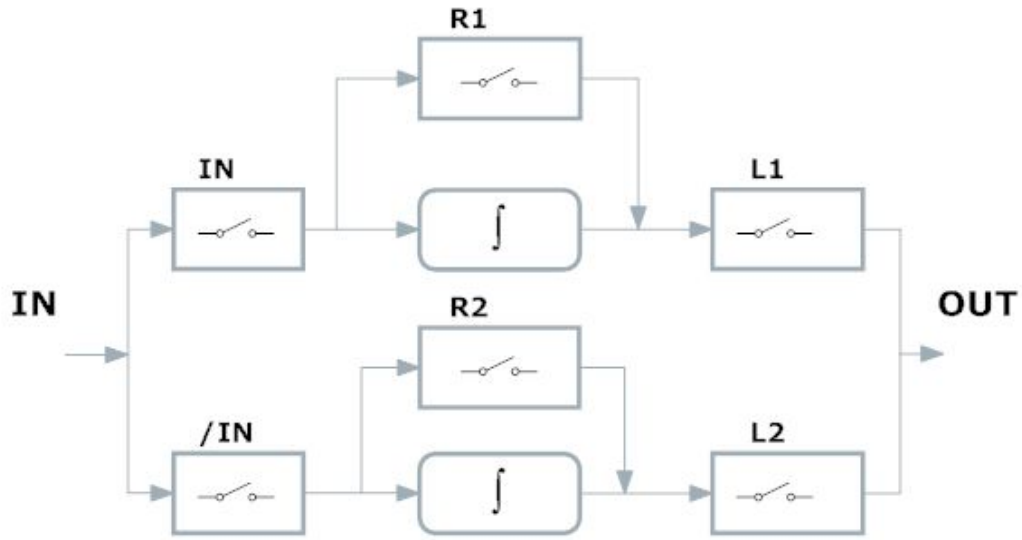


Illustrazione 21: Schema a blocchi del doppio integratore

Ogni canale integra un semiperiodo della forma d'onda che arriva in ingresso: mentre un integratore lavora, l'altro ha il tempo memorizzare il valore calcolato su una capacità di uscita e resettarsi. Lavorano quindi in maniera speculare e alternata uno rispetto all'altro.

Questa configurazione di "doppio canale di integrazione" spiega la presenza degli interruttori nello schema dell'integratore analogico con amplificatori operazionali. Attraverso la loro chiusura permettono di "deviare" il segnale di ingresso verso il canale già resettato e di "caricare" su una capacità in uscita l'effettiva media (sul semiperiodo) della concatenata.

Analizzando i segnali di sincronismo si può capire il funzionamento del doppio canale di integrazione, seguendo la tabella logica nella figura 22.

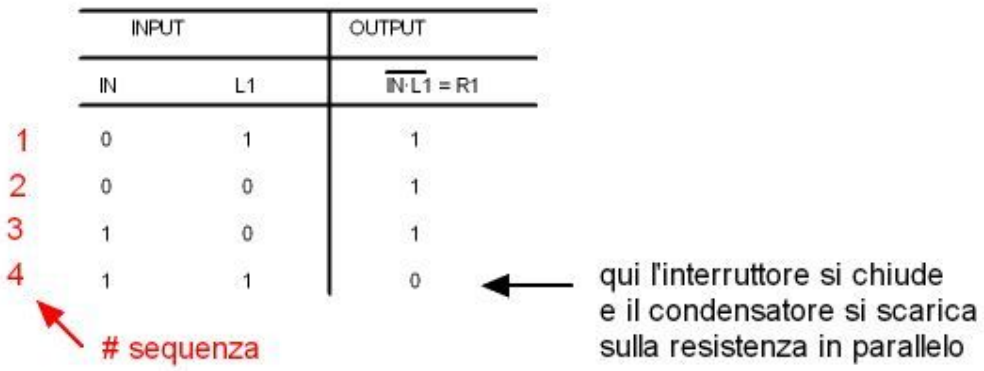


Illustrazione 22: Tabella logica dei comandi degli interruttori dell'integratore

Gli interruttori di questo stadio (integrati DG611 della VISHAY) funzionano in logica negata: con una tensione di comando "alta" (>4V) gli interruttori sono aperti, mentre con una tensione "bassa" (<1V) si chiudono.

Attraverso i segnali, ricavati dallo stadio di sincronismo, **IN** (interruttore in ingresso), **L1** (interruttore in uscita) e **R1** (quando è chiuso scarica il condensatore su R_c), che comandano ognuno la chiusura di un interruttore nella configurazione dell'illustrazione 21, si può capire il funzionamento di un singolo canale di integrazione: quando **IN** è chiuso (sequenza 1 e 2) l'integratore sta calcolando la media mobile della tensione concatenata del motore. Successivamente **IN** viene aperto (sequenza 3), **L1** è chiuso e viene caricata la capacità di uscita con il valore dell'integrazione ottenuto sul semiperiodo di PWM. Qui è possibile campionare il segnale in uscita, stabile per un quarto di periodo. Infine nella sequenza 4 l'integratore si resetta attraverso la chiusura di **R1**, mentre gli altri 2 interruttori sono aperti.

Ogni sequenza dei segnali logici riportati nella tabella logica dell'illustrazione 22 avviene in un quarto di periodo PWM; un ciclo completo copre l'intero periodo.

Mentre un canale integra, l'altro pone in uscita la media del semiperiodo precedente e si resetta successivamente. Lavorano quindi in maniera speculare e alternata uno rispetto all'altro.

2.3.1 Prove e misure sullo stadio di integrazione

Per capire questo funzionamento sono state effettuate numerose misure e prove sulla scheda analogica. Le più interessanti sono quelle riportate di seguito nelle illustrazioni 23 e 24: per ottenerle si è utilizzato come segnale di sincronismo un'onda quadra a 12kHz, ottenuta con un generatore di funzioni, e per i segnali di ingresso una tensione continua a 2,5V, ottenuta attraverso un alimentatore regolabile in continua applicata direttamente agli ingressi dello stadio di integrazione. Si è deciso di "bypassare" lo stadio di riduzione perché non era possibile ottenere in laboratorio una tensione continua abbastanza grande da poter apprezzare le variazioni dei segnali in questo stadio di integrazione.

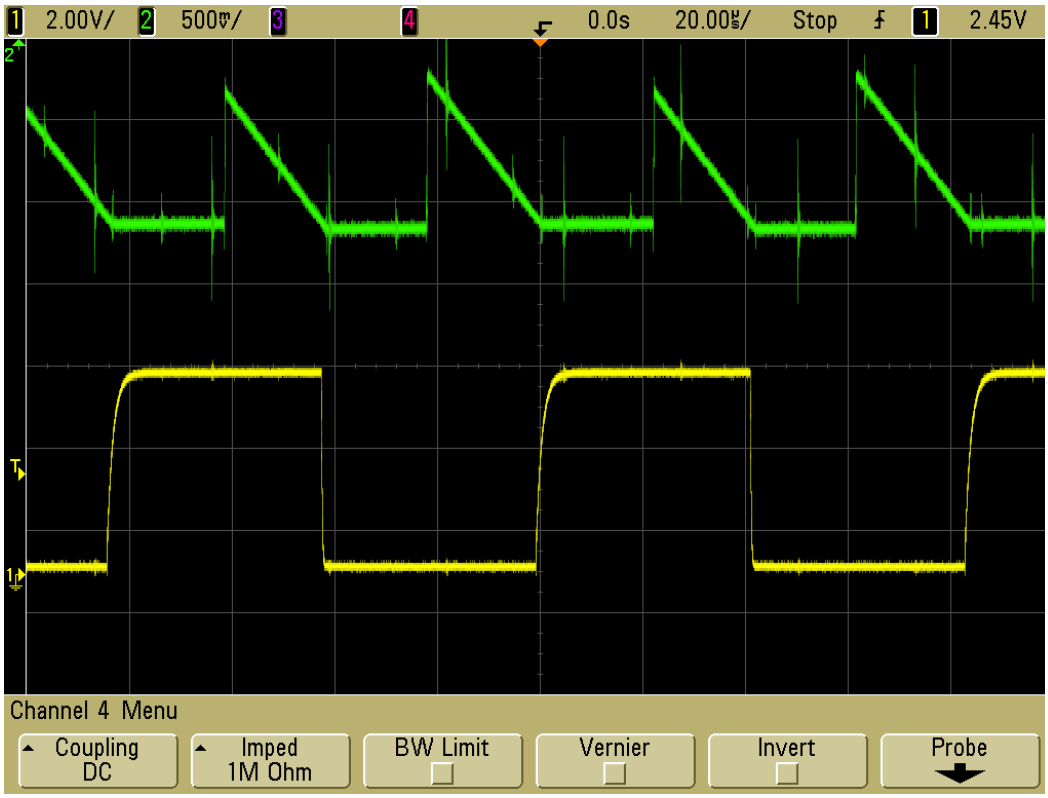


Illustrazione 23: GIALLO=SYNC, VERDE=uscita del doppio canale di integrazione

Con queste caratteristiche, tenendo conto delle formule precedenti e che si integra un segnale costante nel periodo, si ottiene in uscita un segnale continuo con la seguente ampiezza:

$$Vout = \frac{-18705 * Vin * Tc}{2} = \frac{-18705 * 2{,}5\,V}{2 * 12\text{kHz}} = -1{,}95\,V$$

Le misure effettuate (illustrazione 23 e 24) riportano gli stessi valori calcolati, tenendo conto anche delle tolleranze dei componenti utilizzati.

Nell'illustrazione 23 si è evidenziato come il segnale in uscita dal doppio canale abbia frequenza doppia rispetto a SYNC. Questo perché è resa disponibile la media mobile dell'ingresso sul semiperiodo. Nei quarti di periodo, dove il segnale in uscita NON deve essere campionato, si vede direttamente l'uscita dell'integratore (segnale a “triangolo”).

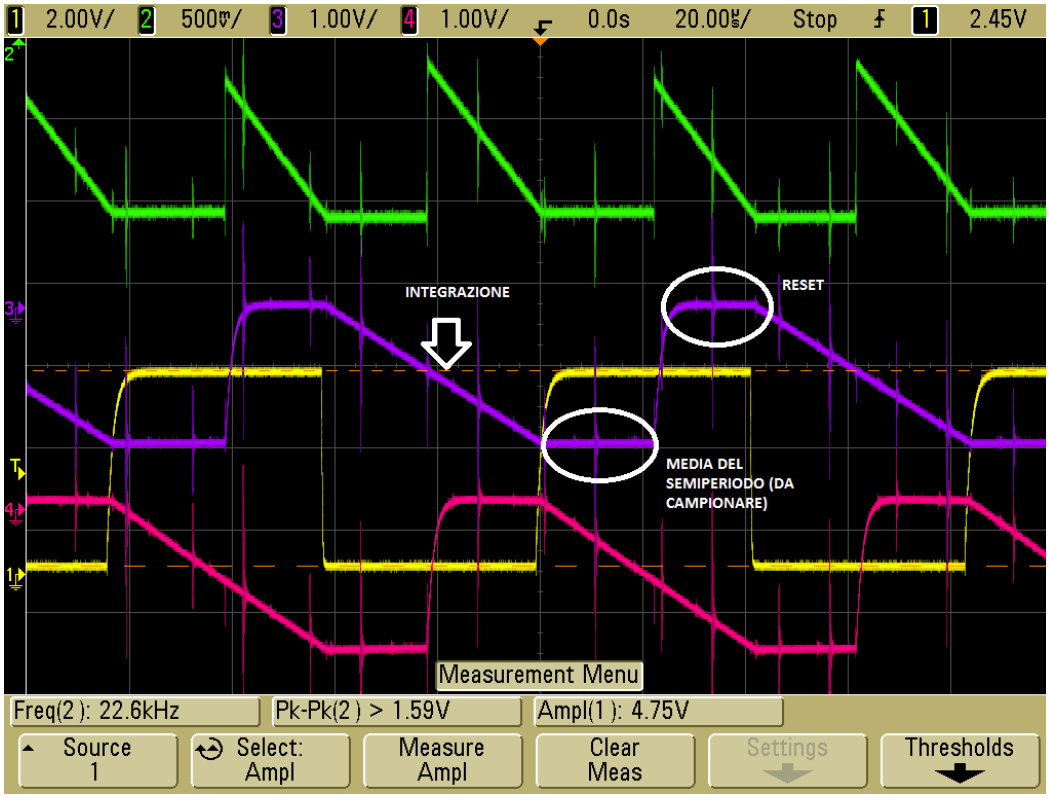


Illustrazione 24: GIALLO=SYNC, VERDE=uscita del doppio canale, ROSA e BLU= uscita di ogni singolo canale

Nell'illustrazione 24 sono messe in evidenza anche le uscite di ogni singolo integratore: qui si può notare come ognuno segua la sequenza di integrazione-campionatura-reset, lavorando in maniera speculare e integrando ognuno per metà periodo del SYNC. Il segnale integrato va campionato (dal sistema di controllo dell'azionamento) ad ogni cambio di fronte di SYNC e rimane costante per un quarto del periodo di sincronismo.

2.4 Gli altri stadi e interconnessione con il resto dell'azionamento

Gli altri stadi che compongono la scheda analogica di misura sono:

- lo stadio di riduzione del segnale d'ingresso: qui le 3 tensioni di fase del motore vengono opportunamente attenuate per poter essere trattate con gli integrati degli stadi successivi e vengono calcolate le loro differenze reciproche, le cosiddette tensioni concatenate.
- lo stadio di alimentazione: attraverso un convertitore switching isolato i 24V in ingresso alla scheda vengono ridotti per generare le tensioni continue che serviranno ad alimentare gli integrati degli altri stadi.
- lo stadio finale: in uscita le medie mobili delle tensioni concatenate vengono "trasformate" in segnali ad esse proporzionali di corrente per rendere questi valori maggiormente immuni ai disturbi, molto spesso non trascurabili negli ambienti industriali dove vengono utilizzati alcuni motori.

Di seguito verranno descritte in maggior dettaglio queste parti del circuito analogico.

2.4.1 Stadio di riduzione del segnale d'ingresso

Le tensioni di fase dei motori di media/grossa portata possono arrivare a qualche centinaio di Volt. Devono quindi per forza essere ridotte in ampiezza per effettuare delle elaborazioni su di esse con i normali amplificatori operazionali. Inoltre nella maggior parte dei casi il "neutro" del motore non è disponibile, anche se in questo caso non è necessario in quanto le fasi vengono trattate, a due a due, come segnali **differenziali** e trasformate in valori **single-ended** (rispetto alla massa comune della scheda) proporzionali alle tensioni concatenate del motore.
Lo schema elettrico di questa parte della scheda è riportato nell'illustrazione 25.

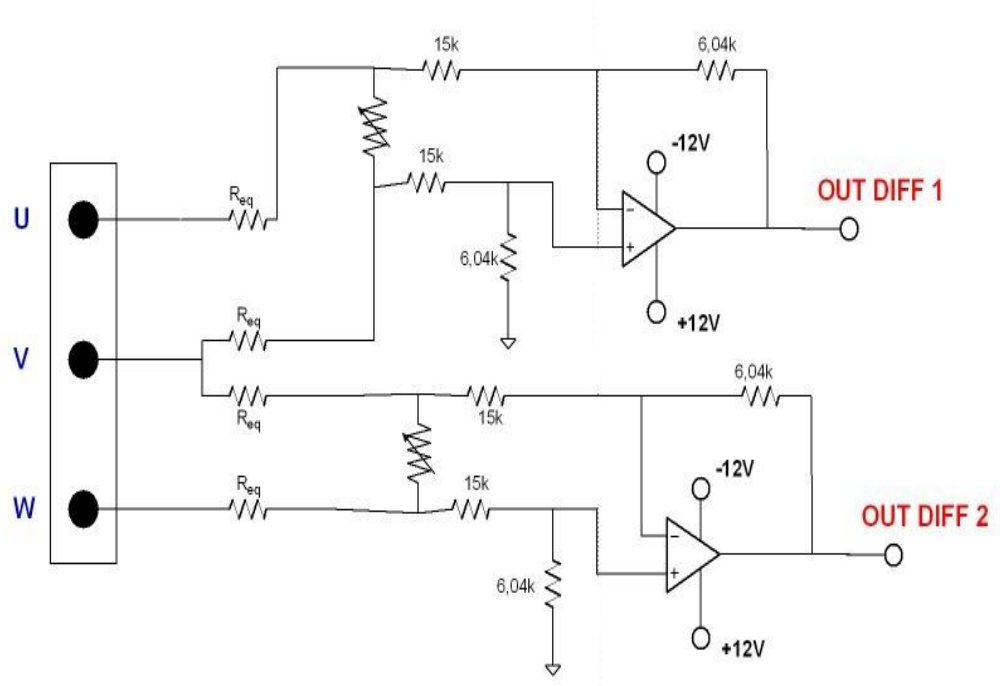


Illustrazione 25: Schema elettrico dello stadio di riduzione

Attraverso una rete resistiva le fasi del motore arrivano in ingresso ad un amplificatore operazionale che calcola la differenza tra di esse e restituisce un segnale, proporzionale alla tensione concatenata del motore, e ridotta di un fattore totale **1/200** (fattore di riduzione della rete resistiva + amplificazione dell'operazionale).

La presenza dei trimmer è resa necessaria dal fatto che le 2 reti resistive devono essere perfettamente bilanciate al fine di ridurre con lo stesso fattore gli ingressi. Questo trimmer viene tarato in fase di collaudo della scheda.

Dalle tre fasi del motore **U**, **V**, **W** vengono calcolate le 2 concatenate (**OUT DIFF 1**=V_{VU} e **OUT DIFF 2**=V_{WV}): quindi sono presenti 2 amplificatori differenziali.

2.4.2 Stadio di alimentazione

La scheda analogica è alimentata a 24V utilizzando il connettore P1. Attraverso questa tensione continua, fornita dall'esterno, vengono derivate le alimentazioni per gli integrati utilizzati negli altri stadi: **+/-12V** e **+/-8V**.
Questo stadio è stato realizzato attraverso un convertitore statico isolato in configurazione **FLYBACK** multi-uscita, schematizzato in figura 26.

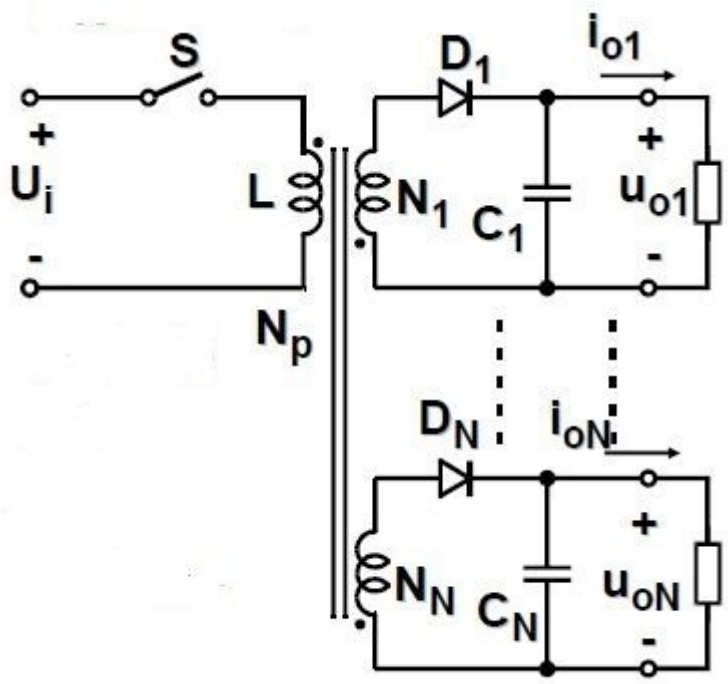


Illustrazione 26: Convertitore Flyback multi-uscita

La presenza del trasformatore assicura l'isolamento galvanico tra ingresso e uscite. Gli elementi reattivi utilizzati nei circuiti di alimentazione hanno dimensioni inversamente proporzionali alla frequenza di commutazione dell'interruttore. Per questo si cerca di rendere il periodo di lavoro del convertitore minore possibile, trovando un compromesso con la potenza dissipata. Nella configurazione flyback il trasformatore è soggetto alla frequenza di commutazione dell'interruttore (un MOSFET) in serie, che è nell'ordine di qualche decina di kHz, ed ha quindi il vantaggio di avere dimensioni molto contenute rispetto ai trasformatori che lavorano alla frequenza di rete.

Esso è in realtà un mutuo induttore, in quanto al suo interno vengono introdotti dei traferri che gli permettono accumulare energia, a differenza dei normali trasformatori. Le equazioni che legano ingressi e uscite del mutuo induttore e il suo simbolo elettrico sono riportate in figura 27.

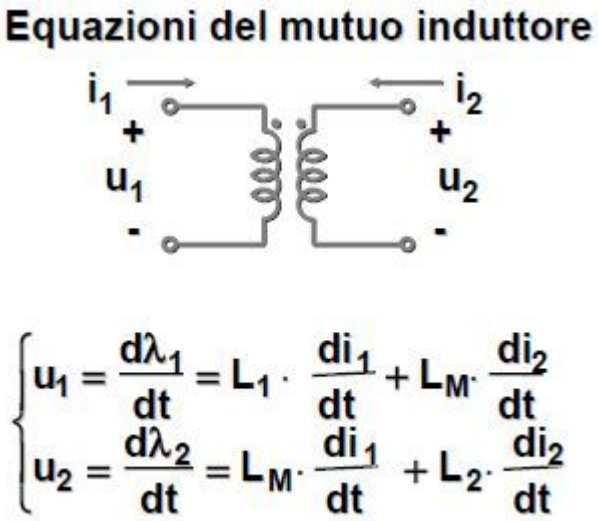


Illustrazione 27: Equazioni del mutuo induttore

Questo configurazione ha il vantaggio di non utilizzare ulteriori induttanze in uscita e permette di realizzare strutture multi-uscita aggiungendo solamente un diodo e un condensatore.

Il funzionamento del Flyback è molto simile a quello di un normale Buck-Boost.

Il suo fattore di conversione (= relazione statica tra la tensione di ingresso e quella di uscita) è:

$$\text{CCM } (I_o > I_{olim}) \qquad M = \frac{U_o}{U_i} = \frac{N_2}{N_1}\frac{\delta}{1-\delta}$$

$$\text{DCM } (I_o < I_{olim}) \qquad M = \frac{U_o}{U_i} = \frac{I_N}{I_o}\delta^2$$

$$I_N = \frac{U_i}{2 f_S L_1}$$

$$I_{olim} = I_N \frac{N_1}{N_2}\delta(1-\delta)$$

Il flyback può funzionare in **CCM** (Continuos Conduction Mode) se la corrente di carico Io non si annulla mai all'interno del periodo di switching ($I_o > I_{olim}$), altrimenti può funzionare in **DCM** (Discontinuos Conduction Mode) ($I_o < I_{olim}$).

Si utilizza generalmente in DCM perché si sfrutta l'intera escursione del flusso, e quindi il nucleo del mutuo induttore risulta più piccolo, e perché si ottengono migliori caratteristiche dinamiche.

L'interruttore è realizzato con un MOSFET (FQD5P10) comandato attraverso un'onda quadra, fornita al suo gate, di frequenza e duty-cycle costanti per ottenere il fattore di conversione voluto e mantenerlo costante durante il funzionamento della scheda. Questa onda quadra di comando è fornita dall'integrato NE555 della STMicroeletronics.

L' NE555 è uno dei circuiti integrati più utilizzati per la costruzione di timers ed oscillatori, sia astabili che monostabili. Nella **configurazione astabile**, il cui schema elettrico è illustrato nella figura 28, consente di generare una tensione che oscilla tra 2 valori , quindi un'onda quadra di frequenza e duty cycle desiderato e determinato dal valore dei componenti ad esso collegati.

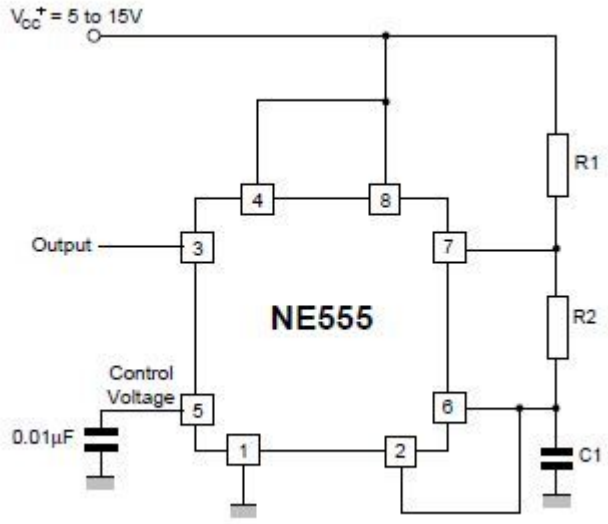


Illustrazione 28: Configurazione astabile dell'integrato NE555

Questa configurazione, nello stadio di alimentazione, è realizzata come nell'illustrazione 28: il circuito è "auto-triggerato" e funziona in free-run come un multivibratore. La capacità C_1 si carica attraverso le resistenze R_1 e R_2 e si scarica solamente attraverso R_2. Il duty-cycle dell'onda quadra generata può essere fissato con precisione attraverso il valore di questi due resistori. C1 si carica e scarica tra $1/3V_{cc}$ e $2/3V_{cc}$ e la frequenza è indipendente dalla tensione di alimentazione.
Il tempo t_1 in cui l'uscita dell'integrato è "ALTA" è dato da:

$$t_1 = 0.693\ (R_1 + R_2)\ C_1$$

mentre t_2 in cui è "BASSA":

$$t_2 = 0.693\ (R_2)\ C_1$$

Quindi il periodo (e quindi la sua frequenza) è:

$$T = t_1 + t_2 = 0.693\ (R_1 + 2R_2)\ C_1$$

$$f = \frac{1}{T} = \frac{1.44}{(R_1 + 2R_2)\ C_1}$$

Il duty-cycle è quindi:

$$D = \frac{R_2}{R_1 + 2R_2}$$

Con i valori utilizzati nella scheda analogica si ha un'onda quadra (che comanda il gate del MOSFET) di frequenza = 44kHz e duty-cycle = δ = 43%.

In questo modo si ottengono, dalle 2 uscite del Flyback, una tensione positiva di 16V e una negativa da -16V. Attraverso 4 regolatori lineari si hanno poi +12V, +8V, -12V, -8V.
Un regolatore lineare è un circuito che provoca una caduta di tensione variabile tale da compensare la tensione continua al suo ingresso per fornire una tensione (minore) continua stabile in uscita. Esso di solito comporta non trascurabili dissipazioni di potenza.
La caduta di tensione avviene di solito su un dispositivo attivo a semiconduttore, comandato da un adeguato circuito di controllo a retroazione che compara la tensione di uscita con una tensione di riferimento e comanda l'elettrodo di controllo del semiconduttore regolatore. Molto spesso (come in questo caso) l'intero circuito descritto sopra è contenuto in un integrato che contiene il riferimento di tensione e altre funzioni come limitazioni di corrente o protezioni per corti in uscita.

2.4.3 Stadio di uscita di conversione in corrente

Le 2 uscite della scheda analogica 4108 sono 2 segnali in corrente proporzionali alla media mobile delle tensioni concatenate del motore sul semiperiodo di PWM. Questo stadio realizza la conversione tensione/corrente attraverso un circuito analogico ad amplificatori operazionali.

Molti sistemi circuitali richiedono sorgenti di corrente controllate in tensione di grande precisione, soprattutto in presenza di carichi variabili. L'approccio tradizionale basato su un unico amplificatore e su una manciata di componenti passivi attorno (si veda per approfondimenti il riferimento bibliografico [5]) può ingenerare errori causati dalle caratteristiche non lineari intrinseche di taluni dispositivi che mostrano spesso valori non trascurabili di guadagno ad anello aperto, reiezione di modo comune, corrente di polarizzazione e tensione di offset. I progetti basati su amplificatori operazionali, per esempio, possono aver bisogno di resistori di precisione per poter determinare il guadagno e opportuni condensatori per stabilizzarlo. Di conseguenza, alcuni circuiti finiscono poi per fornire in uscita correnti che non sono direttamente proporzionali alla tensione d'ingresso.

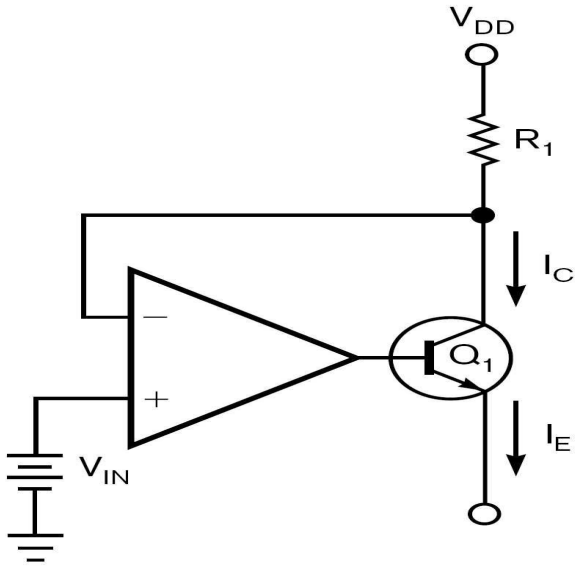


Illustrazione 29: Convertitore V/I ad un op. amp.

Il convertitore tensione/corrente realizzato come nell'illustrazione 29 ha una tensione d'ingresso V_{IN} costante. Esso sfrutta la circostanza che la corrente di collettore rimane sempre approssimativamente uguale alla corrente dell'emettitore e così può garantire un'erogazione di corrente costante, monodirezionale e proporzionale alla V_{IN}. Usando questo principio si può realizzare il circuito generatore di corrente controllato in tensione mostrato nell'illustrazione 30 che usa due amplificatori e due transistor e fornisce un'accuratezza dello 0,01%.

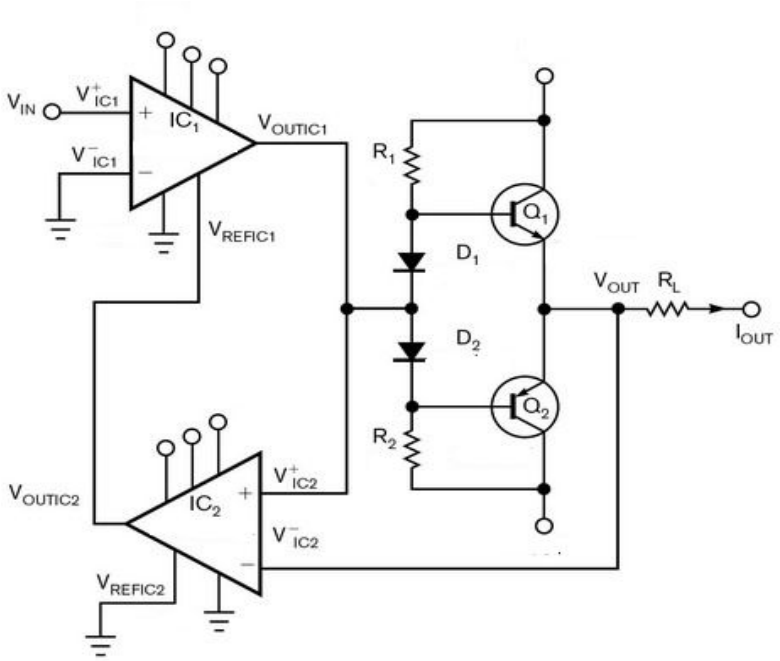


Illustrazione 30: Conv. V/I a 2 op. amp.

I due amplificatori si occupano del controllo della polarizzazione e della correzione degli errori, ma non influiscono sulla corrente d'uscita, giacché di questa si occupano i due transistor Q1 e Q2, il cui guadagno determina anche il livello della corrente erogata. Nei due amplificatori può comunque essere regolato il guadagno da 0 a 10000 fino a decidere la miglior accuratezza possibile anche sui livelli d'ingresso di 1 mV, bastando connettere un solo resistore di valore opportuno fra gli ingressi degli integrati IC1 e IC2 per ottenere il guadagno desiderato.

Il primo amplificatore IC1 controlla la tensione di base nello stadio d'uscita

push-pull, mentre i resistori e i diodi a loro attorno decidono la polarizzazione dei transistor Q_1 e Q_2, eliminando la distorsione d'intermodulazione. L'IC_2 provvede alla correzione errori e determina il voltaggio di polarizzazione base/tensione. L'errore di tensione si può misurare in forma differenziale sulla giunzione fra D_1 e D_2 rispetto alla tensione d'uscita e questo valore può essere riportato al pin di riferimento sull'IC_1 dove si somma alla tensione d'ingresso. Tutto ciò fornisce come risultato una corrente d'uscita direttamente proporzionale alla tensione d'ingresso e garantisce un'accuratezza dello 0,01%.

Nel circuito realizzato nella scheda analogica di misura è utilizzato un solo transistor pnp poiché l'alimentazione è unipolare.

Inoltre il circuito è in configurazione invertente: in questo modo si annulla l'inversione presente nello stadio precedente di integrazione.

Questo circuito garantisce un ampio range d'uscita e ha il pregio di erogare una corrente direttamente proporzionale alla tensione d'ingresso, molto lineare ed estremamente precisa.

2.5 Condizionamento e interfaccia dei segnali di comando

Per ottenere il corretto segnale di sincronismo, fornito alla scheda analogica attraverso il connettore bipolare P5, è stata predisposto un ulteriore circuito realizzato in laboratorio su un "millefori" con componenti PTH.

Questo circuito permette, attraverso l'utilizzo di un monostabile, di modificare il segnale che arriva dal FCPS ed ottenere un segnale in fase con esso ma con duty-cycle al 50%.

L'integrato utilizzato è il **CD4538B** della National Semiconductor. Esso è un multivibratore monostabile con trigger indipendente e controllo di reset. Il suo schema a blocchi è illustrato in figura 31.

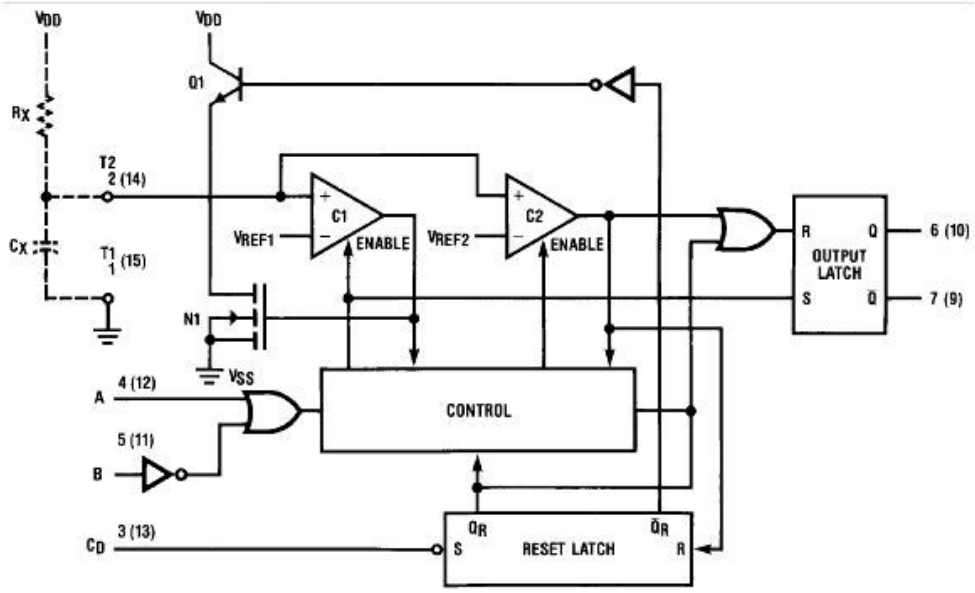


Illustrazione 31: Schema a blocchi dell'integrato CD4538B

Il segnale dal FCPS, che permette al sistema di acquisizione di sincronizzarsi con il periodo di PWM, è un segnale digitale da 0V (livello basso) a 5V (livello alto). Esso è normalmente alto, e all'inizio di ogni periodo di PWM (che in questo progetto è a 12kHz) lancia un impulso a 0V della durata di 200 ns.

E' il segnale verde illustrato nella misura riportata in figura 34.

Il monostabile deve riconoscere il fronte di discesa di questo segnale, denominato nel datasheet l'***evento di trigger***, portare l'uscita Q alta per 42µs, cioè per metà periodo (duty cycle =50%), e tornare alta al prossimo evento di trigger. In questo

modo si otterrà il segnale di sincronismo di cui necessita la scheda analogica per funzionare correttamente, in fase con il periodo di PWM e con duty-cycle al **50%**.

Il circuito nel quale è inserito il monostabile è stato configurato come illustrato nello schema elettrico riportato nell'illustrazione 32.

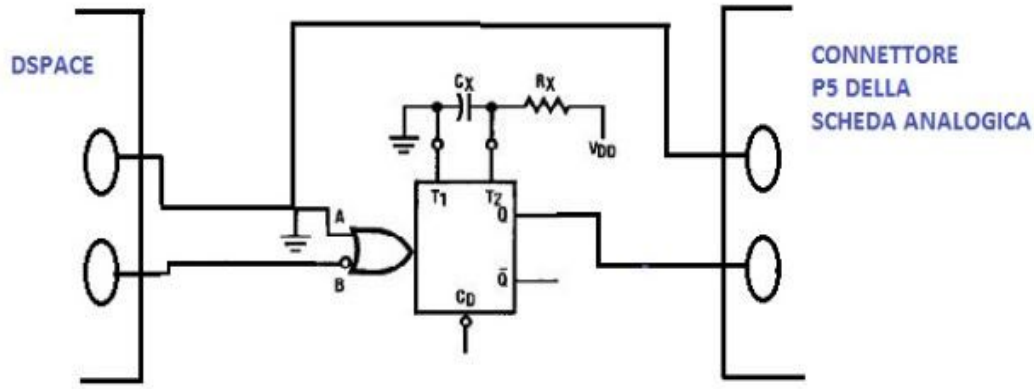


Illustrazione 32: Connessione del monostabile con il resto del sistema

Prima che avvenga l'evento di trigger (= fronte di discesa del segnale in uscita dal FCPS), il monostabile è nello stato di riposo con l'uscita Q a zero e C_x completamente caricato a V_{dd}. Quando sul pin B si verifica un evento di trigger valido, Q si porta ad un livello di tensione pari a V_{dd} e C_x si scarica fino a 0V. A questo punto C_x è abilitato a caricarsi attraverso la resistenza R_x finché non raggiunge una tensione di riferimento che porta al reset del latch in uscita, e quindi Q ritorna a 0V.

Quindi il tempo in l'uscita Q rimane a livello logico alto è determinata dai valori scelti per il resistore R_x e il condensatore C_x. Per avere in uscita un segnale da 0V a 5V, basta porre V_{dd}=5V e il pin A del monostabile a massa (comune fra scheda analogica e FPCS).

La durata del tempo in cui l'uscita Q è a 5V è pari a R_x*C_x.

Per avere un'onda quadra in uscita esattamente con duty-cycle = 50%, si è scelto un condensatore da 4,7nF e per R_x un resistore da 100kΩ in parallelo ad un trimmer (0÷10kΩ) tarabile quando la scheda è accesa per ottenere esattamente il duty-cycle cercato.

I risultati ottenuti sono visibili, attraverso le misure effettuate con l'oscilloscopio, nelle illustrazioni 33, 34, 35: il segnale verde rappresenta l'ingresso del monostabile, mentre quello giallo la sua uscita. Queste prove sul circuito con il monostabile sono state effettuate con una Vdd =8V, e generano esattamente la forma d'onda con duty-cycle al 50% desiderata.

Nella figura 33 l'ingresso è fornito da un generatore di funzioni per provare il circuito, mentre nelle figure 34 e 35 l'ingresso arriva direttamente dal FCPS.

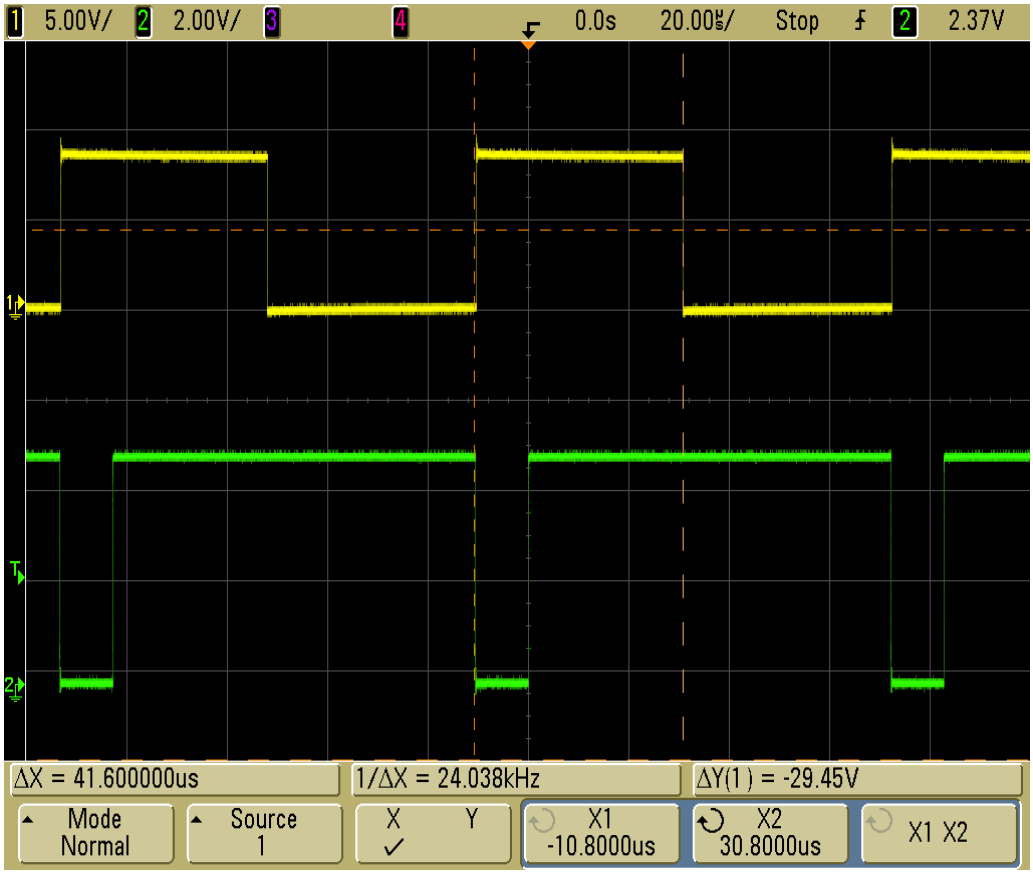


Illustrazione 33: VERDE= ingresso fornito da un generatore di funzioni GIALLO= uscita del monostabile

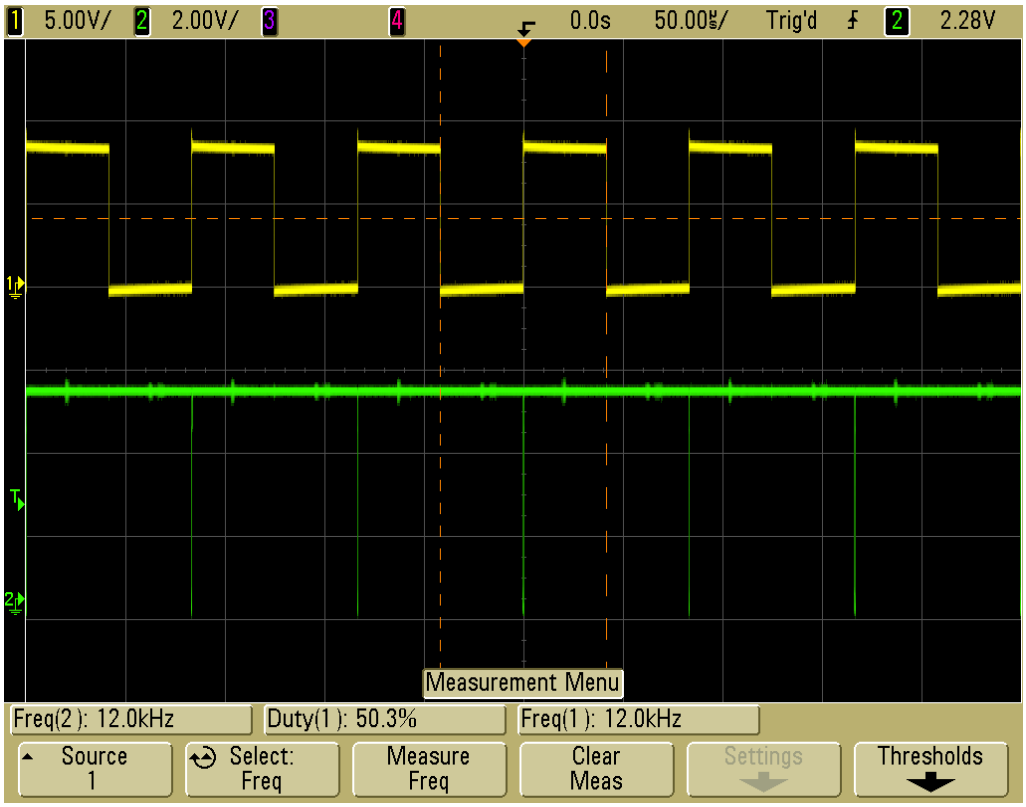


Illustrazione 34: VERDE= segnale fornito dalla DSPACE per sincronizzare il sistema GIALLO= uscita del monostabile

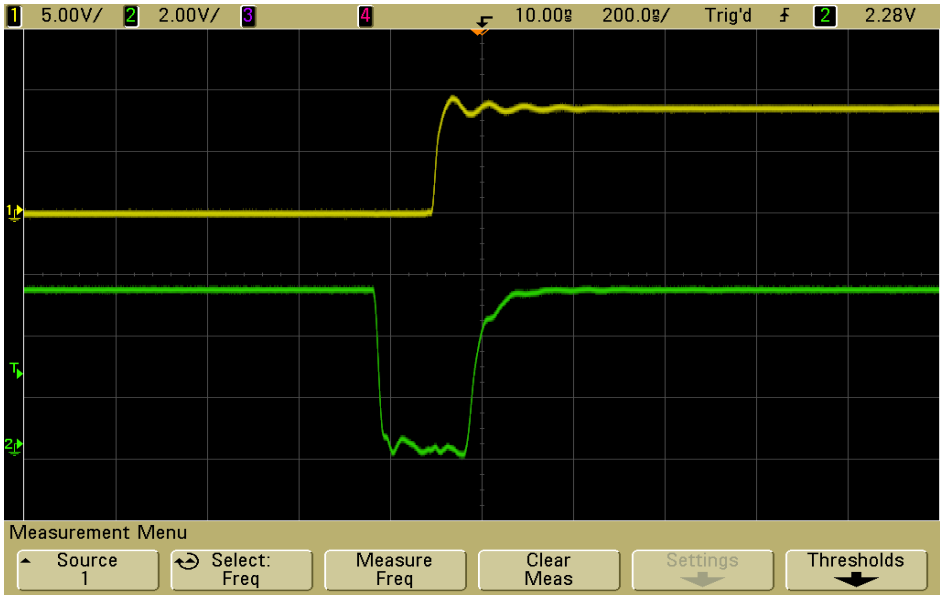


Illustrazione 35: Particolare del fronte di discesa del segnale DSPACE e commutazione del monostabile

3. MISURA DELLE TENSIONI: LA SOLUZIONE DIGITALE

3.1 Studio dell'organizzazione della scheda di sviluppo FPGA

Una **FPGA** (Field Programmable Gate Array) è un insieme di circuiti integrati che contiene molte celle logiche identiche, e ogni cella può essere programmata ad effettuare una determinata funzione logica. Le interconnessioni tra le celle sono programmabili.

Questi dispositivi si distinguono per l'elevata velocità di calcolo e sono utilizzati per la realizzazione di funzioni logiche, macchine sequenziali sincrone e asincrone, reti combinatorie e gestione di led e display.

L'FPGA utilizzata per questo progetto è una ***CYCLONE III*** della Altera, montata su una scheda di test realizzata dalla stessa azienda, la **CYCLONE III FPGA STARTER BOARD**.

La Cyclone III Starter Board fornisce un supporto hardware per sviluppare qualsiasi funzionalità possibile con una FPGA attraverso un ambiente di sviluppo del software per la programmazione del dispositivo digitale. Essa facilita il programmatore ad utilizzare questo tipo di dispositivi digitali attraverso semplici progetti e dimostrazioni.

Questa scheda può essere supportata, attraverso un **connettore HSMC** (High Speed Mezzanine Card), da altre schede che permettono di espandere ulteriormente le funzionalità del sistema digitale.

Nell'illustrazione 36 è possibile avere una visione dell'intero circuito. Sono messi in evidenza i connettori della scheda per interfacciarsi con l'esterno e tutti i principali dispositivi elettronici installati.

Alcune figure illustrate in questo capitolo sono state tratte dal "Reference Manual" della *Cyclone III FPGA Starter Board* (courtesy of Altera®).

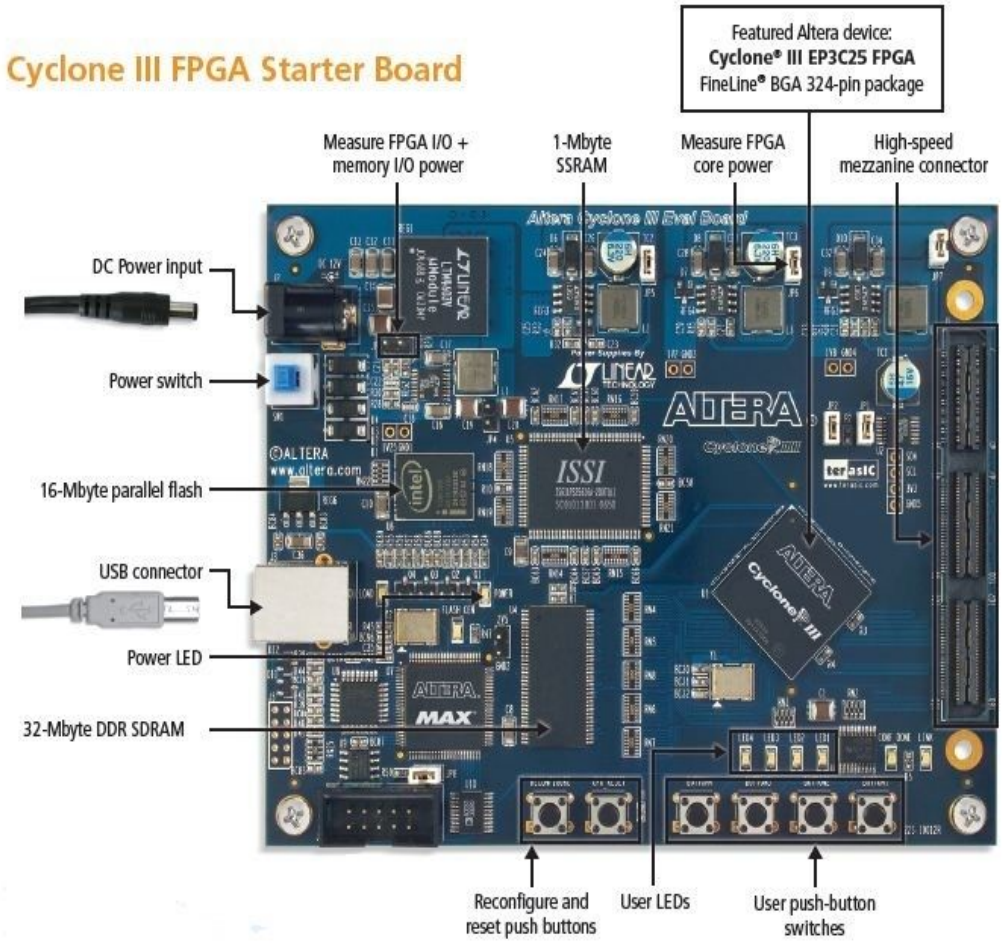


Illustrazione 36: Cyclone III FPGA Starter Board

Gli elementi principali della Cyclone III Starter Board sono:

- l'FPGA Altera Cyclone III EP3C25
- un connettore HSMC per l'interfacciamento con altre schede
- una memoria DDR SDRAM a 32 Mbyte
- una memoria Flash a 16 Mbyte per la configurazione dell'FPGA e il salvataggio dei dati allo spegnimento
- una memoria SSRAM ad alta velocità da 1Mbyte
- 4 pulsanti configurabili via software
- 4 LED programmabili

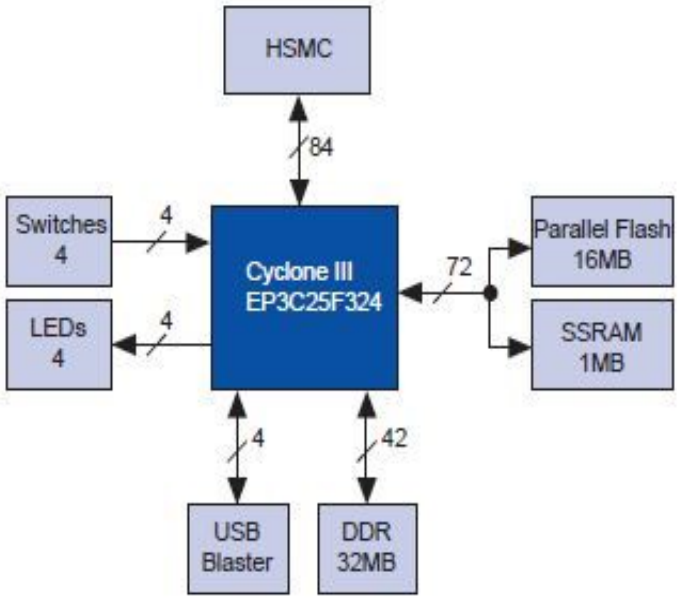


Illustrazione 37: Diagramma a blocchi dell'FPGA

L'FPGA, realizzata all'interno di un package da 324 pin, contiene 25000 elementi logici programmabili, 0,6 Mbits di blocchi di memoria e 16 blocchi moltiplicatori (18x18).

Di seguito verranno descritti in maggior dettaglio gli elementi della scheda (in evidenza nell'illustrazione 38) che risultano più interessanti per lo sviluppo di questo progetto.

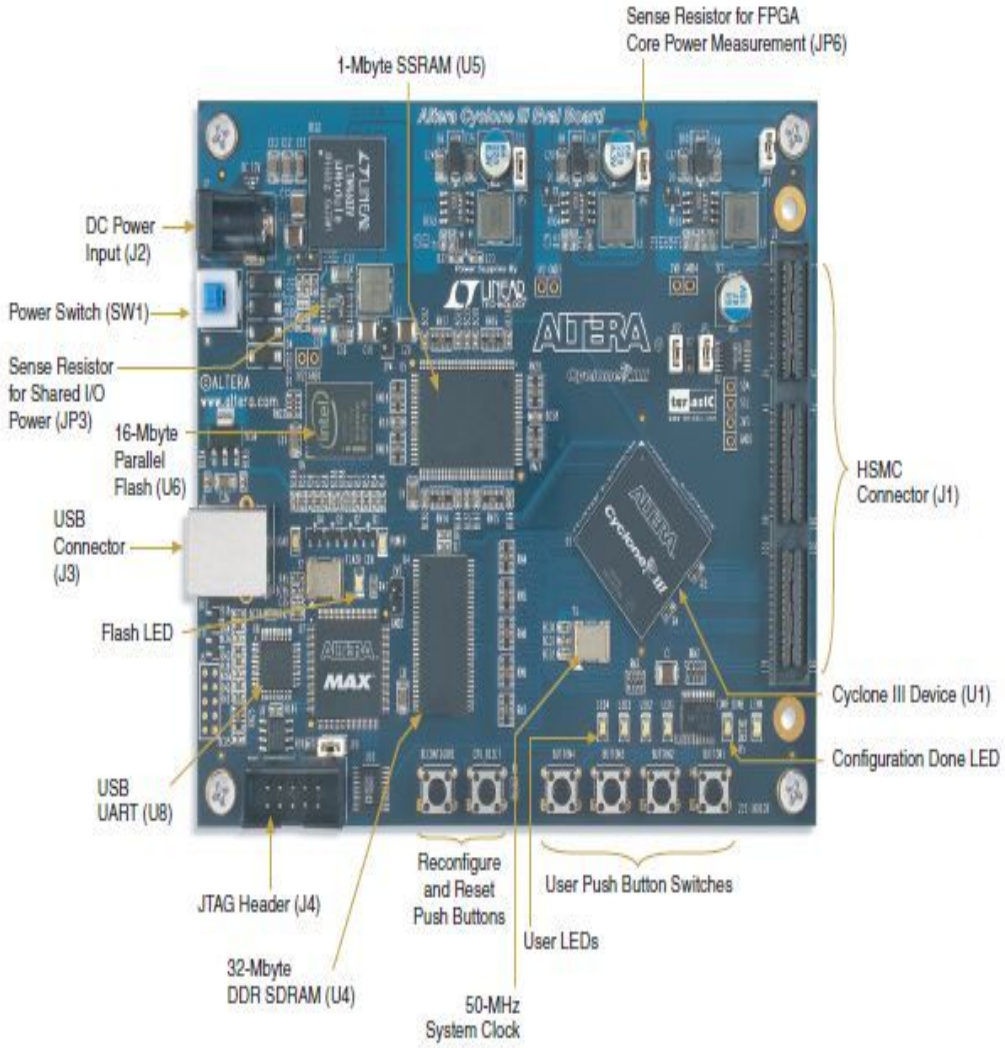


Illustrazione 38: Componenti della Cyclone III Starter Board

3.1.1 Circuito di clock della Cyclone III Starter Board

Il circuito di clock della Cyclone III FPGA Starter Board fornisce un segnale a **50 Mhz,** e tutti gli altri clock distribuiti all'interno della scheda sono generati da esso attraverso dei PLL (anelli ad aggancio di fase). Questi sono:

- clock per la memoria flash
- clock per la memoria SSRAM
- clock fornito al connettore HSMC. Attraverso i PLL dal clock interno possono essere generati clock a frequenze maggiori o minori dei 50 Mhz ed essere messi a disposizione per le schede "figlie" connesse al HMSC.

Nell'illustrazione 39 sono messi in evidenza le connessioni del circuito di clock FPGA con il resto del sistema. In evidenza i pin del connettore HMSC a cui arrivano i segnali generati dal PLL.

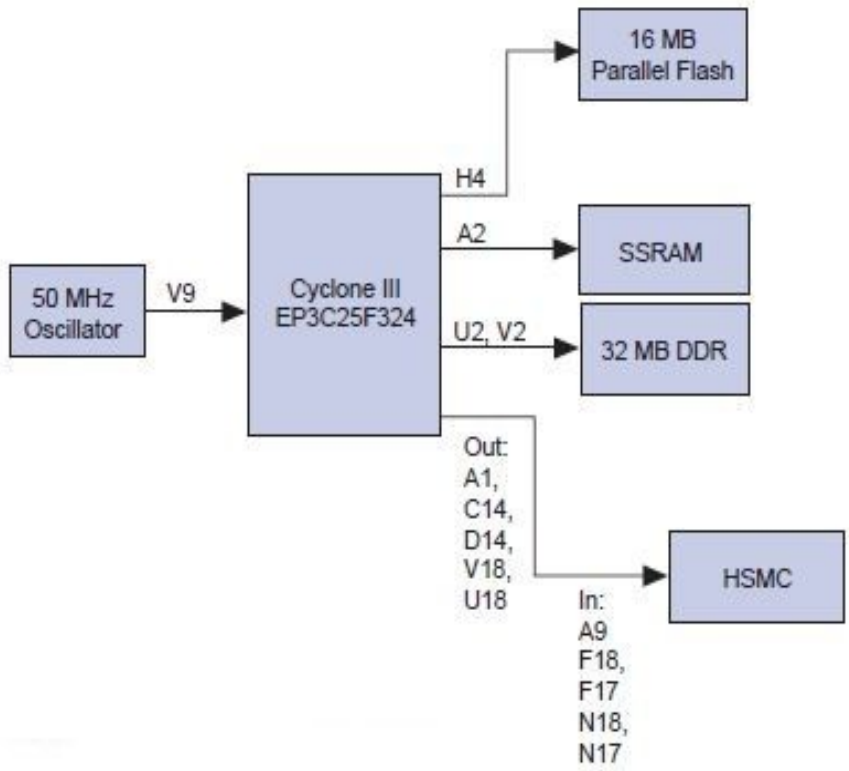


Illustrazione 39: Schema a blocchi del circuito di clock della Cyclone III Starter Board

3.1.2 Interfacce della Cyclone III FPGA Starter Board

I blocchi di interfaccia verso l'utente e verso connessioni con le altre schede sono:

- l'interfaccia USB
- il connettore di espansione HSMC
- pulsanti
- LED

Connettendo il cavo USB (provvisto nel kit) al connettore J3 sulla scheda e, l'altro capo, ad una porta **USB** di un computer, è possibile programmare l'FPGA ed interfacciare l'ambiente di sviluppo del software con essa.
E' messo a disposizione, per programmare l'FPGA e comunicare con essa, anche il connettore JTAG J4 (non è stato utilizzato in questo progetto).

Il connettore **HMSC** è una versione modificata del connettore standard ad alta velocità Samtec, per fornire una maggiore integrità dei segnali scambiati tra le schede connesse. La Starter Board provvede a fornire, con questo connettore, sia 12V che 3,3V per alimentare le altre schede "figlie" connesse. In totale è formato da 84 pin di I/O, ognuno dei quali è bidirezionale (tranne i segnali di clock) e del tipo "***2,5V***": a livello logico alto per tensioni comprese fra 1,7V e 4,1V, basso per tensioni tra -0,5V e 0,7V. La lista dei nomi dei pin (indispensabile durante la programmazione per richiamare ingressi e uscite) è nel manuale della scheda.

I pulsanti che possono interagire con l'utente in base alla programmazione della FPGA sono 4. Appena vengono premuti forniscono un segnale logico alto (=1) finché non vengono rilasciati. Poi sono presenti sulla scheda altri 2 pulsanti non programmabili: il pulsante System-reset e il pulsante User-reset.
Il primo serve per forzare una ri-configurazione dell'FPGA attraverso la memoria flash. Il secondo è un input diretto per il dispositivo Cyclone III per resettare i progetti caricati nel dispositivo (gestibile via software).

Illustrazione 40: Pulsanti della FPGA Cyclone III Starter Board

Anche i LED gestibili dal programma caricato nell'FPGA sono 4. Altri 3 LED (non programmabili) indicano rispettivamente:

- Power LED: si illumina quando la scheda è alimentata attraverso il connettore J2
- Configuration LED: è acceso quando la programmazione della FPGA è avvenuta con successo
- Flash Signal LED: si illumina ogni volta che il programma accede alla memoria flash

Illustrazione 41: LED della FPGA Cyclone III Starter Board

3.2 Scheda di interfaccia A/D e D/A

Gli ingressi che dovranno essere elaborati dalla "*Cyclone III FPGA Starter Board*" sono le 2 tensioni concatenate del motore in alternata. Questi due segnali vengono prelevati dallo stadio di riduzione della scheda analogica di misura, che provvede anche ad attenuarli di un fattore 1/200.

Le 2 tensioni concatenate sono segnali ANALOGICI. I sistemi digitali basati su FPGA lavorano solamente con segnali discreti, quindi per poter elaborare questo tipo di ingressi è necessaria una conversione ANALOGICO/DIGITALE e successivamente le uscite dovranno essere convertite nella maniera inversa.

Per realizzare tutto questo è stato scelto di utilizzare una scheda, prodotta dalla Terasic S.p.A., sulla quale sono presenti 2 convertitori A/D e 2 convertitori D/A: la ***THDB_ADA***, illustrata nella figura 42.

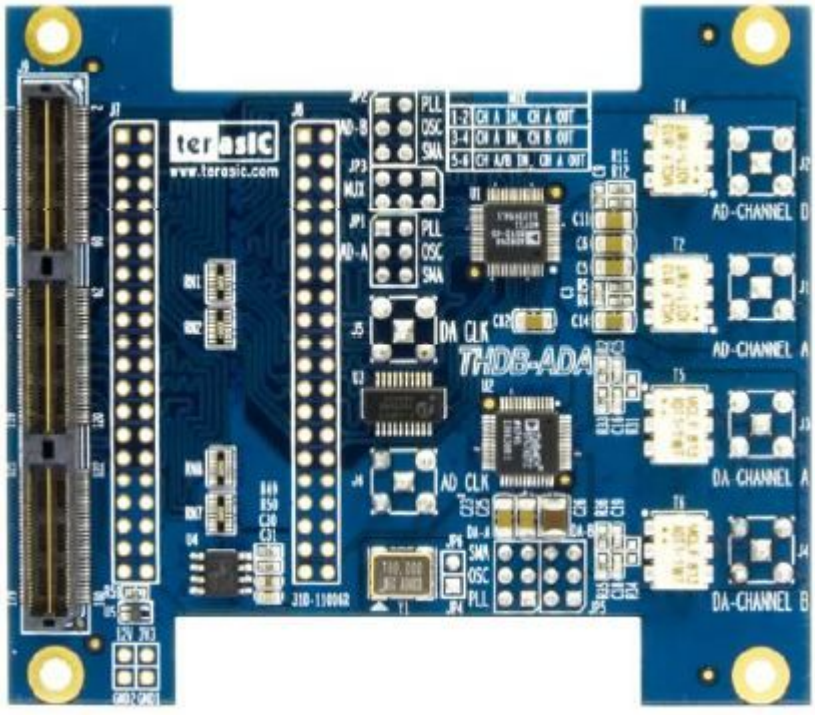


Illustrazione 42: Scheda THDB_ADA

La scheda “figlia” THDB_ADA e la “FPGA Cyclone III Starter Board” sono connesse attraverso il connettore HMSC, come si può vedere nell'illustrazione 43:

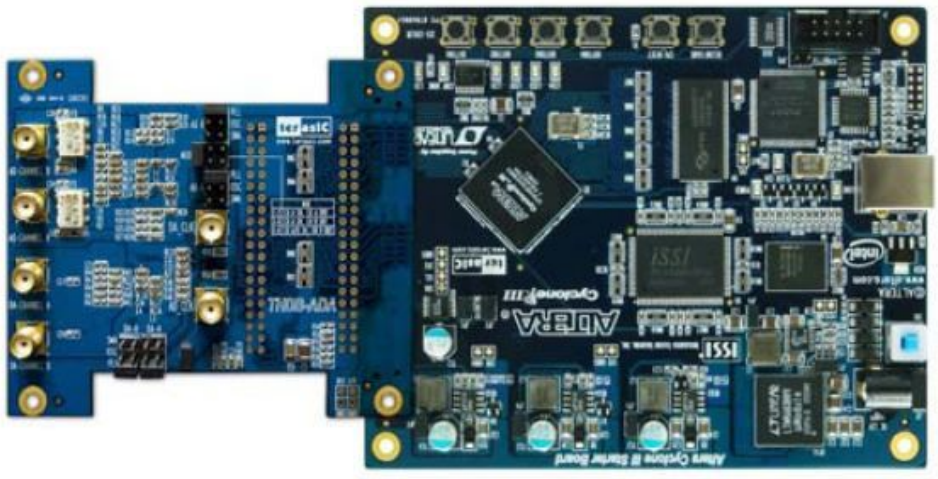


Illustrazione 43: THBD_ADA connessa alla FPGA Cyclone III Starter Board

Per i 2 ingressi e le 2 uscite ci sono 4 connettori BNC. I **connettori BNC** sono una famiglia di connettori unipolari a baionetta usati per l'intestazione di cavi coassiali. Sono adatti per linee con un'impedenza di 50 e 75 ohm.

I due canali A/D hanno 14 bit di risoluzione e speed-rate (=frequenza di campionamento) può arrivare fino a 65 MSPS (=65MHz). Anche i canali D/A hanno 14 bit di risoluzione, ma lo speed-rate in questo caso può arrivare fino a 125 MSPS.

Sulla THDB_ADA è presente un oscillatore a 100 MHz che può essere utilizzato dai convertitori A/D e D/A come sorgente di clock. Altrimenti la scheda, attraverso l'inserimento di alcuni jumper, può ricevere i clock per il campionamento dei segnali attraverso una sorgente esterna (collegata attraverso altri 2 connettori BNC, uno per il campionamento A/D e l'altro per il D/A). Anche attraverso la scheda principale Cyclone III può essere generato un clock con i PLL presenti nella FPGA e trasmesso, attraverso il connettore HMSC, ai 2 convertitori.

Gli integrati utilizzati nella THDB_ADA per il campionamento e la conversione dei segnali sono i **AD9248** e **AD9767** della Analog Devices.

E' indispensabile capire come lavorano nel dettaglio questo tipo di integrati per poter elaborare nella maniera corretta i segnali digitali che andranno all'FPGA.

3.2.1 L'integrato AD9248 per la conversione A/D

L'integrato AD9248 è un ***convertitore analogico/digitale*** duale a 14 bit di risoluzione e alimentato a 3V (tensione sopportata da 2,7V a 3,6V). Esso è caratterizzato inoltre da 2 amplificatori **Sample-And-Hold** che possono campionare segnali analogici fino ad una frequenza di 65MHz.
L'architettura utilizzata per la conversione è una multi-stadio differenziale a pipeline che comprende una logica interna per la correzione degli errori che garantisce nessun "missing code" su tutto il range di temperatura a cui può lavorare l'integrato (-40°C-+85°C). Due distinti segnali di clock in ingresso all'integrato controllano tutti i cicli di conversione esterni. I segnali digitali in uscita possono essere sia nel formato "*complemento a due*" sia nel formato "*offset binary*". E' inoltre presente un pin che indica se si è verificata la condizione di overflow (=segnale analogico da convertire maggiore dell'alimentazione dell'integrato) dei segnali analogici convertiti. Il consumo di potenza dipende dalla frequenza di lavoro, e può variare dai 600mW (@65MHz) ai 180mW (@20MHz).
Le ampiezze dei segnali digitali convertiti sono di 0,05V massimi per il livello "basso", e (V_{dd}-0,05)V minimi per il livello logico "alto".
Nell'illustrazione 44 è riportato il diagramma a blocchi dell'integrato dove sono messi in evidenza i pin utili per interagire con esso, mentre nel datasheet sono riportati anche a quali pin del connettore HSMC arrivano per poter essere richiamati nella programmazione software dell'FPGA.

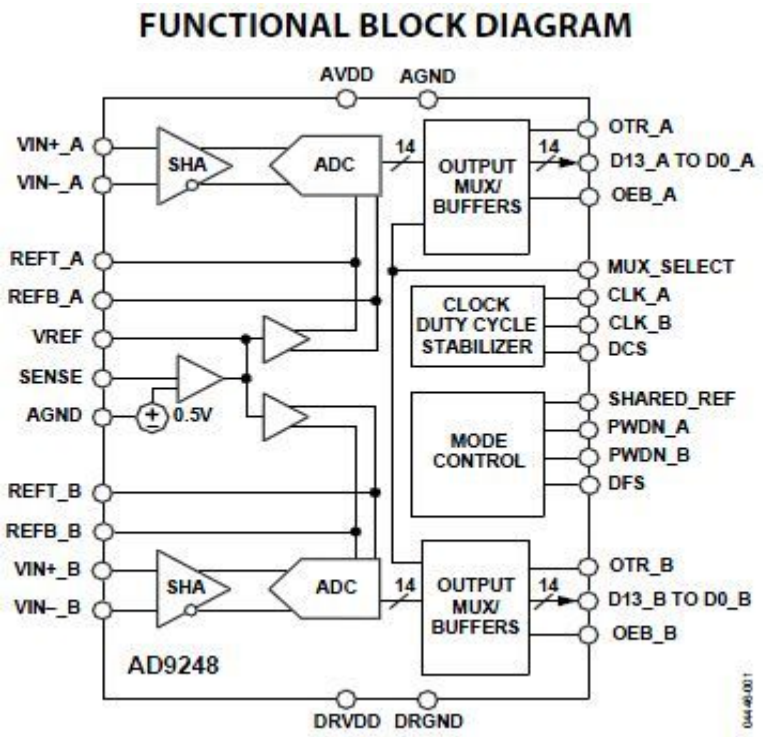


Illustrazione 44: Diagramma a blocchi dell'integrato AD9248

I circuiti equivalenti per gli ingressi e le uscite sono riportati in figura 45.

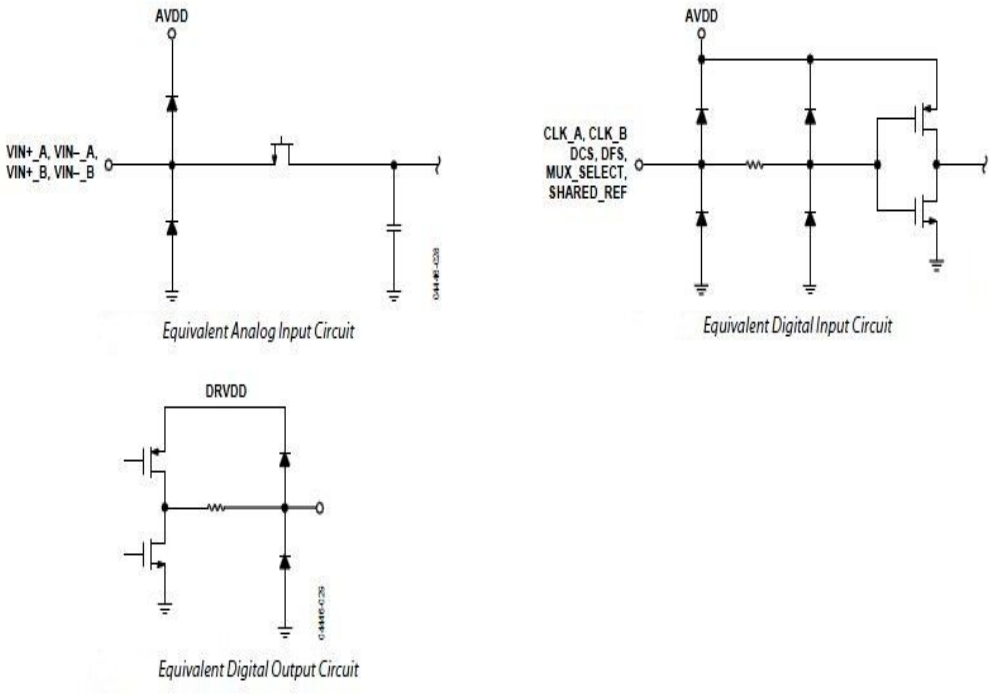


Illustrazione 45: Circuiti equivalenti I/O

Lo stadio di ingresso è costituito da un circuito “Sample-And-Hold” progettato per lavorare al meglio con segnali differenziali, ma che può essere configurato anche per lavorare con segnali single-ended, accoppiati AC o DC.

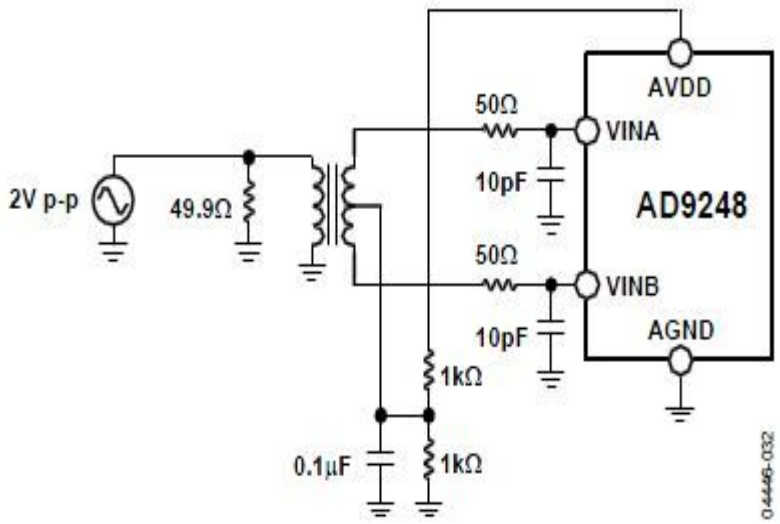


Illustrazione 46: Configurazione dello stadio di ingresso della scheda THDB_ADA

Nella configurazione dello stadio d'ingresso della scheda THDB_ADA, mostrata nell'illustrazione 46, il segnale analogico che arriva al connettore BNC è un segnale single-ended $2V_{pp}$ (al massimo), trasformato in segnale differenziale per far lavorare al meglio l'integrato AD9248 attraverso un trasformatore RF a presa centrale. Il trasformatore utilizzato (**ADT1-1WT)** ha un limite di banda inferiore di 0,4 MHz. Questo rappresenta un limite per la realizzazione del progetto finale, in quanto le tensioni concatenate del motore hanno una banda di frequenze fino alla continua.

Si dovranno quindi necessariamente effettuare delle modifiche hardware alla scheda.

3.2.2 L'integrato AD9767 per la conversione D/A

L'integrato AD9767 della Analog Devices è costituito da un convertitore digitale/analogico duale a 14 bit di risoluzione che può lavorare fino a 125MHz. Esso opera con un'alimentazione di 3,3V e dissipa una potenza di 380mW. **Le uscite analogiche dei DAC sono segnali in corrente, con un fondo scala di 20mA**.

Una configurazione di uscita **DIFFERENZIALE** per questo integrato è l'ideale per ottimizzare le sue prestazioni dinamiche. Essa può essere realizzata attraverso un trasformatore RF o attraverso un amplificatore operazionale. La configurazione a "trasformatore RF", illustrata in figura 47, è quella che garantisce migliori performance per applicazioni ad alta frequenza, ma permette solamente un accoppiamento AC dell'uscita.

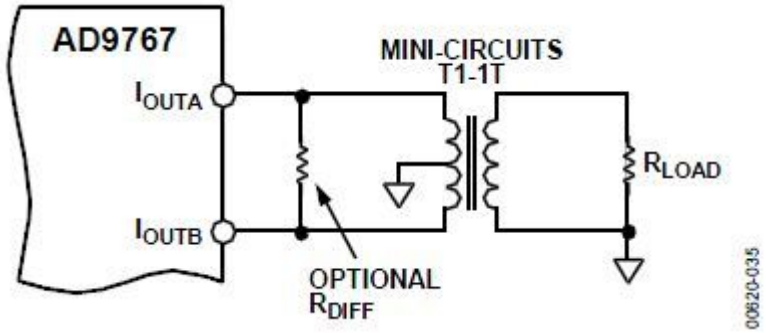


Illustrazione 47: Configurazione d'uscita del conv. D/A con il trasformatore RF

Questa configurazione dà un'eccellente reiezione di modo comune e permette l'isolamento galvanico. La presa centrale del primario deve essere connessa a massa. La tensione misurata al primario è simmetrica intorno al riferimento di tensione (massa) e su RLOAD si legge un valore di tensione (sempre riferito a massa) proporzionale all'uscita del convertitore.

La scheda THDB_ADA presenta questo tipo di circuito per l'interfacciamento delle uscite dei D/A, e quindi ai connettori BNC si avrà un segnale analogico single-ended e accoppiato AC.

Altrimenti è possibile realizzare anche una configurazione d'uscita **SINGLE-ENDED**, che può essere realizzata connettendo IOUTA e/o IOUTB ad una resistenza di carico appropriata (RLOAD), posta all'altro capo a massa.
Questa configurazione (illustrazione 48) è l'ideale per sistemi ad alimentazione unipolare che richiedono un accoppiamento DC e un'uscita in tensione riferita alla massa comune. Alternativamente un amplificatore operazionale in uscita può agire da convertitore corrente/tensione per trasformare IOUTA o IOUTB in un segnale di tensione.

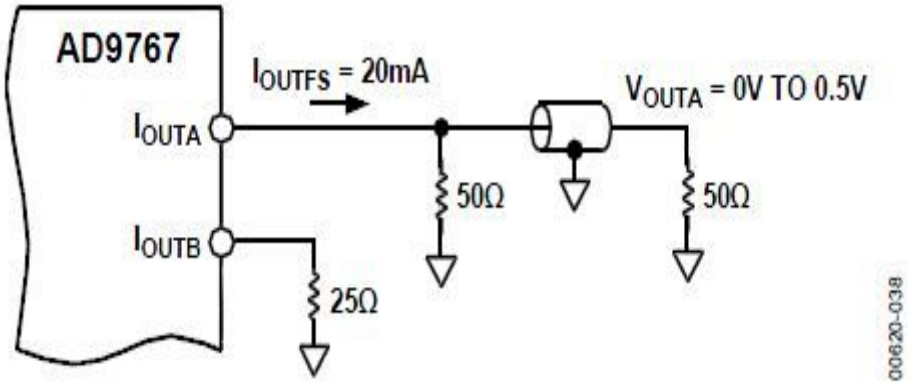


Illustrazione 48: Configurazione d'uscita del conv. D/A single-ended

Nella configurazione dell'illustrazione 48 in uscita dall'integrato AD9767 c'è un **segnale analogico in tensione che varia da 0V a 0,5V**. Infatti la corrente nominale di 20mA scorre lungo una "resistenza equivalente" di 25Ω, determinando il valore di tensione riferita a massa. IOUTB può essere connesso a massa o ad una resistenza di 25Ω. Possono essere utilizzati anche valori differenti di resistenze per ottenere il range di tensione desiderato.
Per le esigenze di questo progetto è necessario realizzare quest'ultima configurazione, in quanto i segnali trattati hanno anche componente continua e quindi è necessario un accoppiamento DC dell'uscita dei convertitori D/A.
Quindi si dovranno effettuare delle modifiche hardware anche per le uscite della scheda THDB_ADA.

3.3 Studio del sistema di sviluppo Quartus II

Il software utilizzato per la realizzazione del programma che andrà poi ad essere memorizzato nella FPGA per svolgere le funzioni desiderate è denominato ***Altera® Quartus® II***. E' prodotto dalla stessa casa produttrice della FPGA e fornito nel kit assieme alla Cyclone III Starter Board, assieme a tutti i manuali per il suo utilizzo. I suoi aggiornamenti sono scaricabili gratuitamente dal sito dell'azienda e si può installare in un normale PC, sia con i sistemi operativi di Microsoft, sia in ambiente Linux.

Il programma **Quartus II** fornisce un ambiente di sviluppo software multi-piattaforma e completo per sistemi SOPC (System On a Programmable Chip) realizzati dall'Altera. Esso include soluzioni realizzative per tutte le fasi di un ciclo di programmazione dei dispositivi logici programmabili Altera:

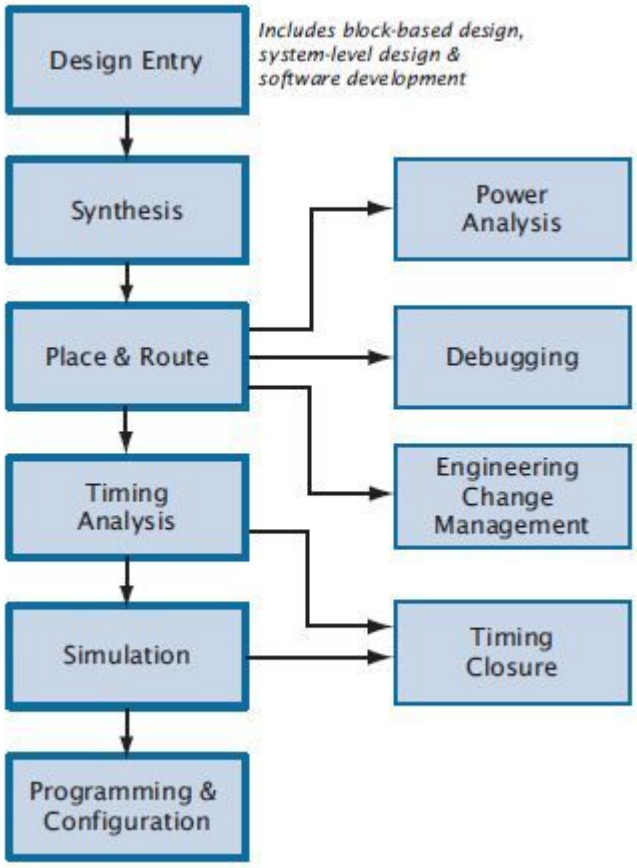


Illustrazione 49: Design flow del Quartus II

Come si può vedere dall'illustrazione 49, con Quartus II si può sintetizzare il progetto che si vuole realizzare, fare un'analisi nel tempo per quanto riguarda la propagazione dei segnali nel circuito digitale e la loro sincronizzazione (simulazione TEMPORALE), analizzare l'evoluzione delle uscite simulando dal PC una manipolazione degli ingressi (simulazione FUNZIONALE), e infine programmare l'FPGA. Inoltre, oltre ai linguaggi di programmazione tipici per questi dispositivi, come il **Verilog** e il **VHDL**, il Quartus II mette a disposizione una interfaccia grafica (=GUI= Graphical User Interface) dalla quale si può intervenire per ogni fase di realizzazione del progetto.

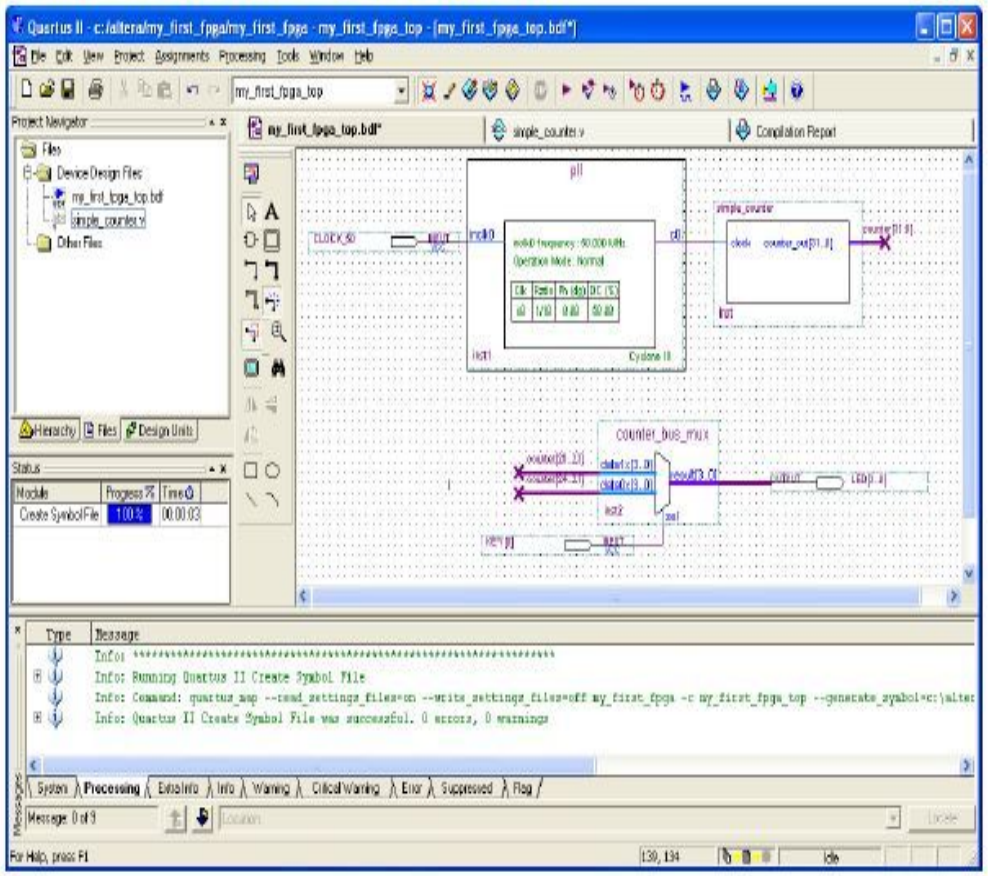


Illustrazione 50: Interfaccia grafica del Quartus II

3.3.1 VHDL

VHDL è l'acronimo di "**V**HSIC **H**ardware **D**escription **L**anguage". A sua volta ***VHSIC*** vuol dire "**V**ery **H**igh **S**peed **I**ntegrated **C**ircuits" ed era un progetto del Dipartimento di difesa americano. Il VHDL nasce nel 1987 quando diventa lo standard IEEE 1076 e nel 1993 ne esce una versione aggiornata, in grado di sfruttare a pieno le capacità di questo linguaggio, in particolare per:

- La sintesi automatica dei circuiti a partire dalla descrizione
- La verifica delle costruzioni temporanee (prototipi)
- La verifica formale dell'equivalenza dei circuiti

Il VHDL si presenta per molti versi simile a un vero e proprio linguaggio di programmazione, in particolare ne usa i tipici costrutti quali if...then...else. Tuttavia, essendo un linguaggio che descrive il funzionamento e la struttura di componenti hardware, ha alcune caratteristiche distintive rispetto ai linguaggi software. La principale è la **concorrenzialità**; con questo termine si indica il fatto che diverse parti di un codice scritto in VHDL devono essere immaginate funzionanti contemporaneamente, perché varie parti di un circuito elettronico funzionano contemporaneamente; in generale invece in un linguaggio software come il C, le funzioni descritte dal codice sono eseguite sequenzialmente, riga dopo riga. Altre caratteristiche che lo differenziano dai linguaggi software usuali come il C sono:

- **I/O**: tutta l'interfaccia ingresso/uscita deve essere descritta nei suoi dettagli di funzionamento, come il protocollo di acquisizione dei dati o la loro rappresentazione.
- **Ampiezza delle variabili**: questa ha un impatto notevole sulle prestazioni del dispositivo utilizzato. Deve essere quindi possibile, pur utilizzando tipi di dati astratti o predefiniti, specificare l'ampiezza in bit delle variabili utilizzate.
- **Temporizzazione**: ogni temporizzazione in VHDL deve essere dichiarata esplicitamente e deve essere rappresentata da uno o più segnali di clock.
- **Sincronizzazione**: la sincronizzazione fra le diverse componenti deve essere definita esplicitamente da un apposito protocollo. Deve quindi essere possibile definire dei vincoli temporali per modellare questo tipo di comportamenti.

Esso permette di modellare facilmente l'interazione tra i vari blocchi funzionali che compongono un sistema. Queste interazioni sono essenzialmente lo scambio di segnali di controllo e di dati tra i vari oggetti che costituiscono il sistema. In un sistema hardware infatti ogni oggetto da modellizzare, che sia esso una semplice porta logica o un complesso microprocessore, reagisce istantaneamente agli stimoli ai suoi ingressi producendo dei cambiamenti sulle sue uscite. Ogni blocco funzionale, a sua volta, è descritto nella relazione ingressi-uscite, usando i classici costrutti del linguaggi di programmazione (if, for, while).

Le principali fasi di progettazione sono due:

- nella prima l'oggetto viene descritto come `ENTITY`, ovvero come il componente viene visto dall'esterno: questa sezione comprende generalmente le porte di comunicazione, i tempi di ritardo, la larghezza dei bus e i parametri di configurazione;
- nella seconda si progetta la `ARCHITECTURE`, ovvero l'architettura interna in cui si descrive come il dispositivo funziona; è questa la parte più importante perché ne costituisce la descrizione funzionale vera e propria.

Anche se ognuno può scrivere questa seconda parte come vuole, di solito vengono usati due stili: **COMPORTAMENTALE** o **STRUTTURALE**. Con il primo si descrive la relazione funzionale ingressi-uscita tramite una funzione o un algoritmo; se invece si usa lo stile strutturale si rappresenta la struttura interna del dispositivo formata da componenti di più basso livello ed i loro collegamenti (RTL: Register Transfer Level), ovvero si decide già con quali oggetti realizzare la funzionalità e li si connette tra di loro. Nella progettazione di un nuovo dispositivo, di solito, si parte da una descrizione comportamentale di alto livello, per passare poi ad una descrizione RTL ovvero costituito dai componenti fondamentali digitali come i registri, l'ALU, i bus e le macchine a stati. Il vantaggio della programmazione strutturale è quello di poter ricavare le informazioni relative a un circuito a diversi livelli di astrazione, supportando descrizioni gerarchiche del sistema digitale, in modo da permettere di realizzare sia un flusso di progetto top-down, sia bottom-up. Sempre in un'ottica di modularità, permette il riutilizzo di unità di progetto precedentemente create.

L'ultimo passo di traduzione del modello RTL in una netlist è eseguito in maniera automatica da un programma software che si chiama "sintetizzatore". Esso è già compreso nel software del Quartus II. Il sintetizzatore produce una netlist, ovvero un file di istanze (=istruzioni su come deve essere strutturato il sistema digitale) di celle della tecnologia su cui viene "mappato" il circuito digitale. La tecnologia è di solito quella delle FPGA o dei circuiti integrati CMOS.

Durante l'esecuzione di tutto il flusso di progettazione si effettuano delle simulazioni per verificare che sia mantenuta la congruenza tra i vari modelli comportamentale, RTL e netlist. Per effettuare

queste simulazioni si usa un test-bench, scritto anch'esso in VHDL, che ha la funzione di generare gli stimoli sugli ingressi del circuito e di verificare la correttezza delle uscite.

3.3.2 Verilog

Il Verilog fu inventato nel 1985 da Phil Moorby presso la Automated Integrated Design Systems (più tardi denominata Gateway Design Automation) come un linguaggio di programmazione dell'hardware. Nel 1990 l'azienda fu comprata da Cadence Design Systems, la quale rese di pubblico dominio il linguaggio, fino ad allora proprietario. Da qui derivò la possibilità di renderlo uno standard, cosa che divenne con il nome di IEEE 1364, con revisioni nel 1995, 2001 e 2005. Cadence attualmente detiene la proprietà completa dei diritti dei simulatori logici della vecchia Gateway.
Verilog è un linguaggio di descrizione dell'hardware (HDL) usato per descrivere sistemi elettronici.

Il linguaggio (a volte chiamato Verilog HDL) supporta la progettazione, la verifica, e l'implementazione di circuiti digitali e più raramente di circuiti analogici o circuiti misti analogico-digitali a vari livelli di astrazione. Gli inventori del Verilog volevano un linguaggio con una sintassi simile al C cosi' che fosse familiare agli utilizzatori e facilmente accettato. Allo stato attuale può considerarsi l'unico linguaggio, assieme al VHDL, utilizzato nel mondo della progettazione e simulazione digitale con una quota di mercato pari a circa il 50% rispetto al più moderno ma assai più rigido VHDL.

Il linguaggio distingue tra caratteri maiuscoli e caratteri minuscoli, ha un preprocessore come il C, e le maggiori parole chiave di controllo del flusso del programma, come "if" e "while", sono uguali al C. La formattazione delle procedure di stampa, gli operatori del linguaggio e la loro precedenza sono simili al C. Il linguaggio differisce dal C in alcuni punti fondamentali. Verilog usa “Begin/End” invece delle parentesi graffe per definire un blocco di codice. Le costanti in Verilog richiedono di essere specificate in termini di larghezza in numero di bit insieme al tipo di base utilizzata per la definizione. Verilog 95 e 2001 non ha strutture, puntatori, sotto-procedure ricorsive, mentre queste sono presenti nel SystemVerilog che ora include queste capacita'. Infine il concetto di temporizzazione, così importante per l' HDL, non esiste nel C.

Il linguaggio differisce dai linguaggi di programmazione convenzionali nell'esecuzione delle istruzioni dato che, essendo un linguaggio che descrive processi paralleli e concorrenti, l'esecuzione non è strettamente lineare e per tutte le altre caratteristiche viste per il VHDL.

Un progetto Verilog consiste di una gerarchia di **moduli**. Ciascuno è definito da un insieme di ingressi e uscite e porte bidirezionali. Internamente contiene una lista di connessioni e registri. Definizione di processi paralleli e sequenziali ne definiscono il comportamento definendo la relazione tra le porte i registri e i fili. Le istruzioni sequenziali sono poste all'interno di un blocco begin/end in ordine sequenziale all'interno del blocco.

Tutti le istruzioni concorrenti e tutti i blocchi begin/end sono eseguiti in parallelo. Un modulo contiene una o più istanze di un altro modulo per definire sotto-comportamenti. Anche per il Verilog, proprio come il VHDL, a lista di connessioni (netlist) è un bitstream utilizzata per un dispositivo logico programmabile (come l'FPGA).

Come riferimento per conoscere nel dettaglio le istruzioni, le modalità di programmazione e i costrutti dei linguaggi Verilog e VHDL sono stati utilizzati i manuali [3] e [4].

Nel seguito si descriveranno dettagliatamente tutte le funzioni messe a disposizione dal Quartus II seguendo la realizzazione del progetto software per calcolare, attraverso l'FPGA, le medie delle tensioni concatenate del motore in alternata, ad ogni periodo di PWM.
Molto utili per imparare ad utilizzare tutte le parti che compongono Quartus II sono stati i **Tutorial** messi a disposizione dall'Altera e dalla Terasic. Nel primo c'è la descrizione passo-passo per la realizzazione di un software generico fino alla programmazione del dispositivo e la sua esecuzione. In particolare si gestisce la velocità di lampeggio dei LED presenti nella Starter Board attraverso la pressione di un pulsante. Nel secondo invece un file in codice Verilog permette di gestire i pin del connettore HMSC interfacciati con i convertitori A/D e D/A.

3.4 Sviluppo del progetto software

Prima di cominciare con la redazione del progetto bisogna assicurarsi che tutto il software **Quartus II** fornito con la Starter Board sia installato nella maniera corretta e in tutte le sue componenti. La licenza per il suo utilizzo deve essere richiesta tramite Internet attraverso il sito dell'Altera. Inoltre deve essere installato il driver per la connessione USB, che permette di interfacciare il PC con la scheda digitale.

Come prima cosa bisogna creare un nuovo **PROGETTO**. Con questo termine si intende l'insieme di tutti i file che contengono delle informazioni per la programmazione della FPGA. I file primari nel progetto Quartus II sono i ***Quartus II Settings File*** (**.qsf**) e ***Quartus II Project File*** (**.qpf**). Nella pagina principale del programma si seleziona **File > New Project Wizard** e si sceglie il nome che si vuole dare al progetto e la directory dove verranno salvati tutti i file associati ad esso. Una volta fatto questo si creano automaticamente i 2 file primari nella cartella di lavoro voluta.

Successivamente bisogna assegnare al progetto la specifica FPGA che si sta utilizzando. Dalla schermata principale si seleziona **Assignments > Device**, si setta la famiglia di dispositivi corretta (Cyclone III) e il numero che identifica l'integrato utilizzato: EP3C25F324C8.

Ora si può cominciare realmente a realizzare l'algoritmo, entrando nella fase, definita nell'illustrazione 49, di **Design Entry.** In questa fase si crea un ***Block Design File*** (**. bdf**), cioè uno schematico (top-level design) nella quale si possono connettere i blocchi logici creati dall'utente (attraverso i linguaggi Verilog o VHDL) o quelli già presenti nella libreria del Quartus II. Questi blocchi logici sono dette **FUNZIONI**. Si presentano come delle “black box” che effettuano una determinata elaborazione logico/matematica e con le quali si può interagire connettendo ingressi e uscite ad altre “black box” o direttamente ai pin I/O dell'FPGA. Nello schematico hanno tipicamente forma rettangolare, nel lato sinistro ci sono le connessioni per gli ingressi e a destra per le uscite. Nel caso di blocchi logici essenziali, essi possono assumere le forme tipiche dell'elettronica digitale, come illustrato nell'esempio di schematico di figura 51.

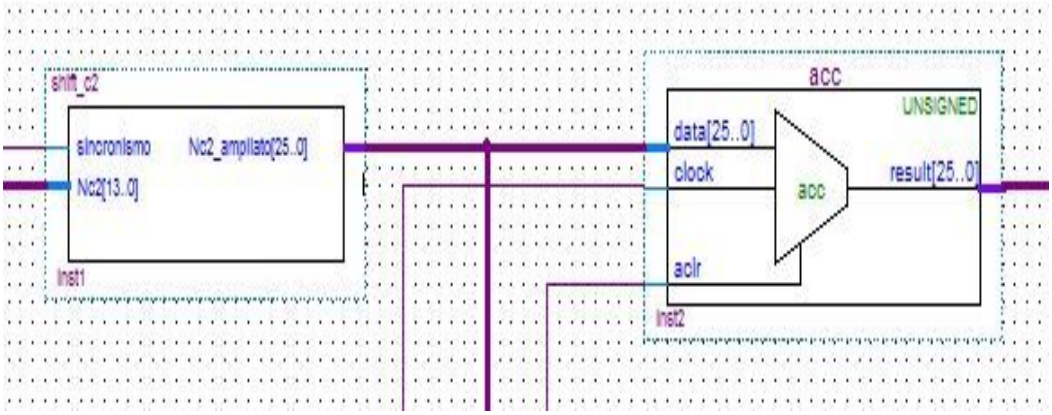


Illustrazione 51: Esempio di connessione di 2 funzioni attraverso i loro I/O

Per creare il Block Design File si sceglie dalla schermata principale **File > New > Block Diagram/Schematic File,** che successivamente andrà salvato con lo stesso nome del progetto dato all'inizio. Da qui in poi apparirà nello schermo una griglia dove possono essere poste le funzioni desiderate e/o i pin di I/O.

Prima di cominciare la programmazione, bisogna decidere nel dettaglio come si potrà realizzare (a livello teorico) l'algoritmo per il calcolo delle medie delle tensioni concatenate sul periodo PWM in maniera digitale.

3.4.1 Descrizione dell'algoritmo per l'elaborazione digitale delle tensioni concatenate del motore

Innanzitutto si analizzano gli ingressi che dovranno essere elaborati in questo progetto. I convertitori A/D della Daughter Board THDB_ADA elaborano le tensioni concatenate del motore (opportunamente ridotte) in un segnale digitale a 14 bit nella codifica ***offset-binary***. Un esempio a 4 bit di questo sistema di rappresentazione binario è mostrato nell'illustrazione 52.

Binary code	Decimal code
1111	7
1110	6
1101	5
1100	4
1011	3
1010	2
1001	1
1000	0
0111	−1
0110	−2
0101	−3
0100	−4
0011	−5
0010	−6
0001	−7
0000	−8

Illustrazione 52: Codifica offset-binary a 4 bit

Questi convertitori sono **BIPOLARI**, in quanto codificano segnali analogici sia positivi che negativi, simmetrici rispetto allo zero (tensione nulla). La curva caratteristica di questo tipo di convertitore, i cui ingressi variano da $-V_{FS}/2$ a $+V_{FS}/2$ (FS = fondo scala), si può vedere nell'illustrazione 53A. Si tenga presente che in tal caso il convertitore presenterà una codifica offset binary in cui le tensioni di fondo scala assumono i seguenti valori digitali:

$-V_{FS}/2 = (000\ldots000)$ … $V_{FS}/2 = (111\ldots111)$

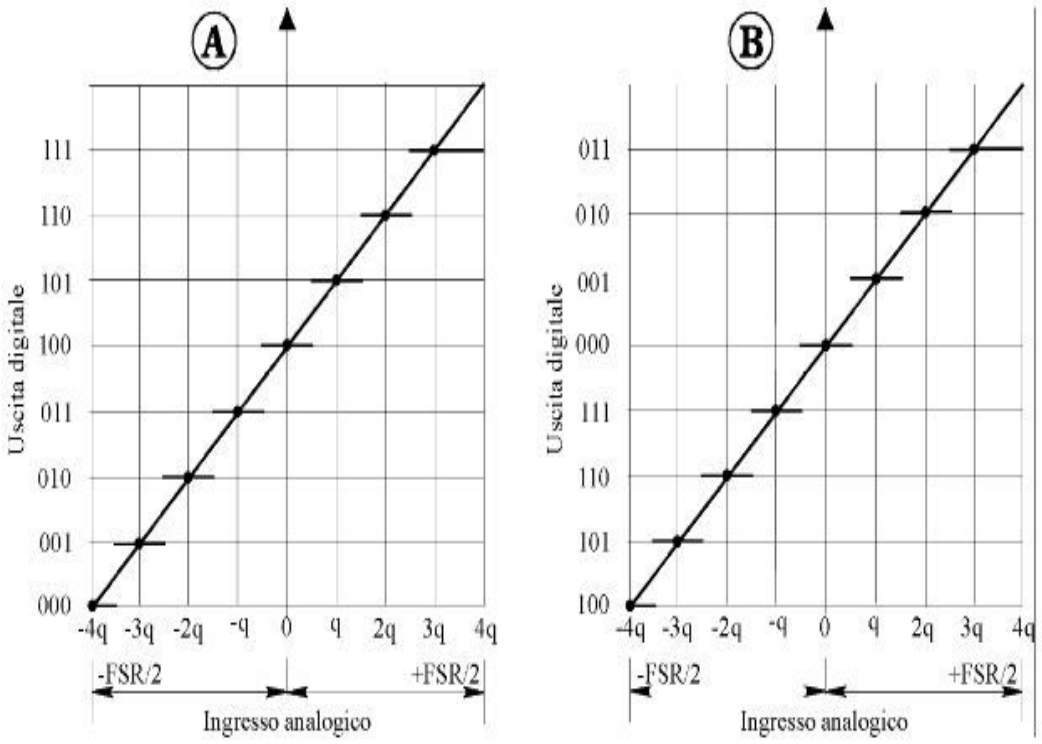


Illustrazione 53: Caratteristica di un convertitore A/D bipolare A) in offset-binary B) in complemento a 2

Con riferimento sempre all'illustrazione 53A (esempio di un convertitore A/D a 3 bit in offset-binary), al primo livello di quantizzazione, corrispondente al valore analogico $-V_{FS}/2$, si assegna il codice (000); mentre all'ultimo livello, corrispondente al valore analogico $V_{FS}/2$, si assegna il codice (111). La tensione nulla è rappresentata da un numero binario con "1" nell'MSB (Most Significant Bit) e tutti gli altri bit a zero. Questo numero è detto **offset**.

L'offset-binary non è molto utilizzato nelle elaborazioni digitali, in quanto le operazioni elementari di somma e sottrazione fra valori numerici risultano molto più efficienti e semplici se si adotta la codifica in ***complemento a due***. Con quest'ultimo sistema di rappresentazione binario infatti, i circuiti (o equivalentemente gli algoritmi) di addizione e sottrazione non devono esaminare il segno di un numero per determinare quale delle due operazioni sia necessaria, permettendo tecnologie più semplici e maggiore precisione; si utilizza un solo circuito, il sommatore, sia per l'addizione che per la sottrazione.

Inoltre un numero binario di *n* cifre può rappresentare con questo metodo i numeri compresi fra -2^{n-1} e $+2^{n-1}-1$, proprio come la codifica offset-binary.

Questo sistema consente di rappresentare lo zero con tutti i bit al valore logico basso e di operare efficientemente addizione e sottrazione, pur mantenendo il primo bit a indicare il segno.

Per esprimere i codici in complemento a due, è sufficiente complementare il bit più significativo della codifica offset-binary. In tal modo si ottiene la corrispondenza:

$-V_{FS}/2 = (100\ldots000)$ … $V_{FS}/2 = (011\ldots111)$

Equivalentemente, si può sottrarre o sommare l'offset per ottenere la conversione di codifica. Nell'illustrazione 53B si può vedere l'esempio di conversione A/D a 3 bit in complemento a due e confrontarla direttamente con la conversione in offset-binary della figura 53A. Si può notare come esse differiscano solo nel bit più significativo.

Ora, per calcolare la media mobile di un segnale numerico in una determinata finestra temporale in maniera digitale, è sufficiente sommare gli N campioni che arrivano dalla conversione A/D durante il periodo di interesse e dividere tutto per N, come riporta la seguente formula:

$$m_g = \frac{x_1 + x_2 + x_3 + \ldots + x_N}{N}$$

In particolare, per replicare il funzionamento della scheda analogica di misura, **si dovranno sommare algebricamente tutti i campioni delle tensioni concatenate che arrivano dai convertitori A/D in un arco di tempo pari al semiperiodo di PWM, e calcolarne la media in base alla formula precedente.**

I convertitori della scheda THDB_ADA campionano i segnali digitali con frequenze nell'ordine delle decine di MHz, quindi nel semiperiodo di PWM (=21μs) vengono sommati un numero tale di valori digitali che rappresentano le tensioni concatenate del motore per cui il calcolo è estremamente preciso e confrontabile con il suo duale analogico.

Per fare un esempio, se si campiona a 65 MHz, sul semiperiodo di PWM vengono elaborati 65MHz/(12kHz*2)=2708 numeri digitali. Quindi in questo modo si ottiene una misura estremamente precisa della media delle tensioni concatenate, e si perviene ad un risultato paragonabile a quello a cui si è arrivati per via analogica, pur con diversi rapporti di amplificazione.

Gli ingressi analogici sono tensioni che possono essere sia positive che negative, e vengono codificate in offset-binary. Per poter sommare algebricamente numeri digitali positivi e negativi in maniera semplice ed efficace bisogna passare alla codifica "***complemento a due***", come spiegato sopra.

Dopo la somma algebrica degli N campioni, bisogna dividere il valore ottenuto per N. Nel sistema binario, la divisione si ottiene semplicemente attraverso lo shift logico a destra di M posizioni. Lo shift a destra di M posizioni binarie corrisponde alla divisione per $2^{M.}$

Dopo aver effettuato la somma e lo shift logico, quindi alla fine del semiperiodo di PWM, il numero ottenuto deve essere ritradotto in offset-binary per essere decodificato nella maniera corretta dai convertitori D/A, in quanto essi sono configurati per elaborare in ingresso questo tipo di valori numerici.

Riassumendo, l'algoritmo per realizzare questa elaborazione digitale deve seguire i seguenti passi:

1) Conversione A/D del dato analogico in un valore numerico in offset-binary ad una frequenza di campionamento fissata.
2) Trasformazione di ogni dato da “offset-binary” a “complemento a 2”.
3) Somma algebrica di tutti i dati campionati nel semiperiodo di PWM.
4) Shift logico a destra di M posizioni del risultato ottenuto, al fine di ottenere un valore digitale coerente con la lunghezza di parola del convertitore D/A (14 bit).
5) Trasformazione della media mobile numerica così ottenuta da “complemento a 2” a “offset-binary.

L'illustrazione 54 mostra uno schema a blocchi che riassume le elaborazioni digitali descritte. Anche in questo caso c'è una sorta di "**doppio canale di integrazione**", ognuno del quale lavora su metà periodo:

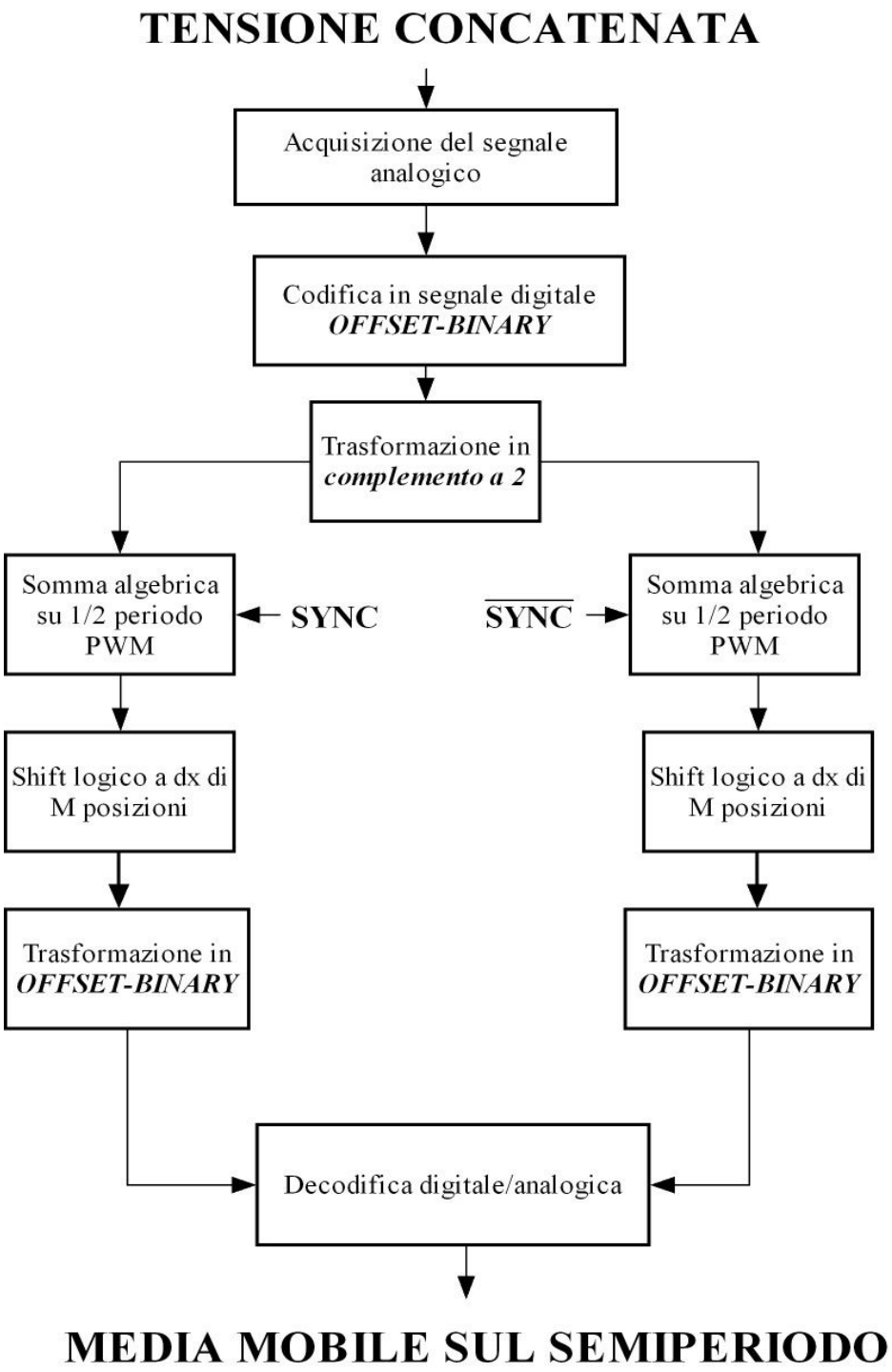


Illustrazione 54: Schema a blocchi dell'algoritmo digitale

Sarà ancora il segnale di sincronismo (con duty-cycle = 50%) a "coordinare" i due canali perché elaborino nella maniera corretta ognuno per metà periodo i valori digitali corrispondenti alle tensioni concatenate campionate.

3.4.2 Scelta della lunghezza di parola ed errori dovuti ad arrotondamento/troncamento

Nella descrizione dell'algoritmo precedente non si è tenuto conto del fatto che sommando N dati digitali a 14 bit si andrà incontro inevitabilmente ad un overflow del risultato finale della somma algebrica se si memorizza il risultato in un registro con lo stesso numero di bit.
Quindi la somma binaria tra numeri codificati in "complemento a due" deve avvenire in registri con un numero di bit maggiore rispetto ai 14 del dato iniziale.
Per calcolare l'esatto numero di bit del registro dove vengono sommati i campioni delle tensioni concatenate nel semiperiodo, si segue il seguente ragionamento: se il convertitore A/D campiona a 65MHz, sul semiperiodo di PWM vengono elaborati 65MHz/(12kHz*2)=2708 numeri digitali. Considerando il caso peggiore, in cui ad ogni campionamento viene elaborato il valore digitale maggiore, cioè:
11111111111111 = 2^{14-1}-1 = 8191
il registro che memorizza il risultato della somma algebrica sul semiperiodo deve prevedere un numero di bit tale da poter rappresentare il numero:
8191*2708=22181228
Per rappresentare 22181228 sono necessari almeno **26 bit**: infatti
2^{26-1}-1 = 33554431. Con 25 bit (2^{25-1}-1 = 16777215), il numero 22181228 non sarebbe rappresentabile.
Ovviamente si potrebbe anche considerare un registro maggiore, ma ciò andrebbe ad aumentare l'inevitabile errore di arrotondamento, poiché del registro a 26 bit si dovranno considerare solamente i 14 più significativi, mentre gli altri non vengono considerati.
Quindi ogni campione in uscita dal convertitore A/D deve prima essere codificato in "complemento a due" e poi trasformato in numero di egual valore, ma a 26 bit anziché 14. I numeri digitali in complemento a 2 si prestano molto bene a questo tipo di trasformazione, in quanto è sufficiente considerare l'MSB del registro a 14 bit, e "prolungarlo" nei restanti bit più significativi del registro a 26 bit, mentre le prime 14 cifre, partendo da destra, rimangono le stesse del valore originale.

Per chiarire meglio la situazione si consideri il seguente esempio:**101 = -3** nella "codifica complemento a 2". Se si volesse tradurre questo valore in un numero digitale a 6 bit, prolungo l'MSB (=1) nei 3 bit più significativi, mentre gli altri rimangono gli stessi: **111101 = -3**. Ovviamente questo ragionamento vale sia per numeri negativi che positivi.

Una volta conclusa l'operazione di somma binaria di tutti i campioni ottenuti in un semiperiodo, sarà sufficiente considerare del registro a 26 bit solamente le 14 cifre più significative, in quanto la conversione D/A finale ha la stessa risoluzione della campionatura iniziale. Questo si ottiene attraverso uno shift a destra di 12 posizioni.

Tenendo conto di tutte queste considerazioni, l'algoritmo finale viene modificato come illustrato in figura 55.

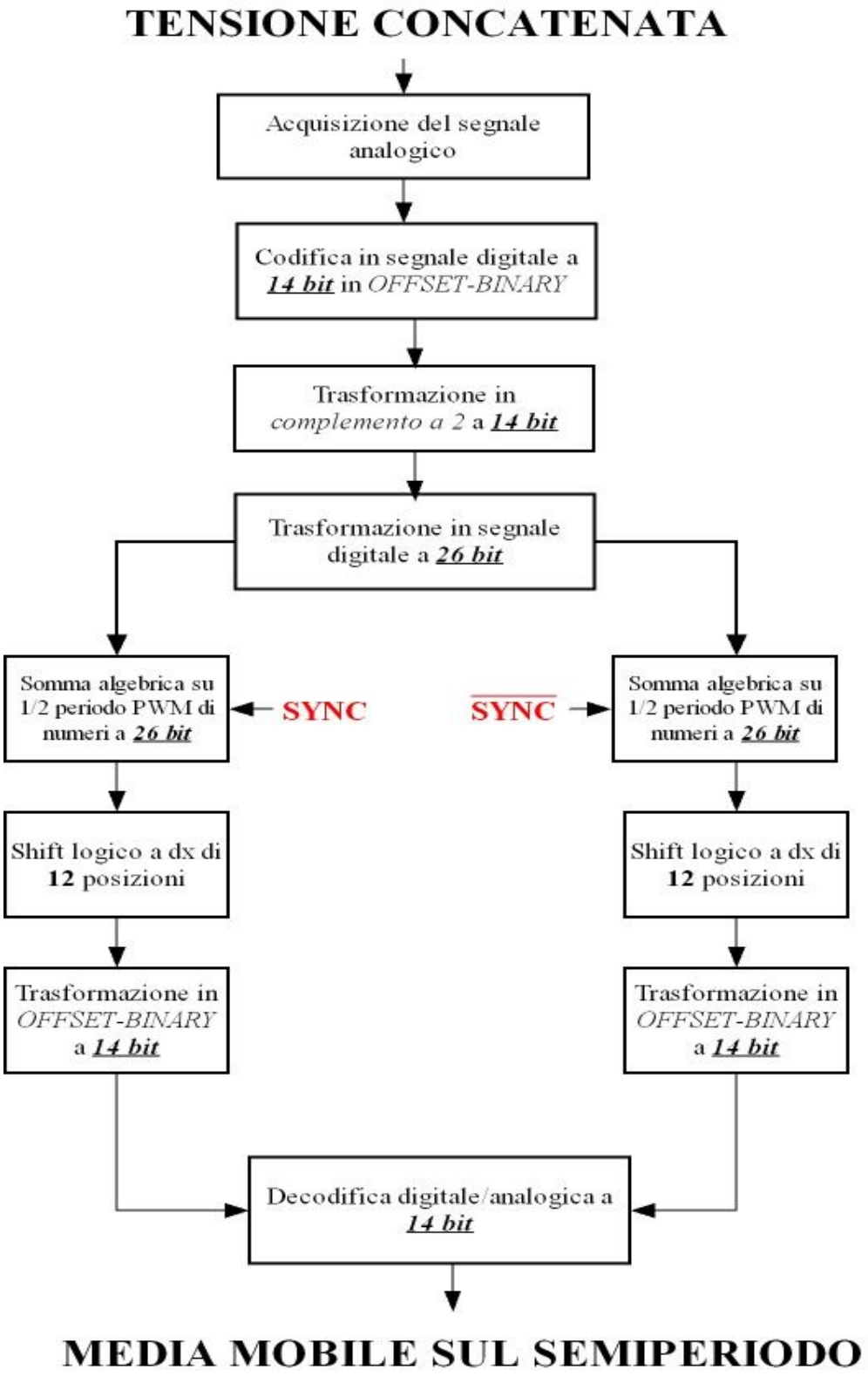
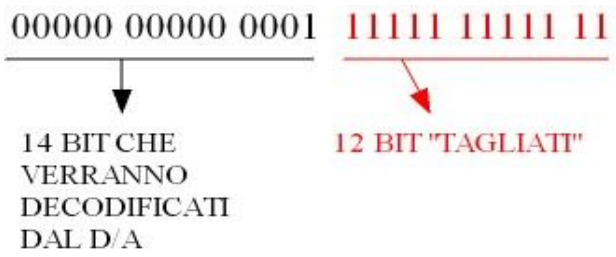


Illustrazione 55: Schema a blocchi MODIFICATO dell'algoritmo digitale

Questa sequenza logica di comandi, che rappresenta l'algoritmo descritto precedentemente, è ripetuta per ognuno dei segnali analogici acquisiti, cioè le **due** tensioni concatenate del motore.

La variazione di tensione percepita dal convertitore A/D per cui c'è la variazione di un bit nella sequenza digitale è $(2V_{pp})/(2^{14}\text{livelli})$ = **122,07μV**. Se ne deduce la grande sensibilità di questo tipo di conversione alle variazioni dell'ingresso analogico.

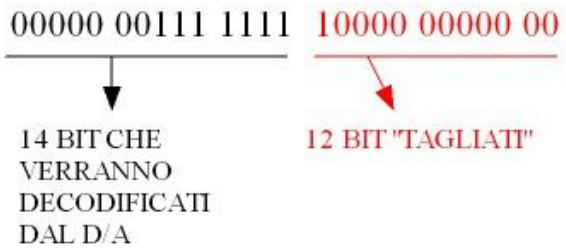
Lo shift a destra di 12 posizioni dopo la somma binaria sul semiperiodo porta inevitabilmente ad un **errore di troncamento**. Si può valutare questa approssimazione nel caso peggiore:



L'errore di troncamento viene calcolato così:

[valore dei 12 bit meno significativi del registro a 26 bit]/[valore rappresentato su 26 bit].

In questo caso è $[(2^{13}-1)-(2^{12}-1)]/(2^{13}-1)$ = **50%**. Ovviamente questo è in assoluto l'errore più grande che può capitare. Questo però si riduce in maniera **esponenziale** al crescere del numero rappresentato nei 14 bit più significativi (e al decrescere dei 12 bit meno significativi), fino ad annullarsi. Analizzando un caso medio:



L'errore di troncamento in questo caso è [(522240)-(522240-2048)]/(522240) = **0,3%**.

Quindi da questi calcoli si può dedurre che mediamente un errore dovuto all'arrotondamento del numero derivato dalla somma binaria ci sia sempre, ma possa essere considerato trascurabile visto la sua natura esponenziale.

Una volta scelto com'è progettato l'algoritmo definitivo, si può riprendere con la definizione del software di programmazione dell'FPGA attraverso il Quartus II.

Dal Block Editor si può cominciare ad inserire i moduli che andranno a svolgere una determinata funzione. I moduli possono essere sintetizzati attraverso i linguaggi Verilog o VHDL, o si possono utilizzare le cosiddette ***Megafunction*** messe a disposizione dal Quartus II. Queste sono dei moduli già ottimizzati dal punto di vista della velocità di esecuzione e occupazione d'area nel dispositivo programmabile che svolgono determinate funzioni, anche molto complesse, già prestabilite.

Innanzitutto si aggiunge un blocco per fornire il clock che fissa la frequenza di campionamento degli A/D, attraverso i PLL della FPGA. In questo modo il clock per la scheda THDB_ADA può essere settato via software e scelto tra i multipli dell'oscillatore interno della FPGA (50MHz).

Dalla schermata principale del Block Editor si sceglie **Edit > Insert Symbol** e nella grafica che appare si clicca su "**Megawizard Plug-in Manager**":

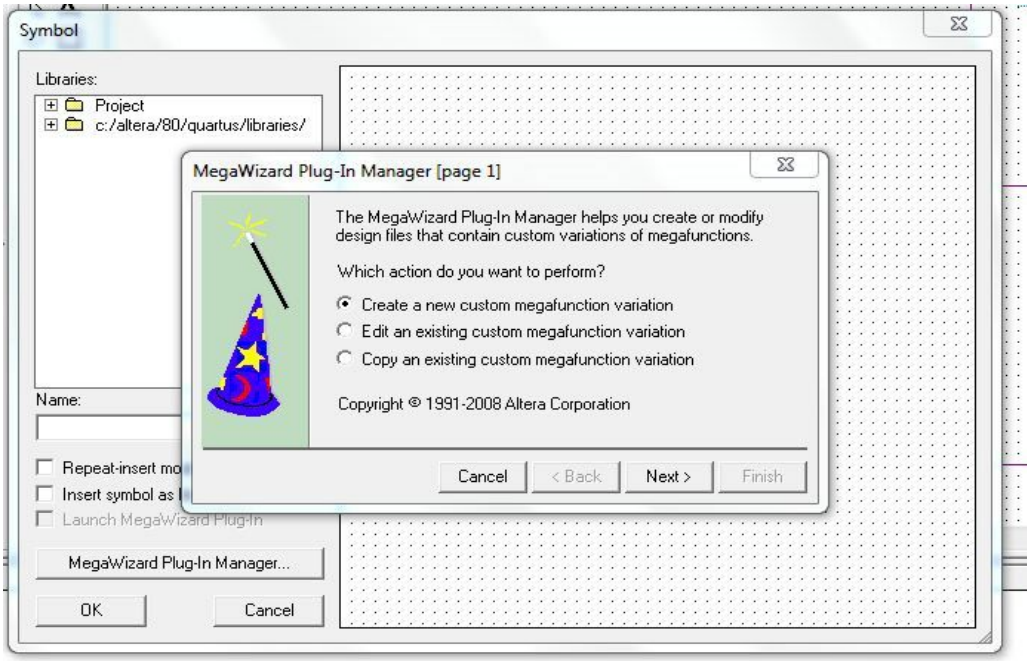
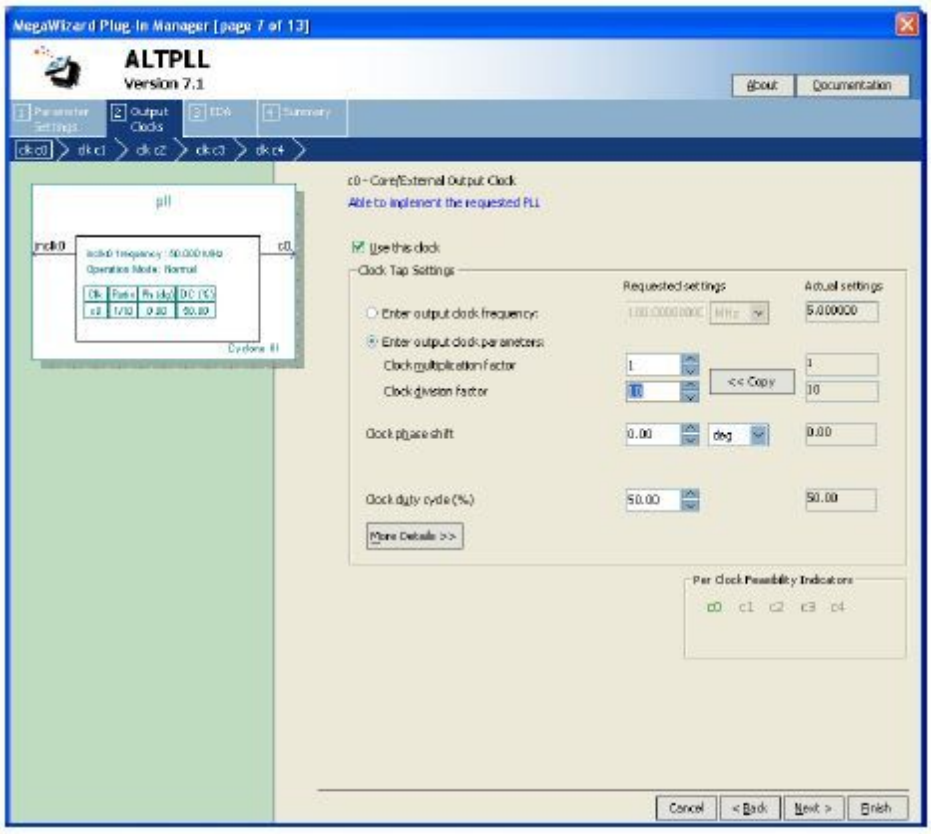


Illustrazione 56: Schermata del Megawizard Plug-in Manager

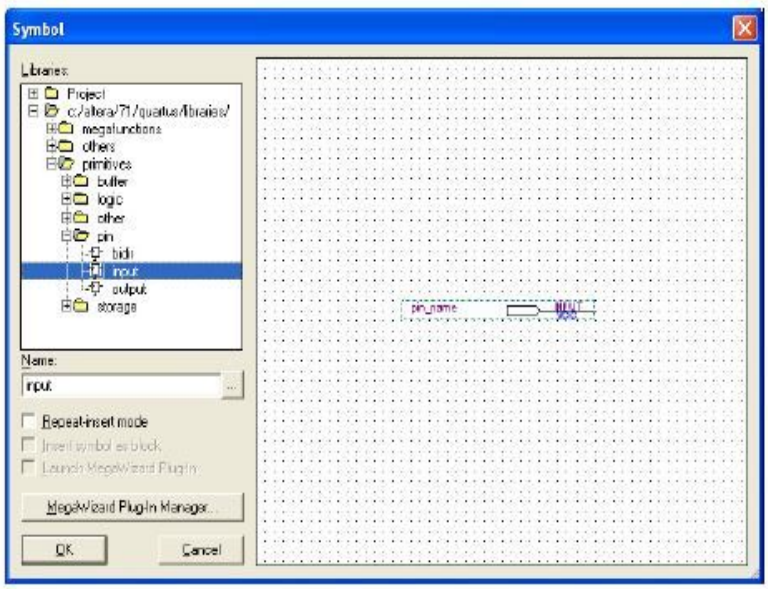
Premendo "Next" appare una lista di Megafunction tra le quali si può scegliere quella che realizza la funzione voluta. In questo caso il modulo cercato è **I/O > ALTPLL**. Da qui in poi ci saranno tutta una serie di settaggi per decidere nome del file, dispositivi sui quali si sta lavorando e linguaggio software nel quale viene sintetizzato il blocco (si pone Verilog HDL di default su tutti da qui in poi). Il suo ingresso corrisponde all'oscillatore dell'FPGA (50 MHz). Ora basta decidere che frequenza generare in uscita in "**Output clocks**":



Qui può essere fissato un fattore moltiplicativo e un fattore di divisione per il clock d'ingresso al fine di ottenere la frequenza cercata. Visto che, per adesso, non serve prestare attenzione a problematiche legate a dissipazione di potenza, **il convertitore A/D può campionare alla frequenza massima permessa, cioè 65 MHz**. Ponendo il fattore moltiplicativo=13 e quello di divisione uguale a 10, si ottiene in uscita (50MHz*13)/10=65MHz.

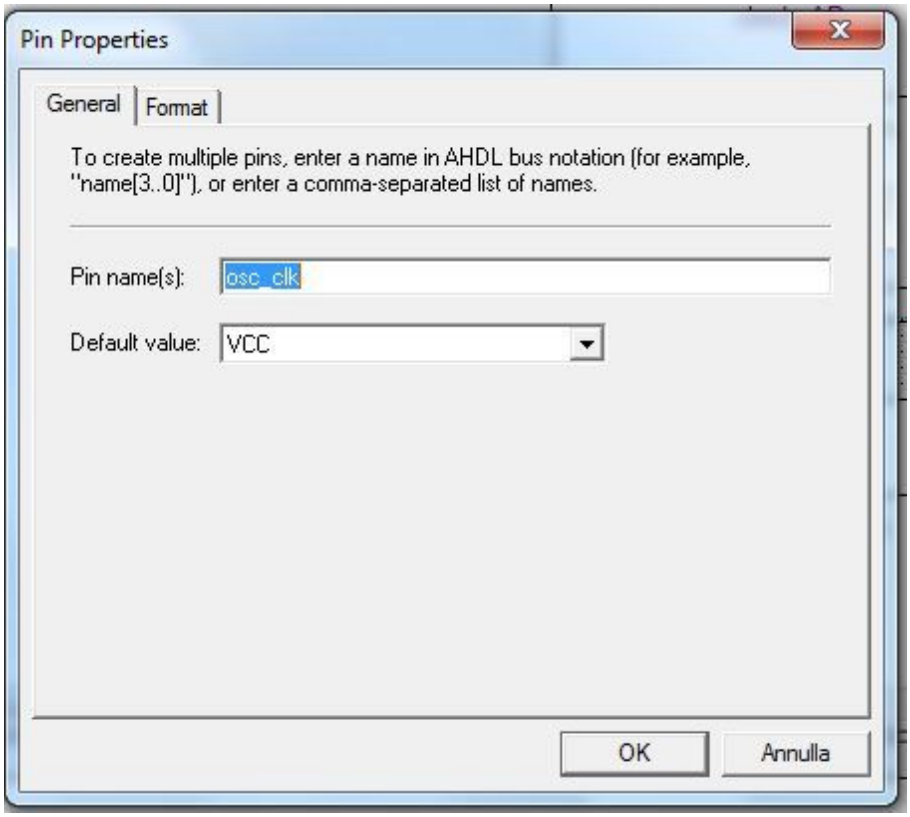
La sintesi della Megafunction (le altre opzioni settabili non servono in questo caso) può essere conclusa cliccando su "**Finish**". Ora è stato creato un simbolo che può essere posto nel Block Editor ed interagire con gli altri moduli attraverso i suoi ingressi e le sue uscite.

Per identificare che l'ingresso del blocco appena generato è un pin dell'FPGA (in questo caso il pin del circuito di clock a 50MHz) si deve inserire un nuovo simbolo, il **Pin Input**. Per porlo nel Block Editor, **Edit > Insert Symbol** e sotto "**Libraries**" si seleziona **quartus/libraries > primitives > pin > input** come nella seguente illustrazione:



Allo stesso modo si possono aggiungere al progetto dei **Pin Output** per assegnare alle uscite della FPGA i risultati delle funzioni svolte. Ogni pin di I/O deve essere identificato attraverso un nome che verrà, in una fase successiva, associato al codice specifico del pin che si può trovare nel "Reference Manual" della scheda.

Cliccando 2 volte sul simbolo che si è appena posto nel Block Editor si assegna un nome: in questo caso "osc_clk":



Con dei **Pin Output** si assegnano alle uscite dell'FPGA connesse con i convertitori A/D della scheda THDB_ADA la frequenza desiderata di campionamento del segnale analogico.

Le connessioni fra i vari blocchi e i pin di I/O si effettuano tramite dei fili di connessione, "**Orthogonal Bus Tool**" se si tratta di una connessione fra numeri digitali a più bit, altrimenti "**Orthogonal Node Tool"** se riguarda la connessione di un bit. Questi si trovano nella schermata a sinistra del Block Editor.

Il progetto, con questi blocchi, pin I/O e le loro connessioni si presenta come illustrato in figura 57.

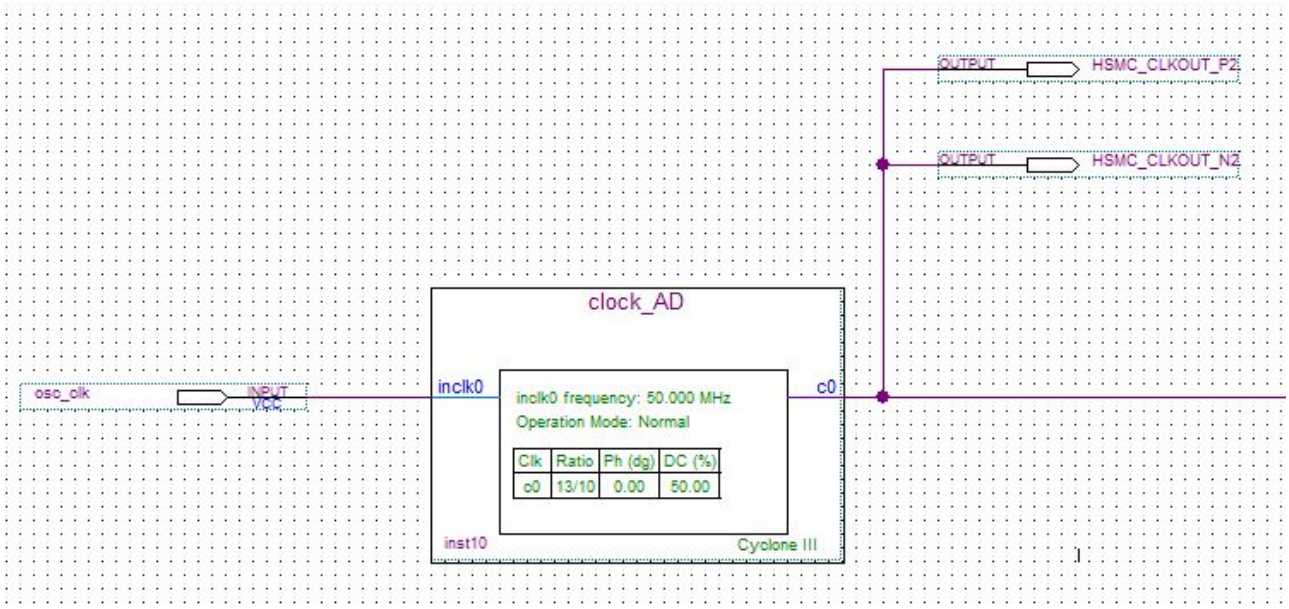


Illustrazione 57: Prima parte dello schematico Quartus II

Sono stati assegnati 2 **Pin Output**, uno per ogni convertitore A/D per fissare le rispettive frequenze di campionamento.

Il prossimo modulo vine progettato nel linguaggio Verilog: la sua funzione è quella di acquisire il segnale digitale in uscita dagli A/D alla frequenza di 65 MHz in offset-binary e codificarlo in "complemento a due". Per aggiungere un blocco in Verilog, si seleziona

File > New > Verilog HDL File. A questo punto si crea nella cartella di lavoro un file con estensione **.v**, che può venire salvato con il nome desiderato (in questo caso **sottrazione_offset.v**). Immediatamente dopo la creazione del file si apre un editor di testo dove si può cominciare a scrivere il programma, di seguito riportato:

```
module sottrazione_offset (sincronismo, in, out);  //dichiarazione modulo
 //tra parentesi sono dichiarati gli ingressi e le uscite del modulo
reg [13:0] out;    //ogni I/O o registro interno utilizzato è dichiarato in questo modo specificando
                   //da quanti bit è formato
output [13:0] out;
input [13:0] in;

input sincronismo;

always @(sincronismo & !sincronismo)   //temporizzazione (sempre necessaria
                                       //nei blocchi Verilog o VHDL)
begin

out [13] = !in [13];       //per la codifica in complemento a 2 basta invertire il MSB
out [12] = in [12];
out [11] = in [11];
out [10] = in [10];
out [9] =  in [9];
out [8] =  in [8];
out [7] =  in [7];
out [6] =  in [6];
out [5] =  in [5];
out [4] =  in [4];
out [3] =  in [3];
out [2] =  in [2];
out [1] =  in [1];
out [0] =  in [0];

end
endmodule
```

Da questo file è possibile creare un blocco da aggiungere nello schematico dove sono posti e connessi tutti gli altri moduli che compongono il progetto. Scegliendo **File > Create/Update > Create Symbol Files for Current Update** si converte "**simple_counter.v"** in un "**Symbol File**"

(**.sym**). In questo modo si effettua anche una compilazione del modulo in Verilog, con la possibilità di vedere se sono stati fatti degli errori nella sintassi del codice elaborato.

Per aggiungerlo nello schematico si clicca 2 volte nella griglia dove apparirà la lista illustrata in figura 58 da cui è possibile selezionare il blocco desiderato:

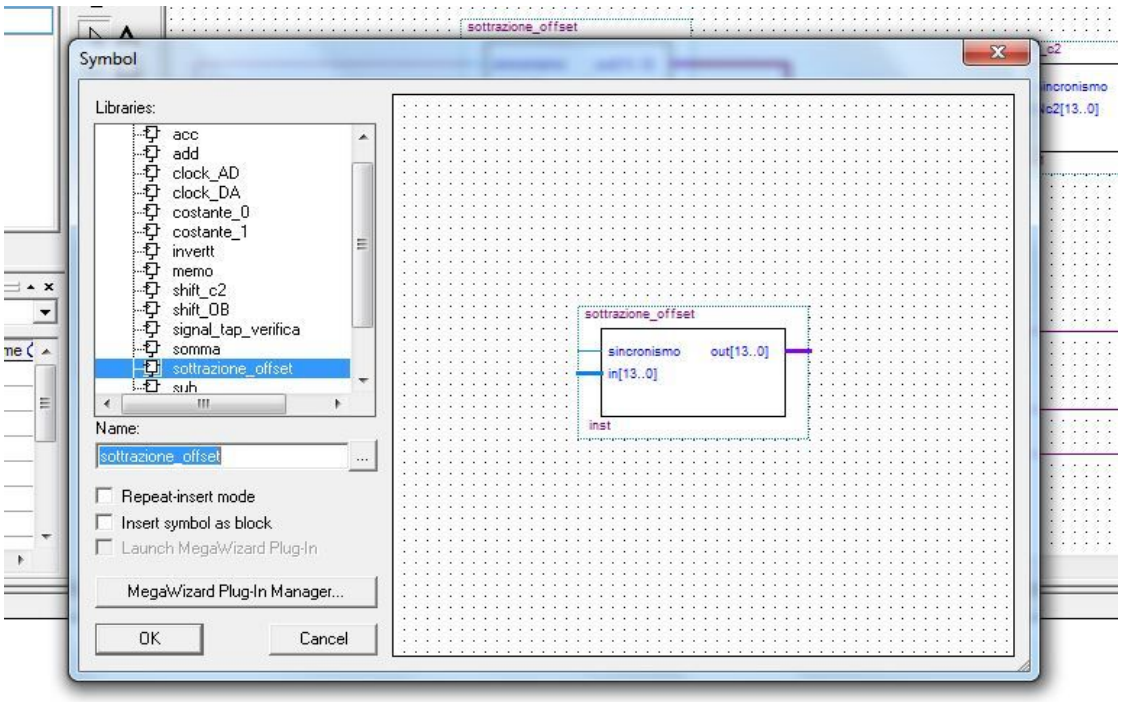


Illustrazione 58: Lista dei blocchi costruiti

L'ingresso di questo modulo è un **Pin Input** che rappresenta il dato convertito a 14 bit che arriva dal convertitore A/D. L'uscita invece è connessa direttamente ad un altro blocco, generato anche questo in Verilog, che svolge la funzione di **estendere il numero digitale codificato in "complemento a due" da 14 a 26 bit**. Di seguito è riportato il codice per generare questo blocco:

```
module shift_c2 (sincronismo, Nc2, Nc2_ampliato);

reg [25:0] Nc2_ampliato;

output [25:0] Nc2_ampliato;

input [13:0] Nc2;

input sincronismo;
```

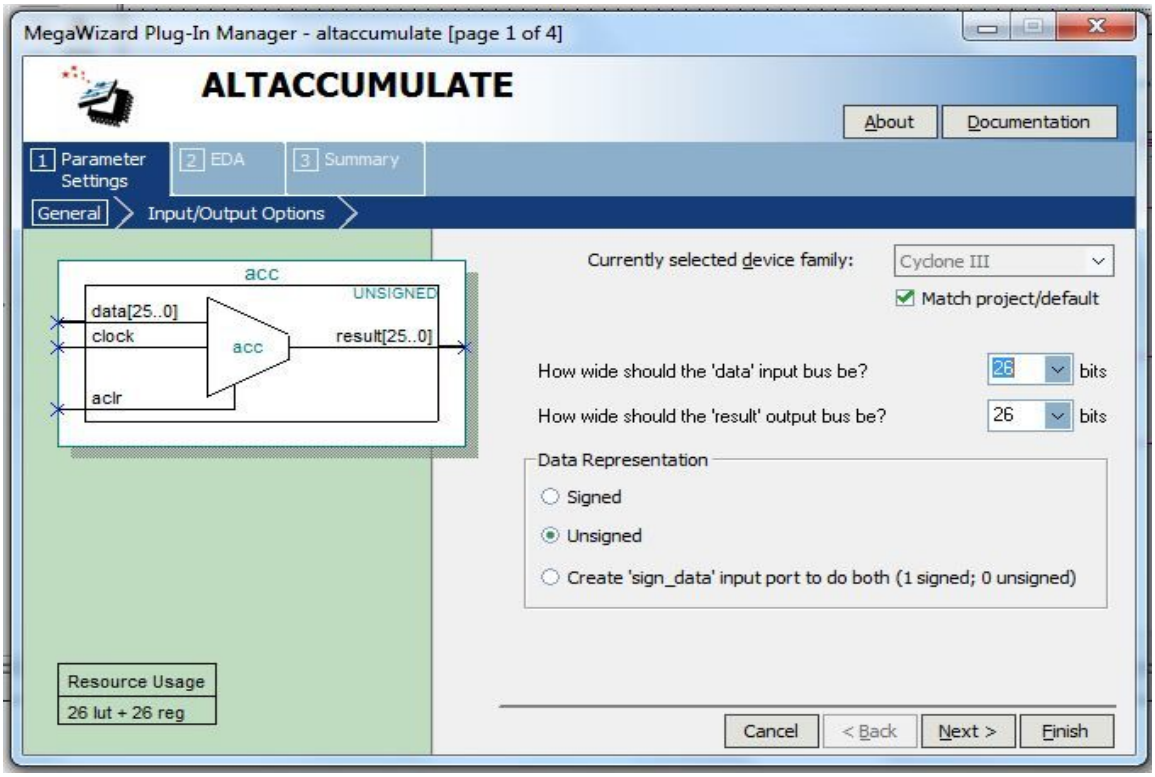
```
always @(sincronismo & !sincronismo)

begin
Nc2_ampliato [25] = Nc2 [13];     //MSB esteso sui bit più significativi
Nc2_ampliato [24] = Nc2 [13];    //MSB esteso sui bit più significativi
Nc2_ampliato [23] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [22] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [21] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [20] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [19] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [18] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [17] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [16] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [15] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [14] = Nc2 [13];      //MSB esteso sui bit più significativi
Nc2_ampliato [13] = Nc2 [13];      //da qui in poi le cifre rimangono le stesse
Nc2_ampliato [12] = Nc2 [12];
Nc2_ampliato [11] = Nc2 [11];
Nc2_ampliato [10] = Nc2 [10];
Nc2_ampliato [9] = Nc2 [9];
Nc2_ampliato [8] = Nc2 [8];
Nc2_ampliato [7] = Nc2 [7];
Nc2_ampliato [6] = Nc2 [6];
Nc2_ampliato [5] = Nc2 [5];
Nc2_ampliato [4] = Nc2 [4];
Nc2_ampliato [3] = Nc2 [3];
Nc2_ampliato [2] = Nc2 [2];
Nc2_ampliato [1] = Nc2 [1];
Nc2_ampliato [0] = Nc2 [0];

end
endmodule
```

Come si può notare dal codice precedente, il bit più significativo del valore a 14 bit viene esteso ai restanti bit più significativi del numero a 26 bit: in questo modo i due registri codificano (in "complemento a due") lo stesso valore intero, ma con un numero di bit maggiore.

Da qui inizia il **doppio canale di integrazione,** come quello implementato nella scheda analogica: il segnale digitale a 26 bit, ottenuto come descritto sopra, dovrà essere "accumulato"con due sommatori differenti,ognuno che lavora su un semiperiodo di PWM. Sarà acquisito, come segnale digitale, il segnale di sincronismo (=SYNC) generato dal monostabile con duty-cycle al 50% e in fase con la PWM che comanda gli IGBT dell'inverter (vedi paragrafo 2.5).
Per realizzare la funzione di somma binaria di numeri digitali codificati in complemento a due c'è una apposita funziona messa a disposizione dal Quartus II tra le sue "Megafunctions" predefinite, denominata **ALTACCUMULATE**. Per conoscere dettagliatamente le possibilità offerte dalle Megafunctions del Quartus II può essere utilizzato l'"help" del programma stesso, o i numerosi Tutorial messi a disposizione gratuitamente dal sito dell'Altera con esempi di utilizzo dei blocchi.
Attraverso il "**Megawizard Plug-in Manager**", tra la lista di moduli predefiniti, si seleziona ALTACCUMULATE nella cartella "**Arithmetic**". Dopo aver dato un nome al file e scelto la cartella dove verrà salvato, appare la seguente schermata, dalla quale è possibile impostare il funzionamento del sommatore in base alle specifiche della funzione che deve svolgere:



La Megafunction ALTACCUMULATE implementa un accumulatore che somma (o sottrae) il dato che gli arriva in ingresso ("***data[25..0]***") ai valori sommati precedentemente, fino a che non viene resettato da un apposito comando digitale ("***aclr***").

Come per ogni modulo, anche questo deve avere una temporizzazione, un clock che decide la cadenza con la quale si svolge l'operazione di somma binaria. Si utilizza in questo caso il segnale a 65 MHz generato dal blocco PLL: in questo modo ogni volta che il convertitore A/D campiona un segnale questo viene sommato agli altri dati ottenuti nel semiperiodo.

Viste le già citate proprietà dei numeri digitali codificati in "complemento a due", si utilizza un solo circuito, e quindi una sola funzione, sia per l'addizione che per la sottrazione e i valori digitali sommati in ingresso possono essere considerati "senza segno".

Per resettare il registro dove vengono accumulati i dati sommati c'è un apposito ingresso, "***aclr***", che annulla il risultato della somma binaria quando è a livello logico "alto". Il segnale ideale per svolgere questo compito è proprio il SYNC: sul semiperiodo dove è "basso", l'accumulatore svolge la sua operazione binaria, mentre quando è "alto" annulla il risultato della somma fino all'inizio del periodo successivo. Sono previste molte altre funzioni per questo blocco, come un riporto in ingresso o in uscita, ma non vengono considerate per questo specifico progetto.

Nell'altro canale di integrazione viene posto un ulteriore modulo ALTACCUMULATE, con lo stesso clock, lo stesso ingresso di dati da sommare e le stesse impostazioni, ma con il segnale "***aclr***" invertito rispetto all'altro canale (SYNC). In questo modo ogni accumulatore somma i segnali digitali che arrivano dal convertitore A/D in un semiperiodo di PWM, e in seguito si resetta in maniera alternata rispetto all'altro canale.

Per generare il negato di SYNC è sufficiente aggiungere il modulo predefinito **LPM_INV** dalla lista dei blocchi logici: esso si presenta in figura come solitamente rappresentato nell'elettronica digitale. Lo schematico, così com'è stato impostato fino a questo punto, si presenta come illustrato in figura 59.

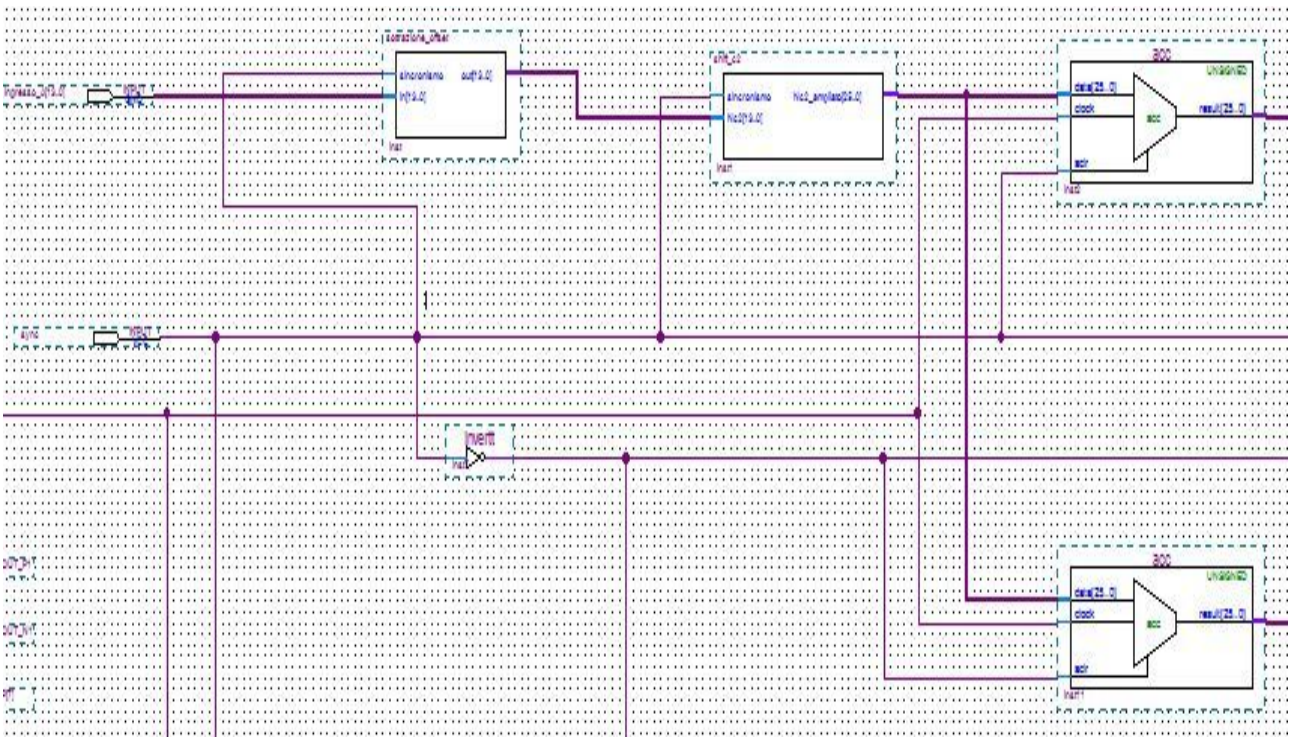


Illustrazione 59: Schematico

L'uscita degli accumulatori è un numero digitale a 26 bit codificato in complemento a 2. Per essere decodificato dai convertitori D/A della scheda THDB_ADA deve essere “trasformato” in offset-binary e riportato su 14 bit. Come è già stato visto, per codificare un numero da “complemento a 2” a “offset-binary” è sufficiente invertire l'MSB o, equivalentemente, sommare l'offset che in questo caso è rappresentato dal numero:

10000 00000 00000 00000 000000 = 33554432

Si è scelto di sommare questo valore attraverso il modulo predefinito **LPM_ADD_SUB**, che svolge la semplice addizione tra i 2 dati che arrivano in ingresso: in questo caso il risultato della somma binaria a 26 bit e l’offset (valore costante).

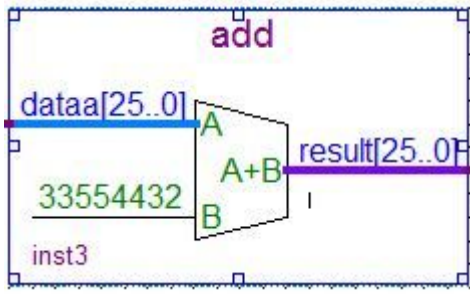


Illustrazione 60: Megafunction LPM_ADD_SUB

Per portare il dato su 14 bit, è sufficiente "shiftare" a destra di 12 posizioni il valore a 26 bit. Equivalentemente si possono memorizzare i 14 bit più significativi del registro esteso e tralasciare i restanti. Questa funzione è realizzata nel blocco "***shift_OB***", realizzato in Verilog con il seguente codice sorgente:

```
module shift_OB (sincronismo, in, out);

reg [13:0] out;
output [13:0] out;
input [25:0] in;
input sincronismo;

always @(sincronismo & !sincronismo)

begin

 out [13] = in [25];   //MSB del numero a 26 bit è memorizzato nell'MSB del numero a 14 bit
 out [12] = in [24];   //e così di seguito per tutti gli altri più significativi
 out [11] = in [23];
 out [10] = in [22];
 out [9] = in [21];
 out [8] = in [20];
 out [7] = in [19];
 out [6] = in [18];
 out [5] = in [17];
 out [4] = in [16];
 out [3] = in [15];
 out [2] = in [14];
 out [1] = in [13];
 out [0] = in [12];

end
endmodule
```

Il successivo blocco deve svolgere la funzione di memorizzare l'uscita del sommatore convertita in 14 bit "offset binary" durante tutto il semiperiodo utile di integrazione e, quando l'integratore si è resettato, mantenere in uscita l'ultimo dato registrato, che corrisponde proprio al risultato dell'integrazione delle tensioni concatenate sul semiperiodo, fino a quando l'accumulatore non ricomincia a svolgere la somma binaria.

In pratica il modulo deve registrare il risultato dell'integrazione dei campioni nel semiperiodo e renderlo disponibile in uscita durante il reset dell'accumulatore, perché possa essere mandato al convertitore D/A che lo decodificherà. Questa funzione può essere svolta da un LATCH**,** realizzato mediante la Megafunction **LPM_LATCH.**

Il Latch è provvisto di un ingresso denominato ***"gate"*** che abilita il modulo in questo modo:

"gate"=1 => uscita = ingresso

"gate"=0 => uscita = ultimo ingresso memorizzato quando ***"gate"***=1

Questo blocco è proprio il più idoneo per la funzione che si vuole realizzare, ponendo ***"gate"***= SYNC nel canale di integrazione dove l'accumulatore ha come segnale di "***clear***" del registro di somma SYNC. In questo modo, il Latch fornisce il risultato real-time della somma dei campioni delle tensioni concatenate, fino a quando non finisce il semiperiodo. In quello successivo l'accumulatore non sta integrando, mentre il Latch mantiene costante l'ultimo valore memorizzato della somma, cioè l'integrazione della concatenata su un semiperiodo.

In maniera duale anche nell'altro canale c'è un Latch in uscita al sommatore con ***"gate"***= SYNC e il "***clear***" dell'accumulatore uguale a SYNC. Nell'immagine 61 è riportato un parte dello schematico con gli ultimi blocchi descritti precedentemente:

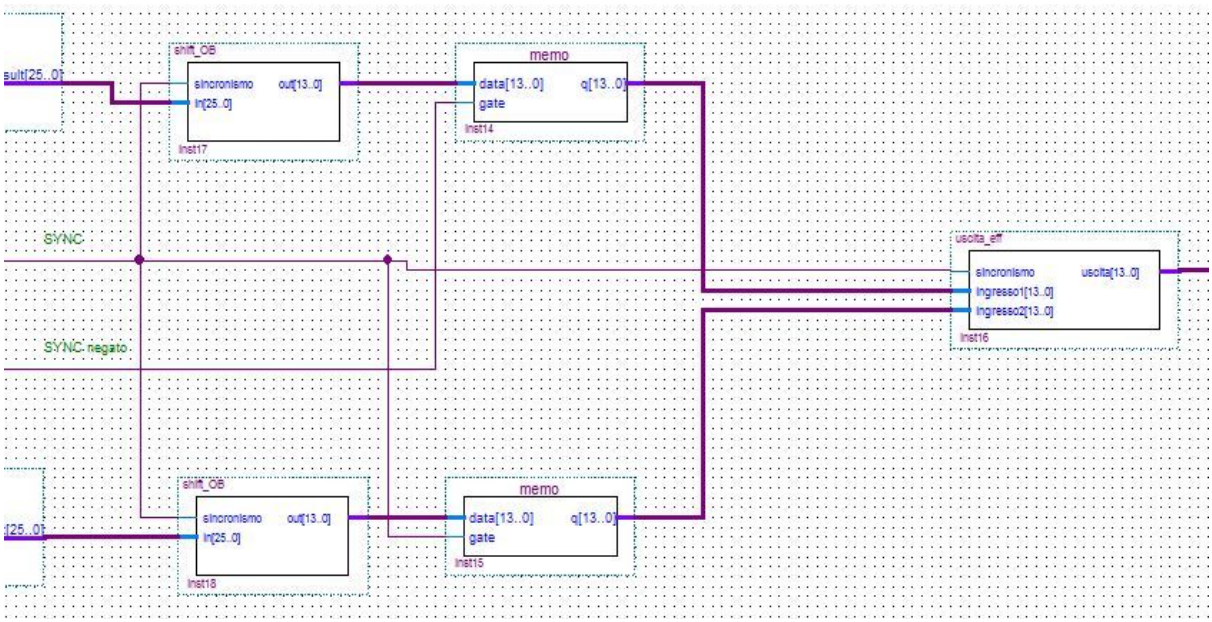


Illustrazione 61: Parte finale dello schematico realizzato

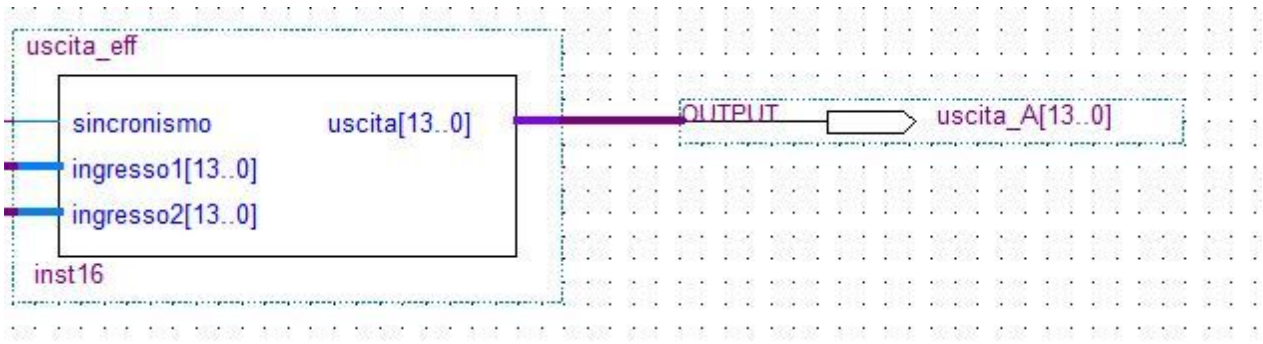
Agendo in questo modo si ripercorre lo stesso funzionamento dello stadio di integrazione della scheda analogica. Ad ogni cambio di fronte del segnale di sincronismo è disponibile l'integrazione sul semiperiodo precedente del dato campionato in ingresso. Così come è stata finora progettata la soluzione digitale, ci sono però 2 uscite che lavorano alternativamente, una su un semiperiodo di PWM e l'altra sul successivo. Si deve quindi elaborare un blocco che "inglobi" in un'unica uscita i risultati dell'integrazione che poi verrà mandata in uscita ai convertitori D/A della daughter board THDB_ADA. Il modulo per compiere questa funzione è stato realizzato in linguaggio Verilog, e successivamente creato il simbolo per aggiungerlo allo schematico. E' stato denominato "***uscita_eff.v***" e sfrutta anch'esso il segnale di sincronismo: quando SYNC è alto, l'uscita è uguale al risultato dell'integrazione effettuata sul canale che in quel semiperiodo è resettato (quello con "***clear***" dell'accumulatore = SYNC e il ***"gate"*** del Latch = SYNC), altrimenti è uguale a quella dell'altro canale (quello con "***clear***" dell'accumulatore = SYNC e il ***"gate"*** del Latch = SYNC). In questo modo viene memorizzato e mandato al convertitore D/A solamente il risultato finale dell'integrazione sul semiperiodo. Di seguito è riportato il codice elaborato per questo modulo:

```
module uscita_eff (sincronismo, ingresso1, ingresso2, uscita);

reg [13:0] uscita;
output [13:0] uscita;
input [13:0] ingresso1;
input [13:0] ingresso2;
input sincronismo;

always @(sincronismo or !sincronismo)
begin
if (sincronismo)   //registro uscita canale che si sta resettando (SYNC=1)
                uscita = ingresso1;
else                 //registro uscita canale che si sta resettando  (SYNC=1)
                uscita = ingresso2;
end
```

endmodule



Con un **Pin Output** in uscita dal modulo si manda il risultato al convertitore D/A, del quale si può decidere la frequenza con cui decodifica i segnali digitali che gli arrivano. E' stato scelto di utilizzare la stessa frequenza di campionamento dei convertitori A/D, quindi 65 MHz (generati attraverso la Megafunction PLL), con altri 2 **Pin Output** in uscita al PLL.

Tutti i moduli precedentemente descritti e il doppio canale di integrazione sono sdoppiati nello schematico per integrare anche l'altra tensione concatenata del motore. Le uniche differenze stanno nei **Pin Input** e i **Pin Output** che saranno ovviamente diversi.

Come ultima aggiunta allo schematico per compiere il progetto si deve assegnare il valore costante "0" (valore logico basso) a 2 pin del connettore HMSC che comunica con la daughter board THDB_ADA per abilitare entrambi i canali A e B di campionatura dei segnali analogici e ad altri 2 pin il valore costante"1", per comunicare3 che essi lavorino in modo indipendente l'uno dall'altro (modalità **dual port**) (illustrazione 62).

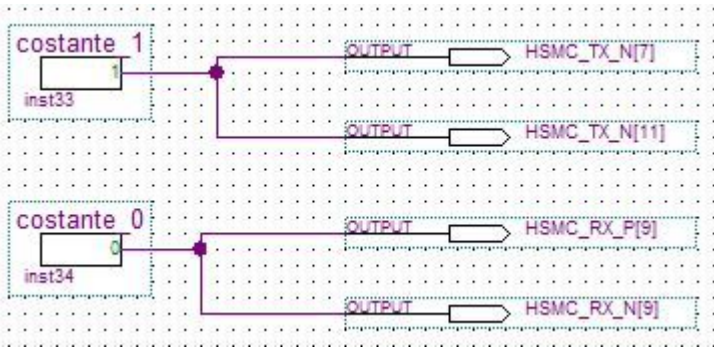


Illustrazione 62: Costanti per decidere le modalità di funzionamento degli A/D ed abilitarli

Ora che l'algoritmo, previsto per realizzare il progetto, è stato completato e realizzato attraverso il sistema di sviluppo Quartus II, bisogna assegnare i Pin di I/O della FPGA ai **Pin Input** e **Pin Output** messi nello schematico. Prima di fare questo però, è consigliabile lanciare dal Quartus II i comandi **Processing > Start > Start Analysis & Elaboration.** In questo modo viene creata internamente al progetto una lista di tutti gli I/O utilizzati e questo facilita molto l'assegnazione successiva dei pin.

Eseguendo **Assignments > Pins**, viene aperto il cosiddetto ***Pin Planner,*** uno schema della FPGA con la lista degli I/O utilizzati per assegnarli ai vari pin della FPGA:

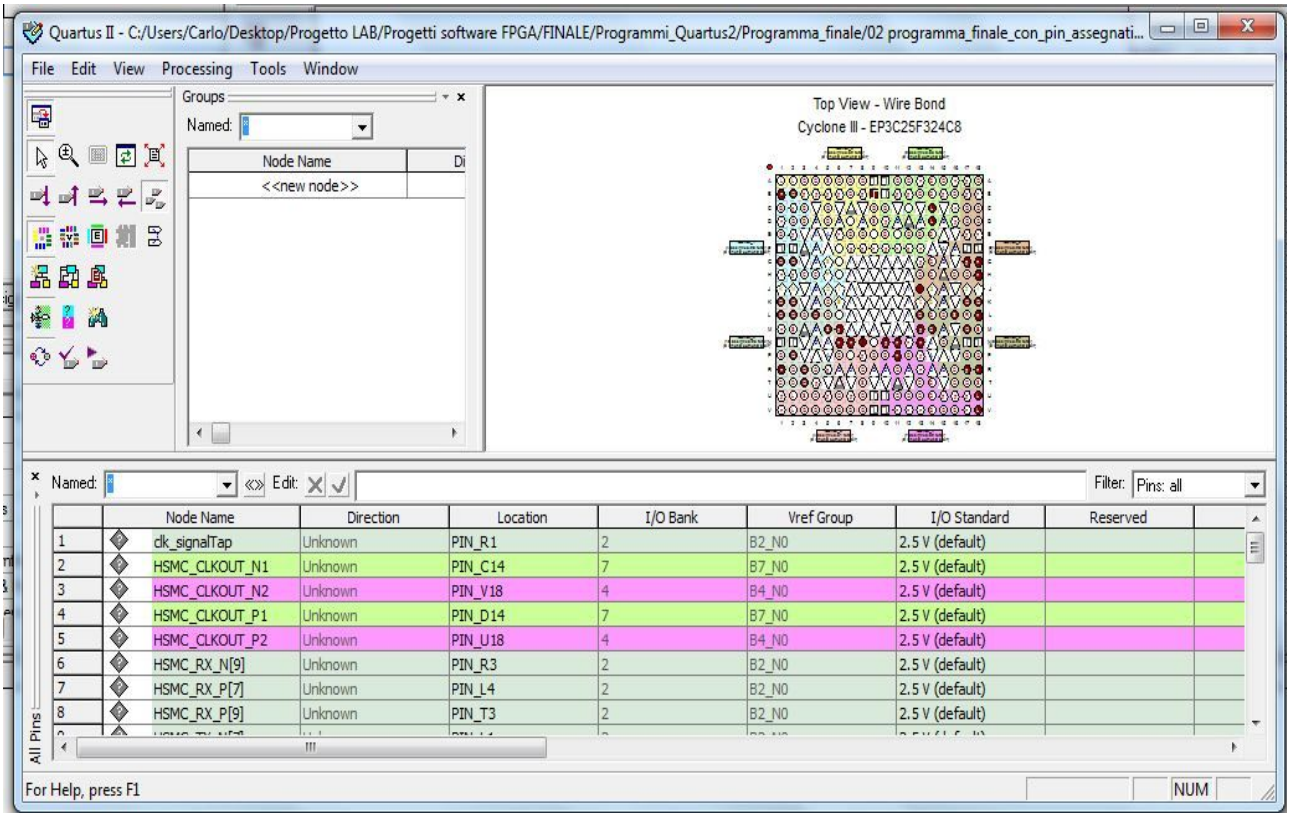


Illustrazione 63: Pin Planner

Poiché precedentemente è stato lanciato il comando **Processing > Start > Start Analysis & Elaboration**, è già presente nella tabella del Pin Planner una lista di tutti i **Pin Input** e **Pin Output** utilizzati nel progetto: ad ognuno bisogna adesso assegnare il corretto codice di Pin della FPGA.

Tutti i codici si possono trovare nel *Reference Manual* della ***Cyclone III Starter Board*** o si possono cercare dagli schemi elettrici delle 2 schede.

Di seguito è riportata una tabella con tutti gli I/O utilizzati e le rispettive porte a cui devono essere corrisposti nel ***Pin Planner***, con una breve descrizione della funzione svolta da ognuno:

Nome Pin nel progetto	Codice Pin della FPGA	Input/ Output	Descrizione Pin
osc_clk	B9	I	Clock della Cyclone III a 50MHz
HMSC_CLKOUT_P2	U18	O	Clock per l'ADC (B) dal PLL (65 MHz)
HMSC_CLKOUT_N2	V18	O	Clock per l'ADC (A) dal PLL (65 MHz)
HMSC_CLKOUT_P1	D14	O	Clock per l'DAC (B) dal PLL (65 MHz)
HMSC_CLKOUT_N1	C14	O	Clock per l'DAC (A) dal PLL (65 MHz)
HMSC_RX_P[7]	L4	O	Clock per scrittura su porta B (65MHz)
HMSC_TX_P[7]	L2	O	Clock per scrittura su porta A (65MHz)
HMSC_TX_N[7]	L1	O	Modalità Dual Port (=1)
HMSC_TX_N[11]	H18	O	Configuratore per gli ADC
HMSC_RX_P[9]	T3	O	Enable per l'ADC (A)
HMSC_RX_N[9]	R3	O	Enable per l'ADC (B)
ingresso_A [13]	G17	I	Tensione concatenata 1 dall'ADC (A)
ingresso_A [12]	G18	I	"
ingresso_A [11]	K18	I	"
ingresso_A [10]	L18	I	"
ingresso_A [9]	L16	I	"
ingresso_A [8]	M17	I	"
ingresso_A [7]	L13	I	"
ingresso_A [6]	M14	I	"
ingresso_A [5]	R17	I	"
ingresso_A [4]	R18	I	"
ingresso_A [3]	M6	I	"
ingresso_A [2]	N6	I	"
ingresso_A [1]	M13	I	"
ingresso_A [0]	N13	I	"
ingresso_B [13]	E17	I	Tensione concatenata 2 dall'ADC (B)
ingresso_B [12]	E18	I	"
ingresso_B [11]	H17	I	"
ingresso_B [10]	H18	I	"
ingresso_B [9]	L17	I	"
ingresso_B [8]	M18	I	"
ingresso_B [7]	L14	I	"

ingresso_B [6]	L15	I	"
ingresso_B [5]	P17	I	"
ingresso_B [4]	P18	I	"
ingresso_B [3]	R5	I	"
ingresso_B [2]	R4	I	"
ingresso_B [1]	T17	I	"
ingresso_B [0]	T18	I	"
uscita_A [13]	K1	O	Valore efficace della tensione concatenata 1 al DAC (A)
uscita_A [12]	K2	O	"
uscita_A [11]	G1	O	"
uscita_A [10]	G2	O	"
uscita_A [9]	B1	O	"
uscita_A [8]	B2	O	"
uscita_A [7]	P11	O	"
uscita_A [6]	K17	O	"
uscita_A [5]	N11	O	"
uscita_A [4]	N10	O	"
uscita_A [3]	J13	O	"
uscita_A [2]	N8	O	"
uscita_A [1]	N7	O	"
uscita_A [0]	M5	O	"
uscita_B [13]	L5	O	Valore efficace della tensione concatenata 2 al DAC (B)
uscita_B [12]	K5	O	"
uscita_B [11]	H1	O	"
uscita_B [10]	H2	O	"
uscita_B [9]	C1	O	"
uscita_B [8]	C2	O	"
uscita_B [7]	T16	O	"
uscita_B [6]	R16	O	"
uscita_B [5]	N15	O	"
uscita_B [4]	N16	O	"
uscita_B [3]	H16	O	"
uscita_B [2]	H15	O	"
uscita_B [1]	T2	O	"
uscita_B [0]	M3	O	"
SYNC	P2	I	Segnale di sincronismo con δ=50%

Per assegnare i pin nel ***Pin Planner***, è sufficiente cliccare nella colonna **Location** e digitare il codice corrispondente a ogni pin (compare anche una lista di tutti i pin tra cui si può scegliere appena si clicca sopra una casella della colonna **Location**).

Il software Quartus II tiene automaticamente traccia di quale banco I/O fa parte il pin scelto (ogni banco è distinto da un diverso colore della casella) e di quale gruppo "*Vref*". Quindi per questa colonna non bisogna aggiungere nient'altro a quello che si setta in automatico. Lo stesso discorso vale per la colonna "I/O standard", poiché nella ***Cyclone III Starter Board*** tutti i segnali hanno la codifica digitale "2,5V" (=default nel Pin Planner), che definisce i livelli di tensione per considerare i segnali digitali "alti" o "bassi".

Ora la progettazione del software e i settaggi necessari per definire le modalità di funzionamento della scheda possono definirsi concluse. Per programmare il dispositivo digitale, è necessario prima COMPILARE IL PROGETTO. Attraverso la compilazione vengono creati i bitstream, cioè le sequenze di codice in linguaggio macchina, scaricati poi nella FPGA per programmarla. Il file più importante creato dalla compilazione del progetto è quello con estensione **.sof** (SRAM Object File), ed è quello che viene usato per programmare il dispositivo digitale. Dopo aver salvato il progetto, compresa l'assegnazione dei pin, si sceglie dal menù **Processing** la voce **Start Compilation**. Durante il processo di compilazione il Quartus II fornisce utili informazioni sullo stato della elaborazione svolta, mentre appare una schermata come quella illustrata nella figura 64.

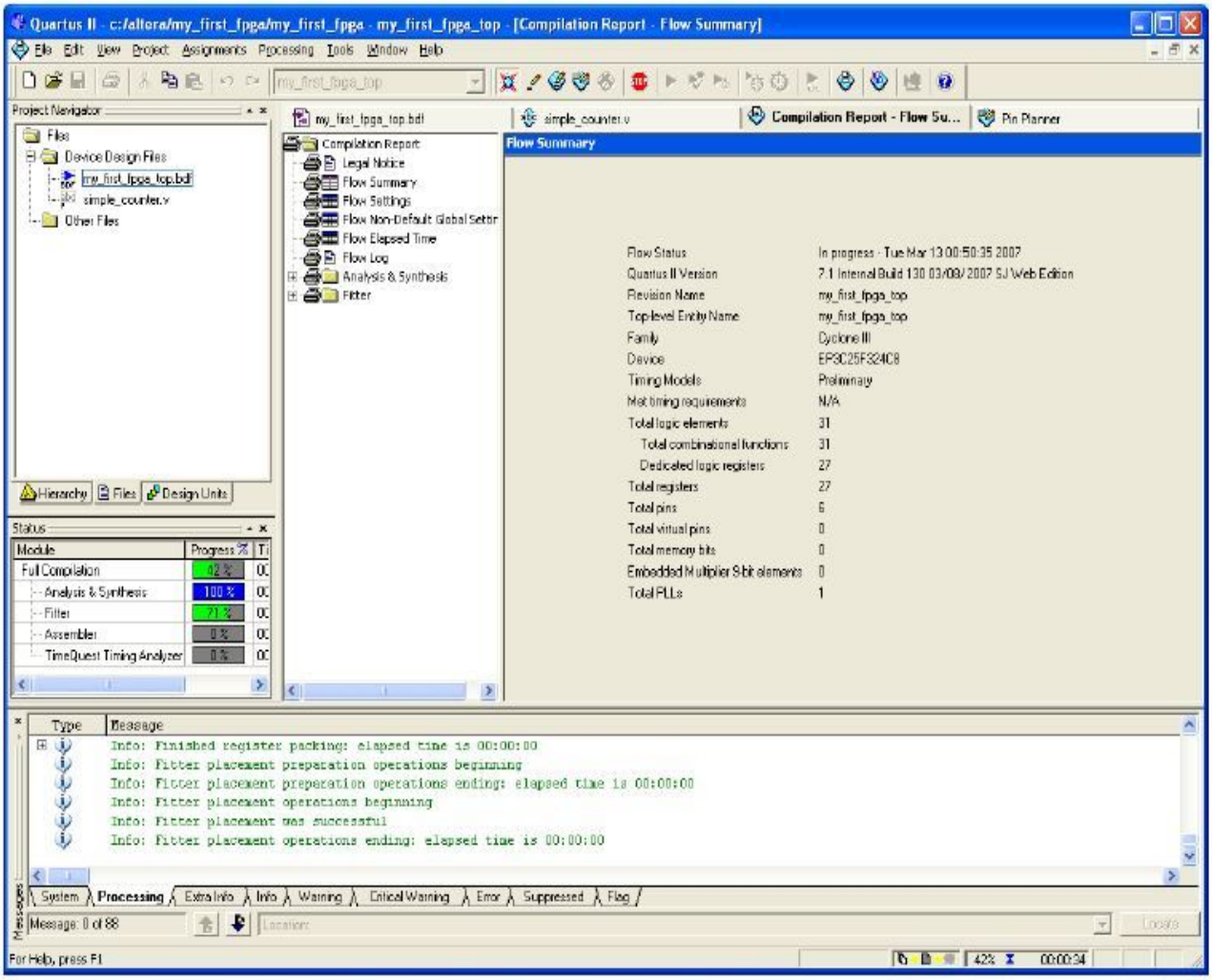


Illustrazione 64: Compilazione

Alla fine del processo il progetto, se non sono stati riscontrati errori, è stato compilato. Possono esserci molteplici "**Warnings**" alla fine della compilazione, ma questi sono dovuti solamente al fatto che alcuni pin non sono stati assegnati o altre informazioni riguardi funzioni temporali. Questi comunque non influiscono sulla correttezza del processo svolto. I risultati della compilazione sono disponibili nel **Compilation Report** finale, dove si possono leggere il numero di pin, porte logiche e registri utilizzati. Se durante l'elaborazione del progetto appare l'errore:

"**Error: current license does not support incremental compilation**"

bisogna modificare il file "***.qsf***" del progetto aggiungendo dall'editor del Quartus II la seguente riga di codice:

"**set_global_assignment_name INCREMENTAL_COMPILATION OFF**"

(informazione ricavata dal forum del sito dell'Altera)

Dopo la compilazione e il debug del progetto realizzato, si può programmare il dispositivo digitale presente nella scheda di sviluppo scaricando il file ***.sof*** nell'FPGA attraverso il circuito USB-Blaster presente a bordo della *Cyclone III Starter Board*. Prima di tutto, bisogna alimentare la scheda connettendola al suo alimentatore fornito nel kit. Successivamente si connette il PC dove è installato Quartus II con la scheda attraverso il cavo USB. Ora si può accendere la scheda premendo il pulsante **SW1** sulla *Cyclone III Starter Board*: il corretto funzionamento della scheda è segnalato dall'accensione di un led verde.

Scegliendo **Tools > Programmer** dai menù principali del Quartus si apre una schermata come illustrato in figura 65.

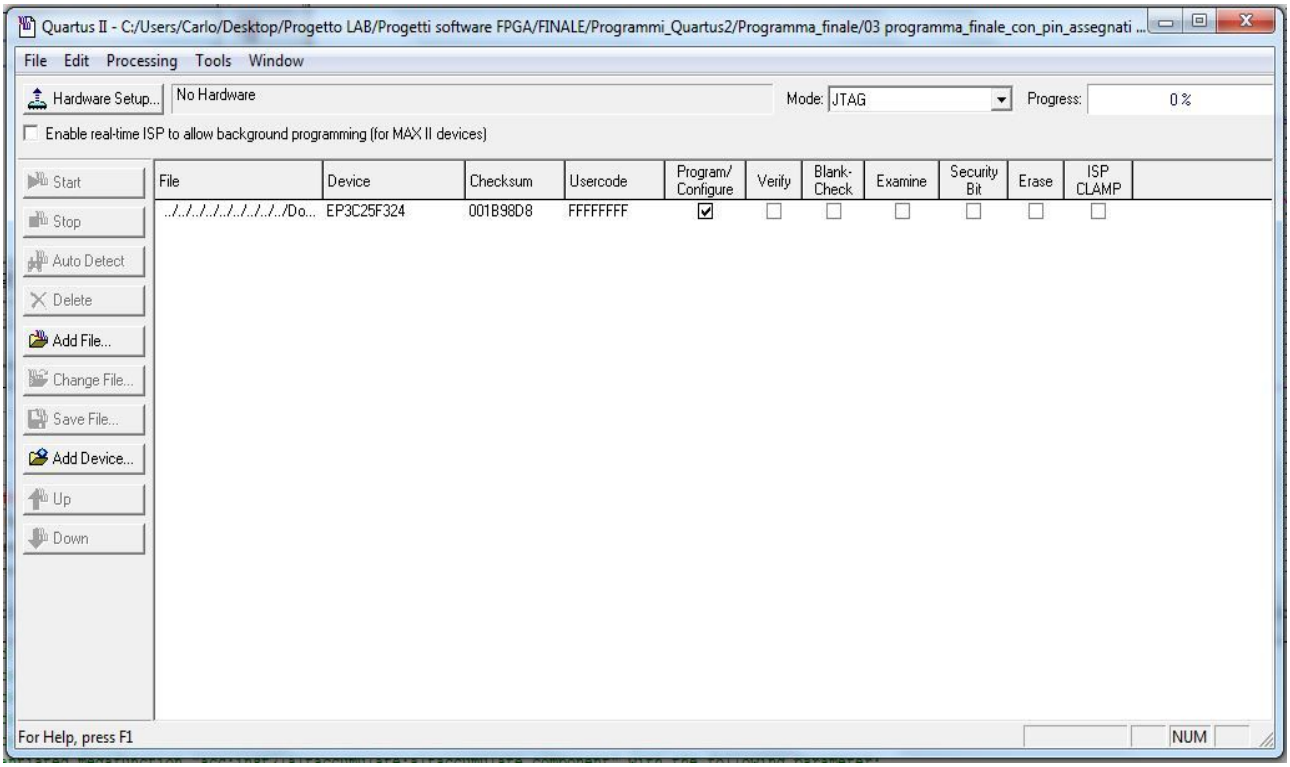


Illustrazione 65: Programmer

Cliccando sulla tab in alto a sinistra **"Hardware Setup"** si seleziona la modalità di connessione alla scheda: sotto la voce **Currently selected hardware** scegliere l'opzione USB-Blaster [USB-0] e cliccare nel tasto "**Add hardware..**". Ora si aggiunge il file ***.sof*** generato dalla compilazione (icona **Add file..**) dalla cartella dove sono contenuti tutti i file riguardanti il progetto realizzato. Dopo aver verificato che sia accesa l'opzione **Program/Configure** di fianco al nome del file selezionato, si può cominciare la programmazione cliccando su "***Start***". Quando lo status della del processo arriva al 100%, l'FPGA è stata programmata ed il progetto realizzato si sta eseguendo real-time da subito.

3.5 Debug e simulazione del progetto software realizzato con Quartus II

Ora che il progetto è completato e l'FPGA programmata si può cominciare a vedere come effettivamente funziona il software realizzato. Prima di connettere il sistema di controllo del motore si effettua un "debug" dell'algoritmo implementato nel dispositivo logico programmabile connettendo un **generatore di funzioni** ai connettori BNC di ingresso della scheda THDB_ADA. In questo modo si crea un segnale analogico periodico (onda quadra, onda triangolare, sinusoide, ecc..) per i convertitori A/D che verrà elaborato dall'FPGA. Si può analizzare l'evolversi dei risultati dell'elaborazione digitale di questi segnali in due modi:

- Prelevando con l'oscilloscopio, attraverso i connettori BNC di uscita della THDB_ADA, i segnali che restituiscono i convertitori D/A.
- Usando il tool messo a disposizione dal Quartus II "**SignalTap II Embedded Logic Analyzer**", che funziona come un oscilloscopio "virtuale": l'evoluzione temporale di qualsiasi ingresso, uscita o nodo dello schematico realizzato può essere visualizzato sullo schermo del PC, sia in forma analogica che numerica.

Il **SignalTap II Embedded Logic Analyzer** è uno strumento, messo a disposizione dal programma Quartus II, che permette di esaminare i segnali che si propagano nei vari nodi del dispositivo digitale senza l'utilizzo di pin di I/O aggiuntivi, mentre l'FPGA sta funzionando. E' quindi uno strumento molto utile per il "debug" del programma, in quanto permette l'analisi precisa dell'evoluzione delle uscite del sistema o dei nodi interni senza l'utilizzo di dispositivi aggiuntivi esterni.

Dopo aver compilato correttamente il progetto e completato l'assegnazione dei pin, si seleziona dal Menù principale del Quartus II **File > New > SignalTap II Logic Analyzer File**.

A questo punto apparirà la schermata del SignalTap illustrata in figura 66. Dopo aver salvato il file con estensione **.stp** viene chiesto se si vuole associare il file al progetto corrente. Cliccando su OK si è realizzato un SignalTap File per il programma realizzato.

Ogni schematico può avere più file .stp associati, ognuno che analizza diversi segnali, ma solo uno per volta si può eseguire (opzione utile solo per progetti molto complessi). Se, durante questa sequenza di comandi per creare un SignalTap File, appare l'errore "SignalTap not avaible under the current license", selezionare dal Quartus II i comandi Tools > Options. Nel menù General, selezionare "Internet Connectivity" e cliccare il pulsante "Talckback Options". Selezionare la casella "Turn on TalkBack feature" e poi OK. Poi spegnere e riaccendere Quartus II per rendere effettiva la modifica.

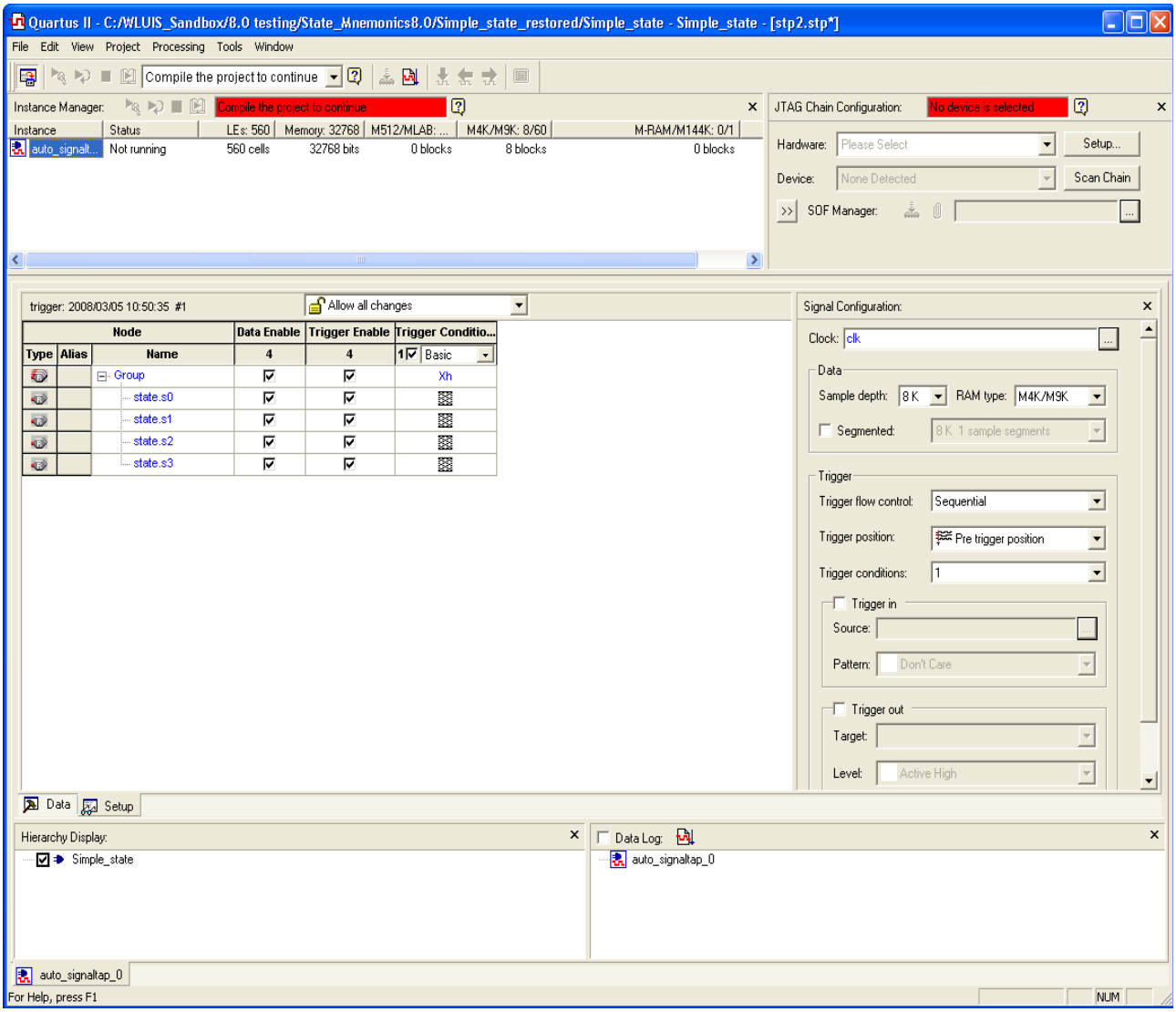


Illustrazione 66: Schermata del SignalTap II

Per eseguire il SignalTap, è necessario configurare l'hardware. Selezionando **Setup**, come illustrato in figura 67,

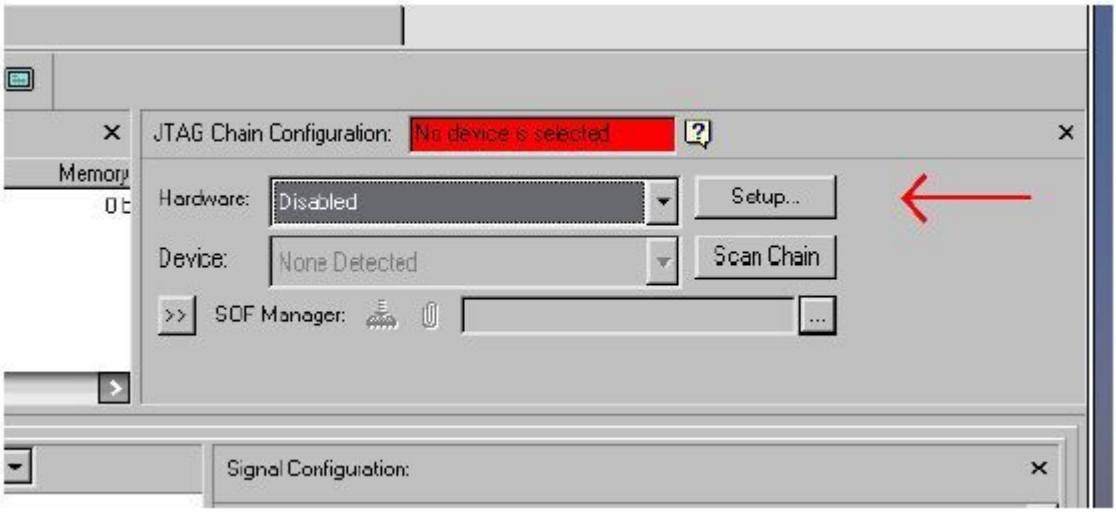
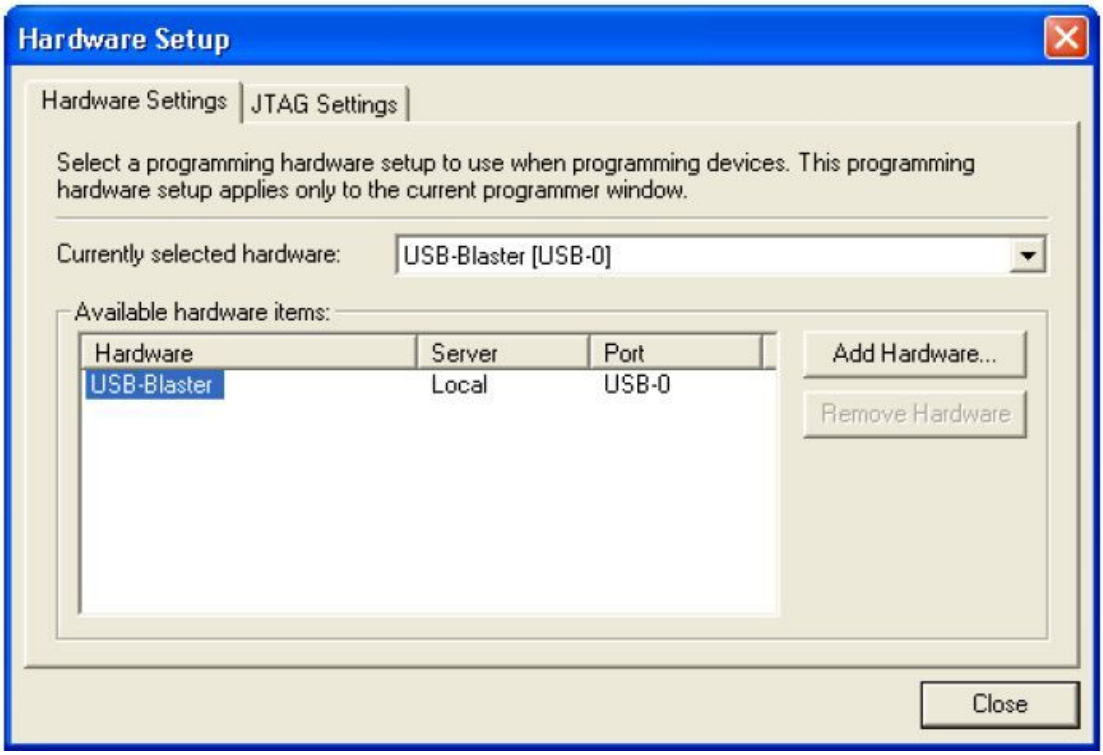
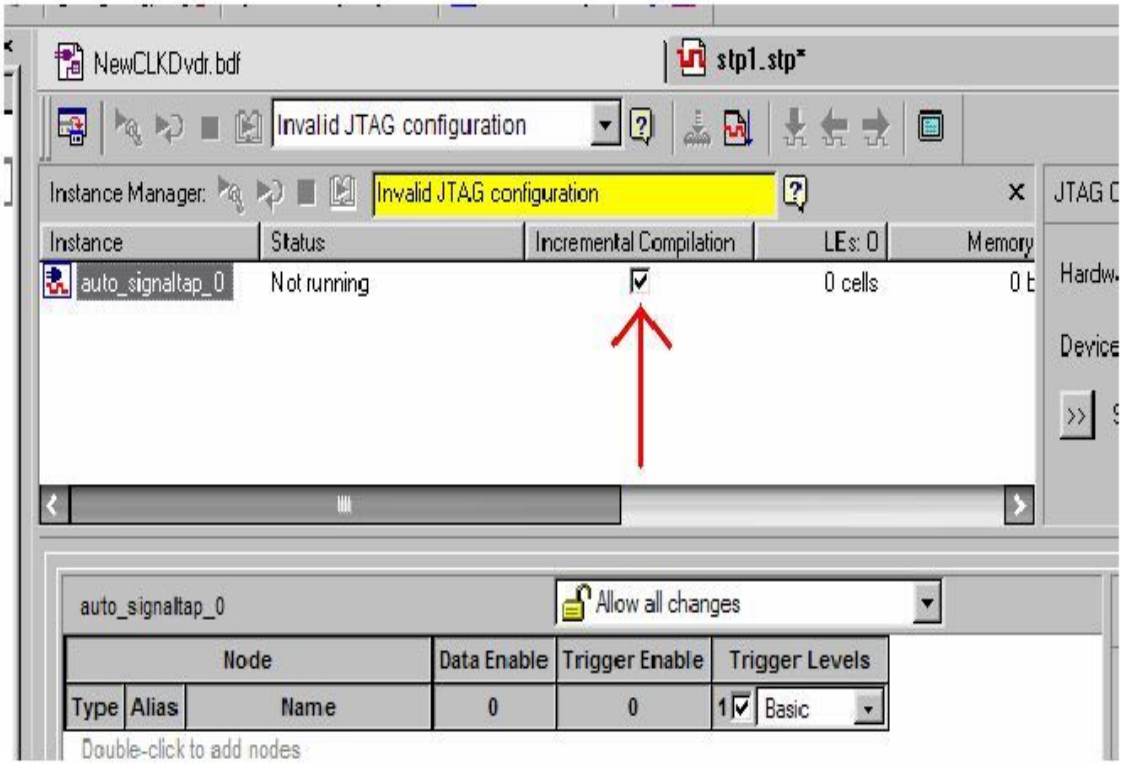


Illustrazione 67: Tasto di Setup per configurare l'hardware

appare una finestra, come la seguente, da dove si seleziona USB-Blaster [USB-0] sotto la voce "Currently selected hardware":



Nell'**Istance Manager**, togliere l'opzione “Incremental Compilation”:



Ora è il momento di decidere quali nodi del progetto si vogliono visualizzare con il SignalTap. Cliccando due volte nella scritta a centro pagina “ Double-click to add nodes”, si apre il **Node Finder**, cioè una lista di tutti i possibili segnali che possono essere visualizzati, in base allo schematico realizzato. Ogni nodo che si vuole visualizzare deve avere un nome oppure gli deve essere assegnato un Pin Input/Output per essere analizzato con il SignalTap.

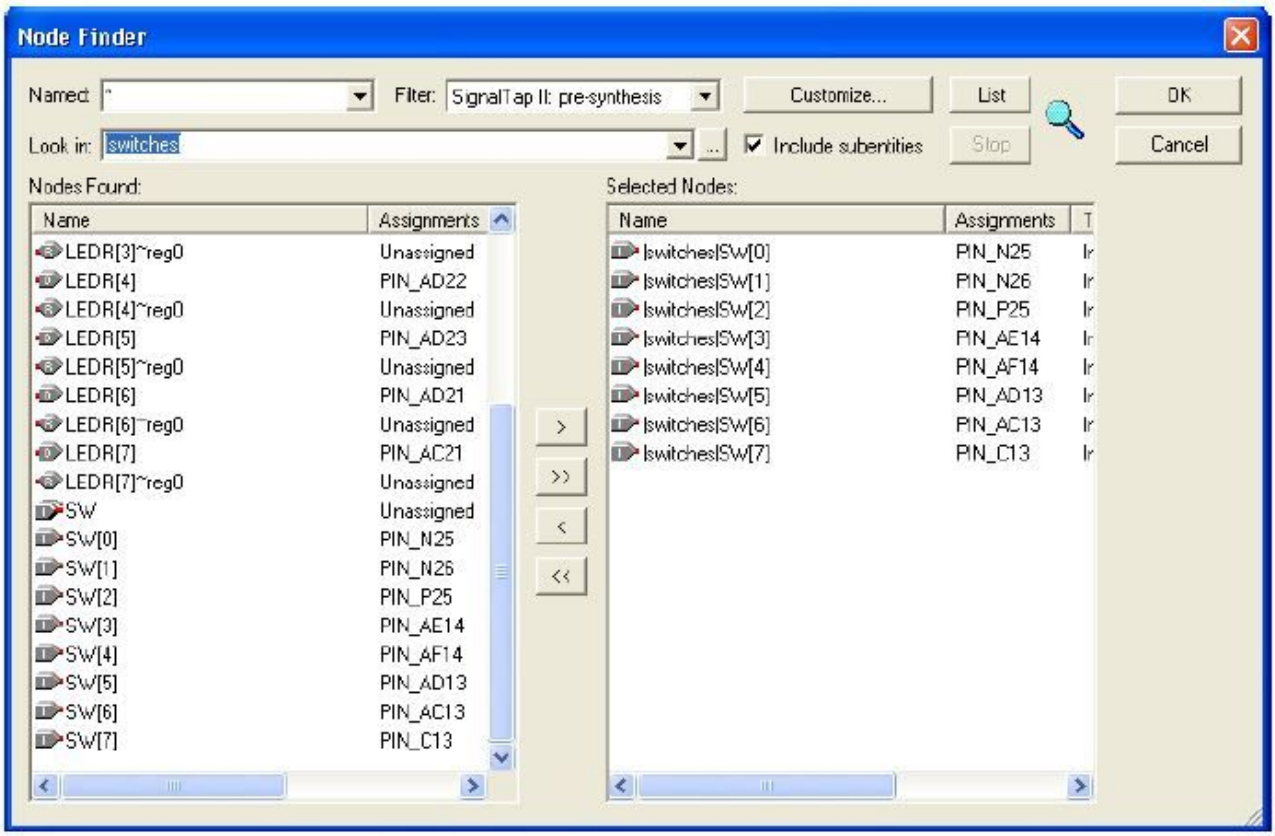


Illustrazione 68: esempio di Node Finder

Per creare una lista, selezionare nel campo “Filter” la voce “ SignalTap II: pre-synthesis” e cliccare sul tasto **List**. A questo punto appare un elenco dei possibili nodi visualizzabili. Si è deciso di visualizzare, in questo progetto, i nodi **ingresso_A**, **uscita_A** e **sync.** Per selezionarli basta cliccare sul nome del nodo e premere il pulsante >.

E' necessario specificare anche a che frequenza deve lavorare il modulo SignalTap generato. Per assegnarli un clock, selezionare la finestra **Setup** del SignalTap, illustrata nella figura 69, e alla voce “Signal configuration” scrivere il nome del nodo che corrisponde al clock che si vuole assegnare al modulo SignalTap. In questo progetto si è deciso di creare un clock apposito a 125 MHz, generato da un altro modulo PLL, aggiunto nello schematico, la cui uscita è contrassegnata da un Pin Output denominato **clk_SignalTap**. Maggiore è la frequenza di clock per il SignalTap, migliore è la risoluzione della forma d'onda visualizzata. Il segnale utilizzato come clock non può essere visualizzato e analizzato nel SignalTap.

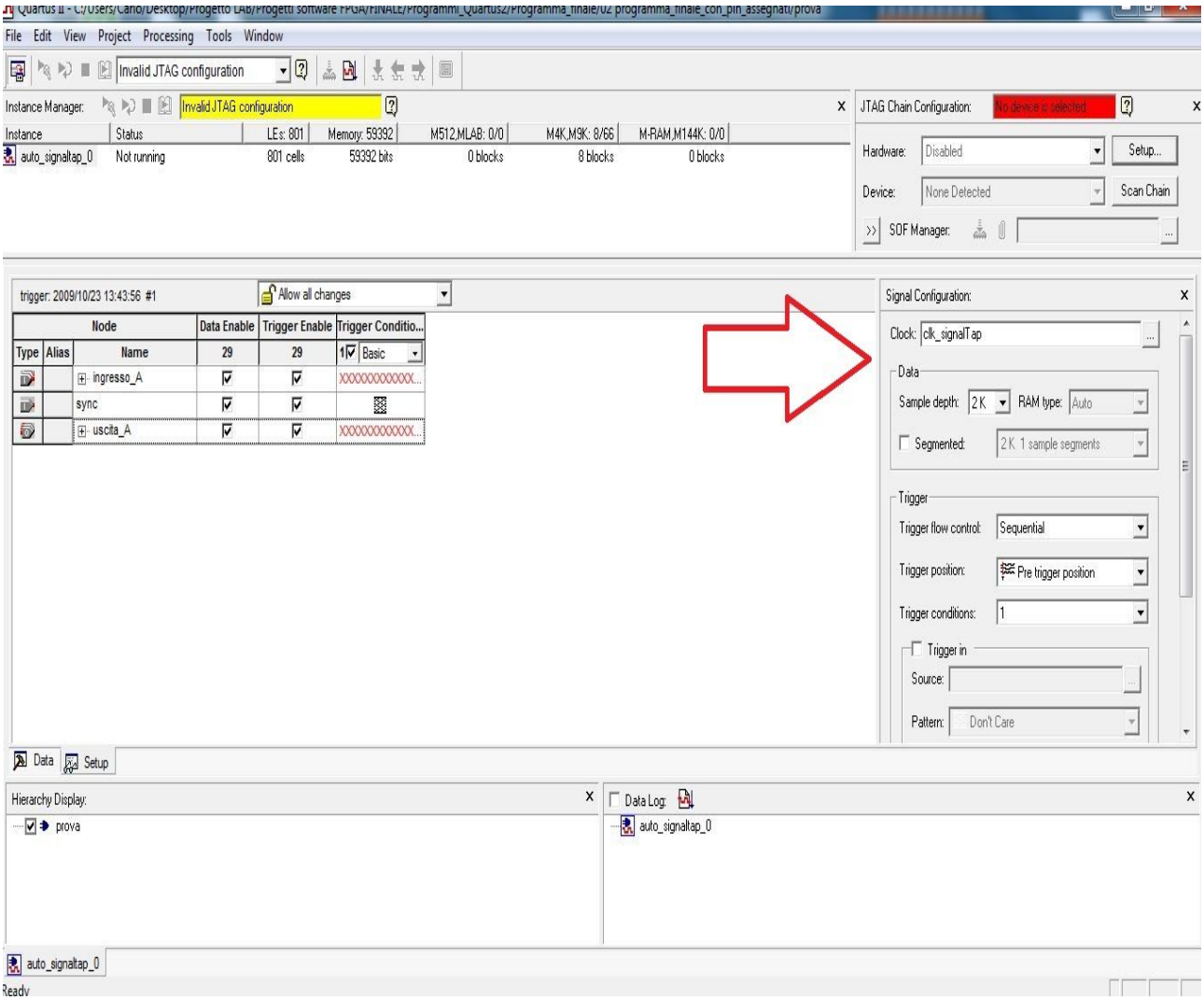


Illustrazione 69: Finestra Setup del SignalTap: la freccia rossa segnala dove aggiungere il segnale di clock

Il SignalTap mette a disposizione molte altre possibilità di configurazione del programma per analizzare nel migliore dei modi i segnali a cui si è interessati, come la possibilità di inserire una condizione di Trigger (al verificarsi di un evento il segnale viene acquisito e visualizzato sullo schermo) o la profondità dei dati campionati, ma per questo progetto non sono necessarie.

Per rendere effettive le modifiche apportate al sistema così configurato è necessario ricompilare il progetto e ri-programmare l'FPGA come descritto nel paragrafo precedente. Una volta concluse correttamente queste operazioni si può eseguire il SignalTap tornando nella sua schermata principale e cliccando su "***Autorun Analysis***", per aggiornare continuamente i segnali acquisiti, o "***Run Analysis***", per acquisire le forme d'onda una sola volta.

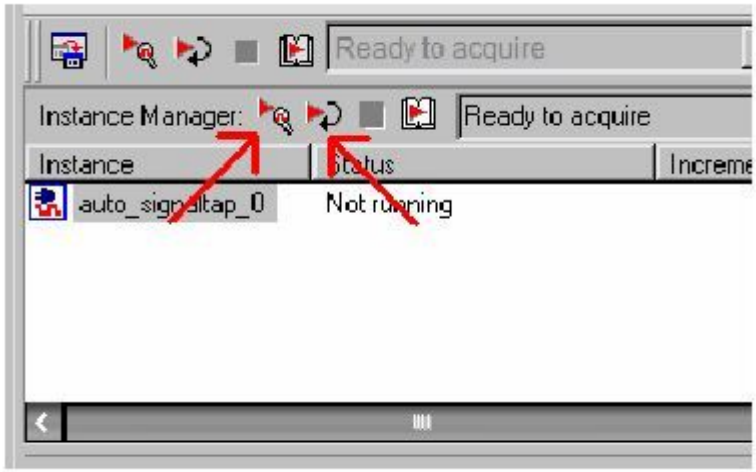


Illustrazione 70: Contrassegnate dalle frecce rosse "Autorun Analysis" o "Run Analysis"

Il generatore di funzioni utilizzato per questa analisi genera sia l'ingresso analogico per il convertitore A/D (canale A), sia un segnale ad onda quadra per il **sync**, che entra direttamente nel pin assegnato al sincronismo dell'FPGA. Essendo generati dallo stesso dispositivo, questi due segnali sono in fase. Questa condizione è la stessa che avviene nel sistema con l'azionamento del motore: infatti ogni periodo di PWM che viene elaborato dall'FPGA è in fase con il segnale di sincronismo con duty-cycle al 50%.

Si è scelto di impostare come ingresso analogico una sinusoide di ampiezza $1V_{pp}$ a 12kHz a valor medio nullo (offset = 0). Integrando sul semiperiodo una sinusoide, in uscita si trova un segnale ad onda quadra, in fase con sync e di ampiezza proporzionale al valore medio della sinusoide sul semiperiodo precedente.

Il segnale analogico dal generatore di funzioni passa per un circuito, con l'integrato AD8138, che attenua il segnale di un fattore 1,5. Il motivo di questo fattore di attenuazione e la configurazione dello stadio interposto tra l'ingresso e l'A/D è spiegato nel paragrafo 3.5.1.

Per vedere le differenze fra le uscite visualizzate con il SignalTap e i segnali che vengono convertiti dal D/A è connesso anche un oscilloscopio ai BNC della scheda THDB_ADA. In questo modo si può analizzare il segnale analogico, dopo l'elaborazione digitale, sia visualizzandolo con il SignalTap, sia analizzando le immagini dall'oscilloscopio. E' importante configurare l'impedenza del generatore di funzioni a 50Ω. Questo perché l'impedenza degli ingressi della scheda Terasic THDB_ADA è di 50Ω. In questo modo si può visualizzare correttamente, attraverso l'oscilloscopio, l'effettivo segnale che arriva ai convertitori A/D.

Nelle figure seguenti, che riportano i risultati ottenuti con il SignalTap, il primo segnale visualizzato (dall'alto) è l'ingresso analogico che arriva al convertitore A/D, il secondo è il segnale di sincronismo in fase con l'ingresso, mentre il terzo è l'uscita dei convertitori D/A dopo l'elaborazione

digitale. Ognuno di essi può equivalentemente essere visualizzato con il suo corrispettivo digitale, cioè i bit da cui è formato.

Le figure 71 e 72 illustrano i risultati ottenuti:

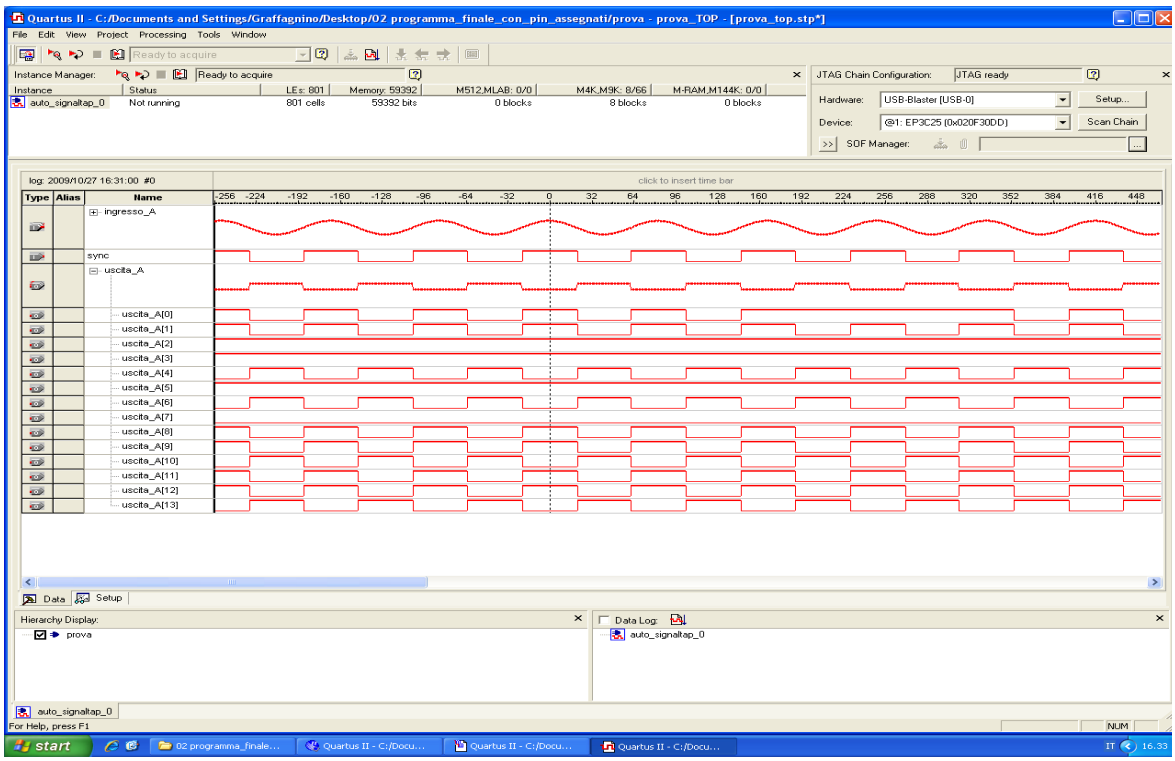


Illustrazione 71: SignalTap: ingresso=sinusoide 1Vpp a 12kHz

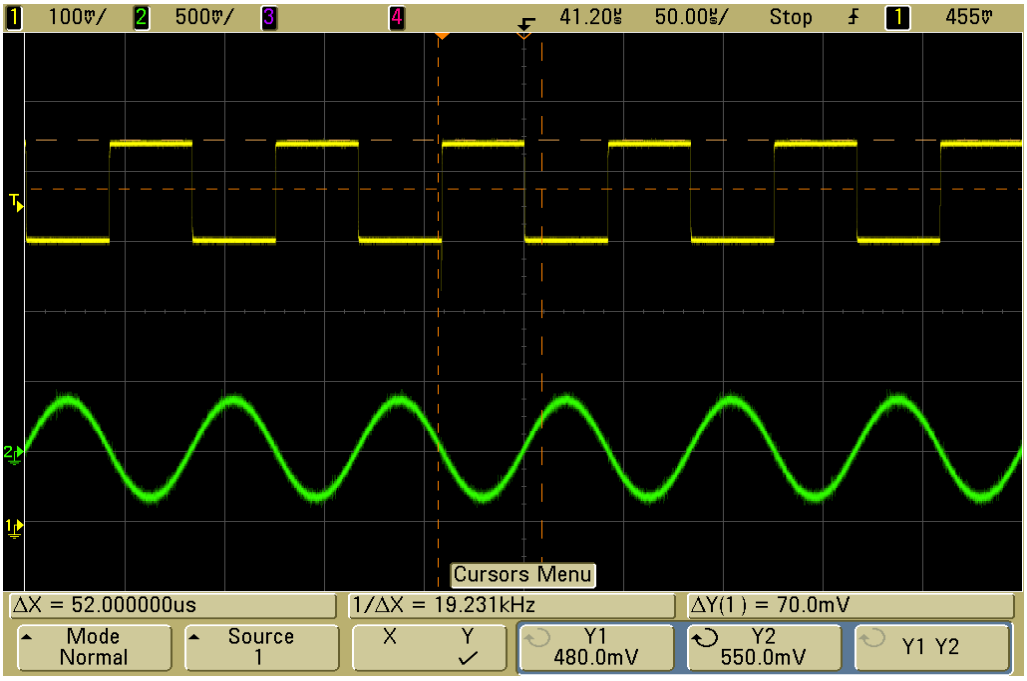


Illustrazione 72: Oscilloscopio:ingresso=sinusoide 800mV a 12kHz (VERDE) uscita= onda quadra (GIALLO)

Successivamente sono stati posti segnali d'ingresso con caratteristiche diverse dal precedente per studiare le variazioni in uscita. Ecco alcuni esempi:

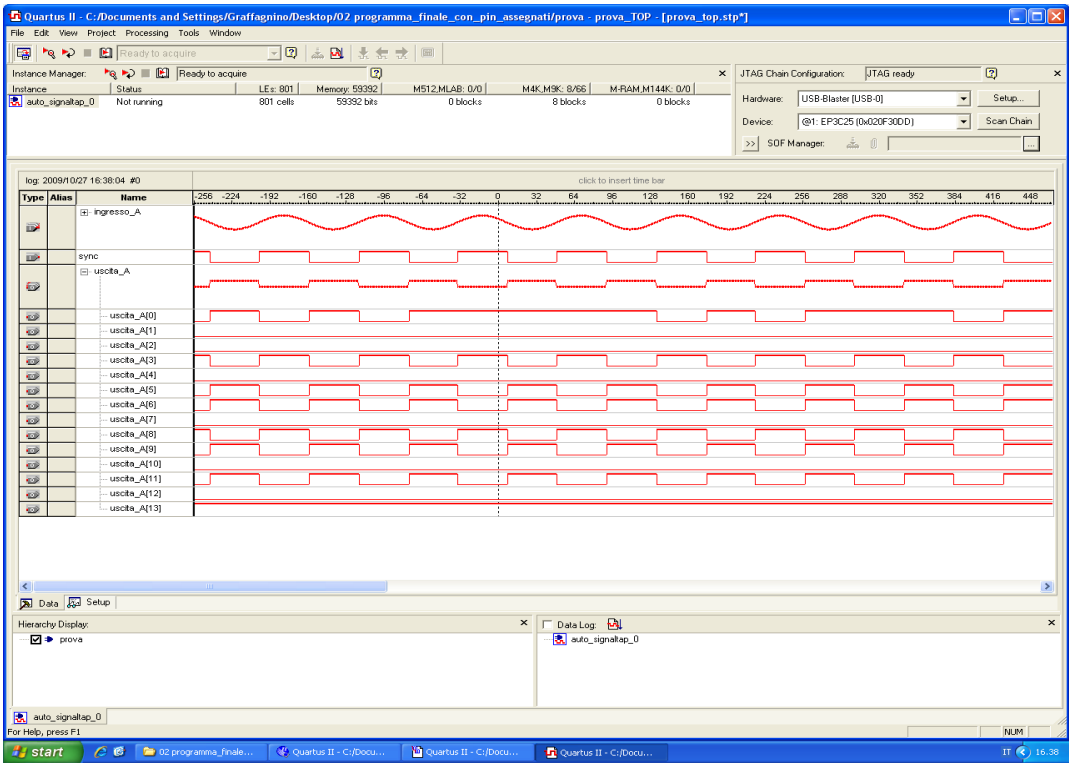


Illustrazione 73: SignalTap: ingresso=sinusoide 1Vpp a 12kHz con offset 400mV

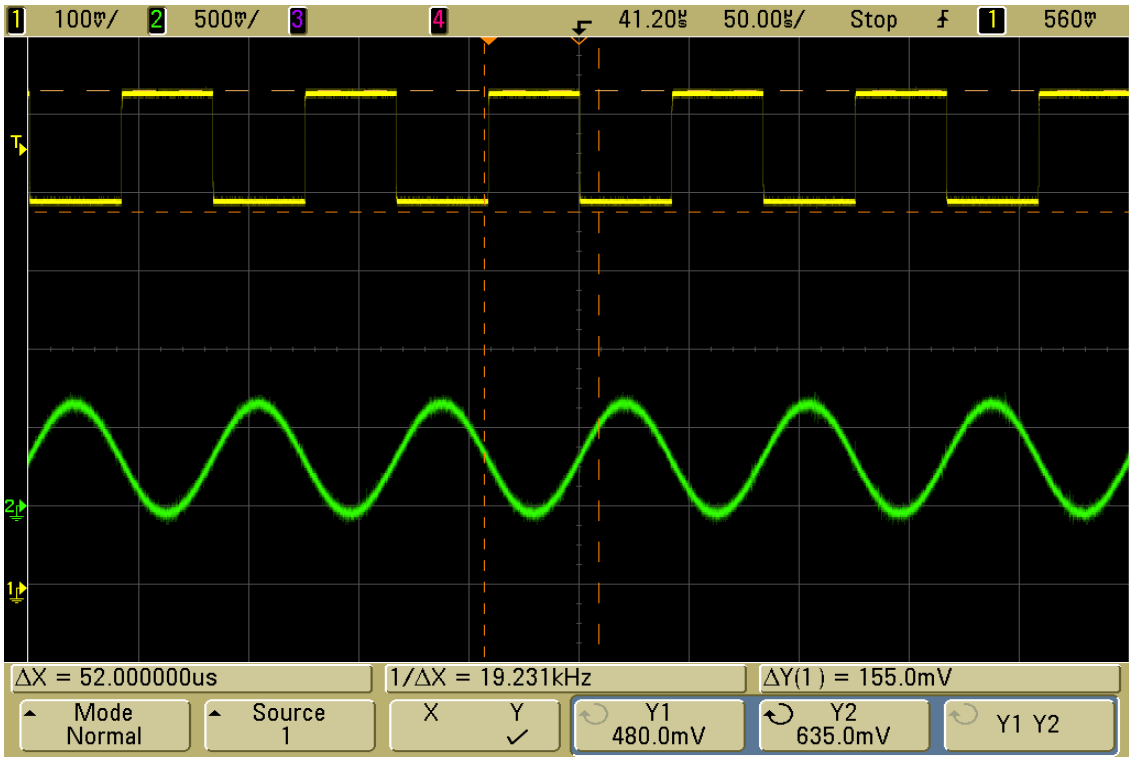


Illustrazione 74: Oscilloscopio

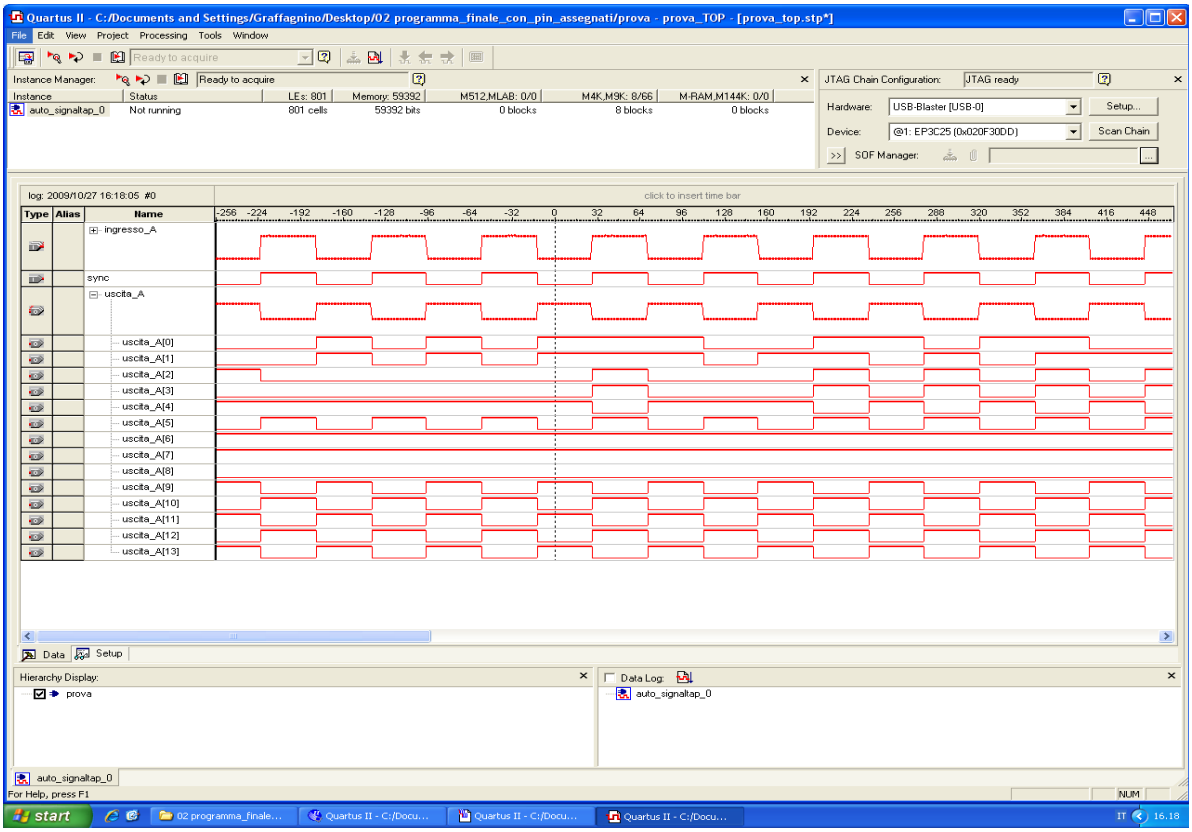


Illustrazione 75: SignalTap: ingresso=onda quadra 1,5Vpp

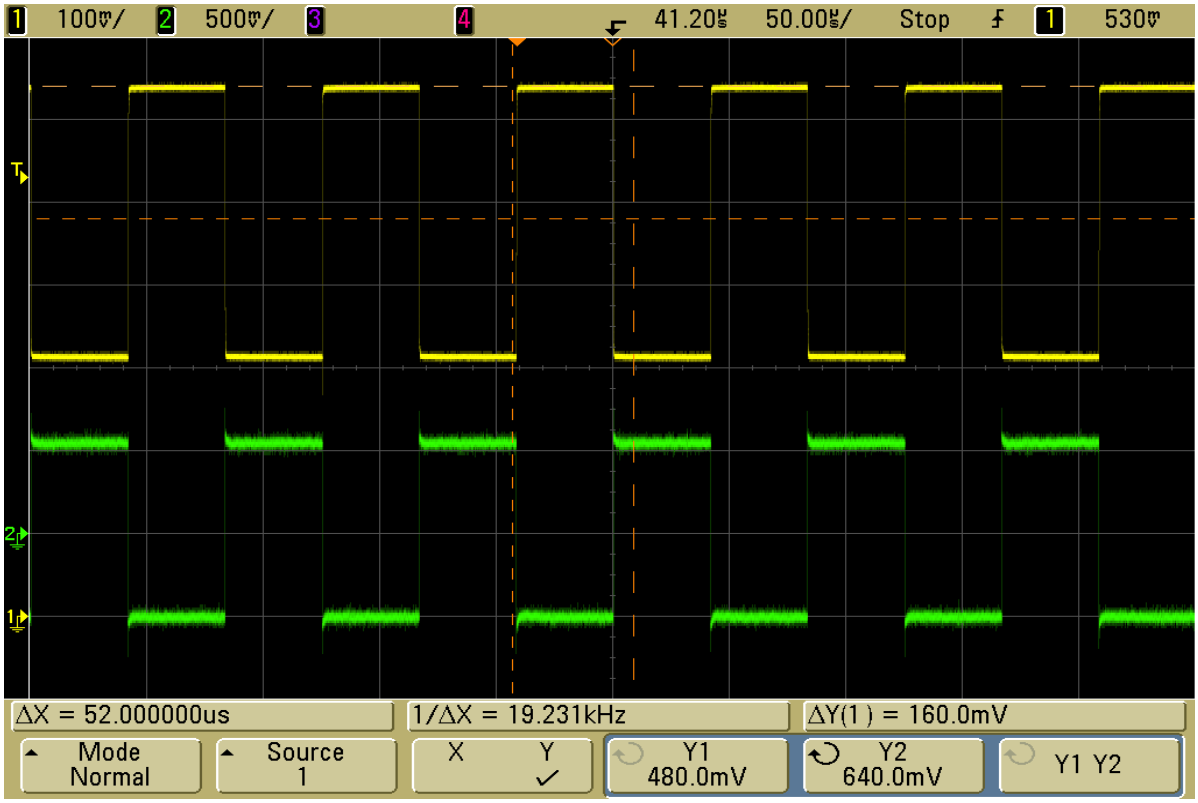


Illustrazione 76: Oscilloscopio

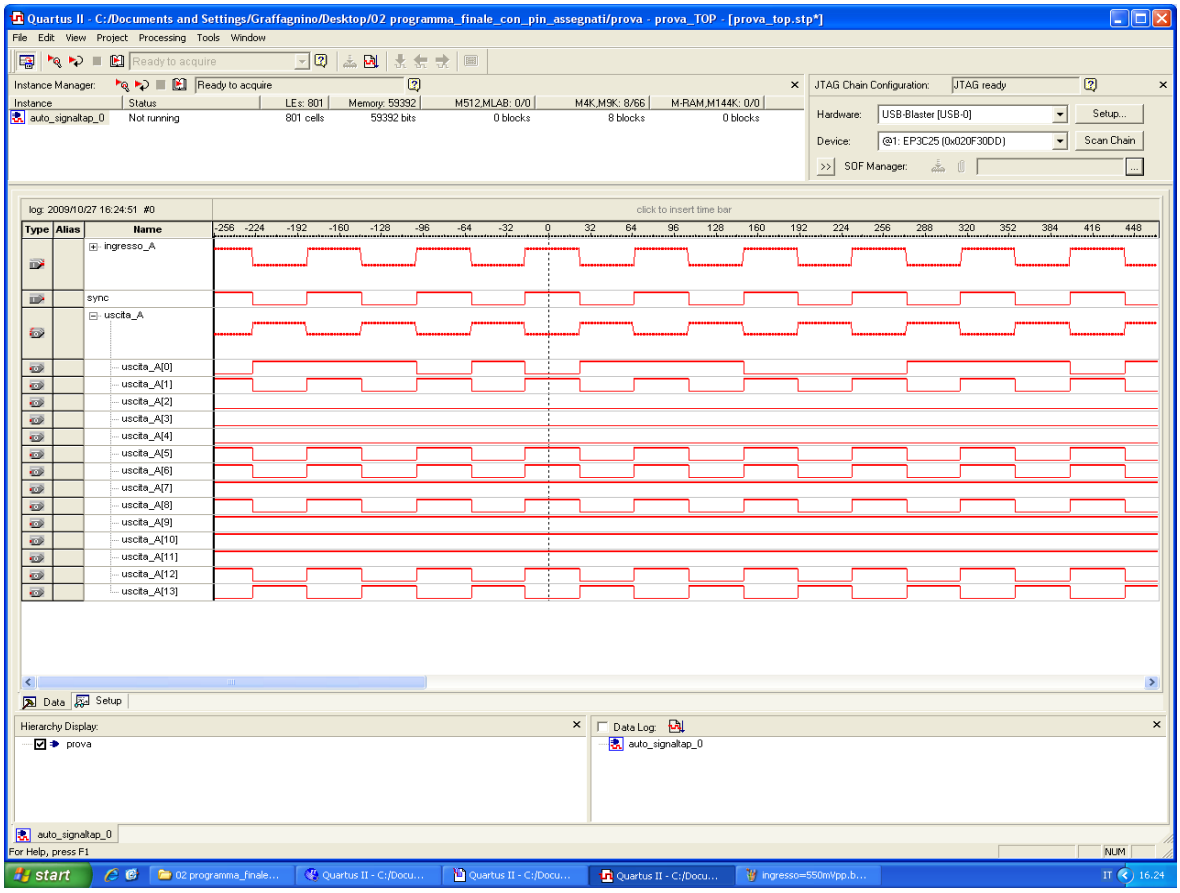


Illustrazione 77: SignalTap: ingresso=onda quadra da 0V a 1V

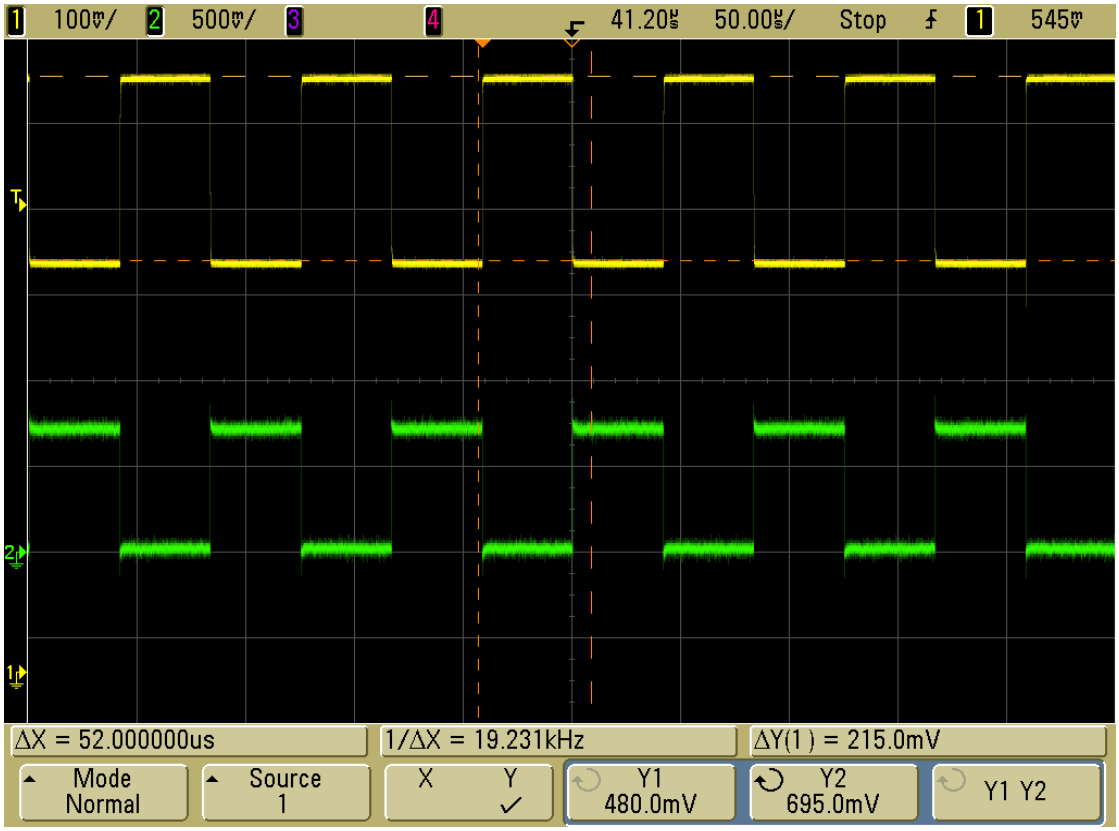
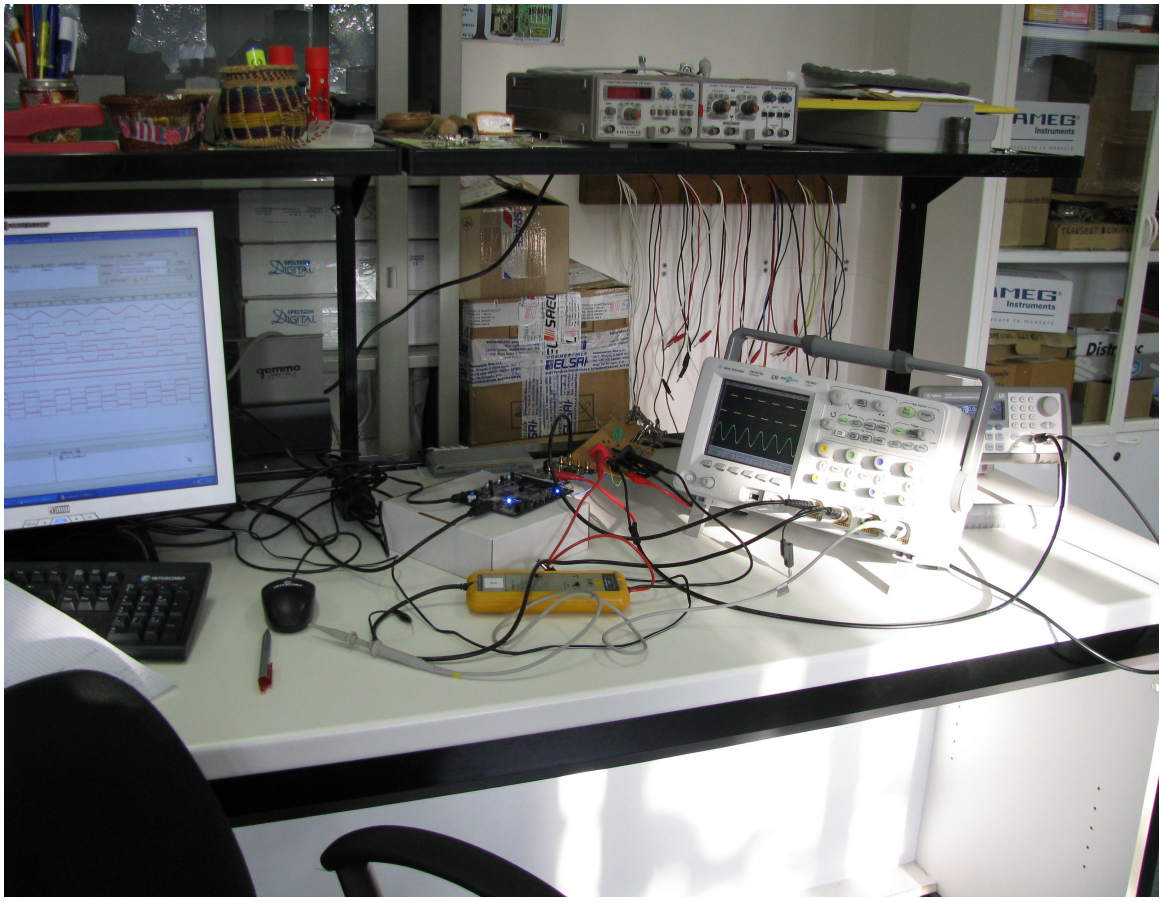
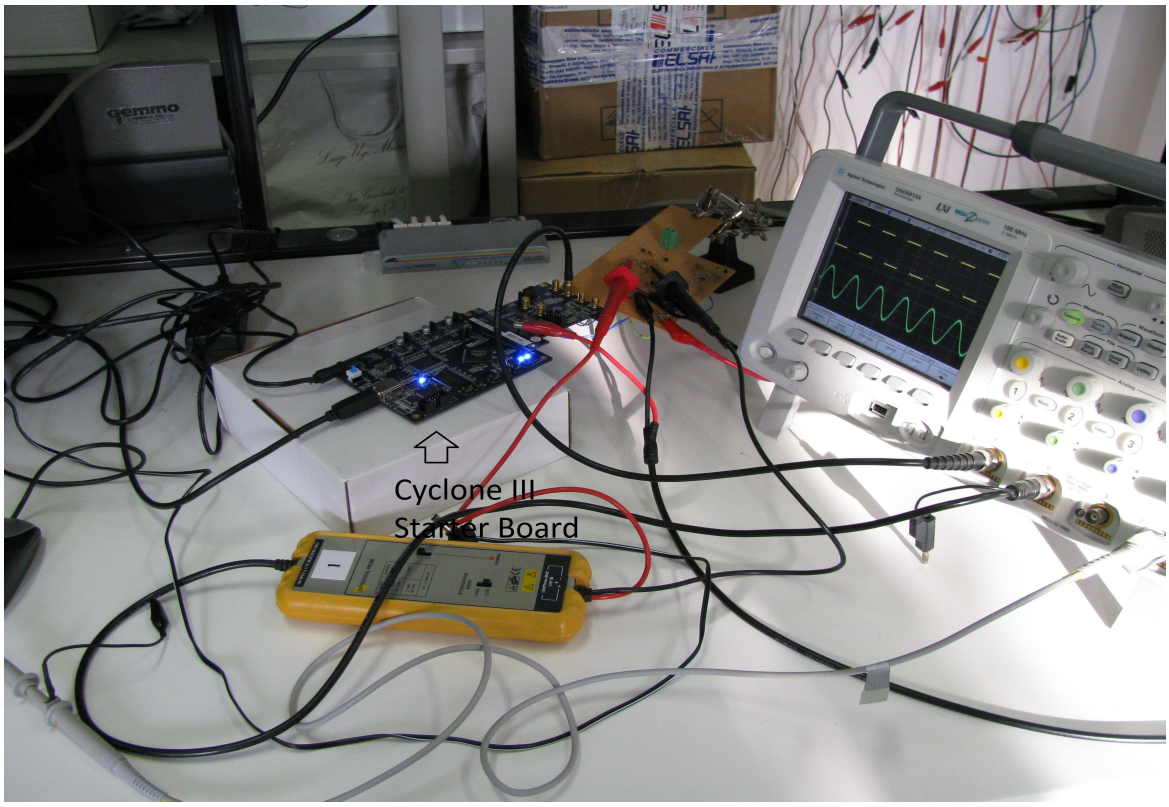


Illustrazione 78: Oscilloscopio

Di seguito sono riportate alcune foto del sistema di misurazione realizzato:





Alternativamente, si può effettuare un “debug” preliminare del software realizzato attraverso il **Quartus II Simulator**. Questo tool permette di effettuare sia simulazioni funzionali che temporali impostando un segnale d'ingresso predefinito ed analizzando le evoluzioni delle uscite o dei nodi intermedi.

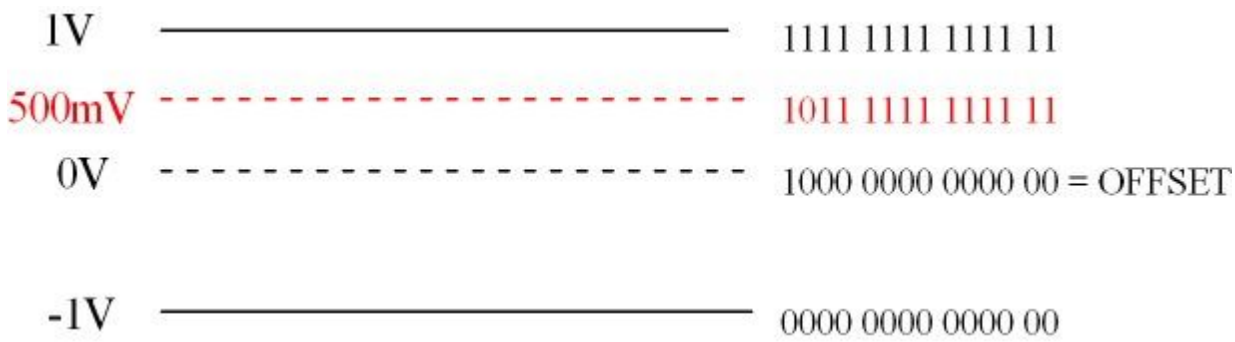
Ovviamente questo tipo di simulazione è molto meno precisa di quella effettuata con il SignalTap o un oscilloscopio esterno che visualizza i risultati ottenuti, ma è molto utile nelle fasi intermedie di realizzazione del software per vedere l'evoluzione dei valori digitali nei nodi intermedi dello schematico.

Per creare un segnale d'ingresso per il progetto realizzato, si seleziona **New > Vector Waveform File**. A questo punto viene creato un file con estensione **.vwf** e appare una finestra da dove è possibile generare un segnale digitale a proprio piacimento, periodico o casuale per ogni PinInput inserito nello schematico. Dopo aver compilato e programmato l'FPGA, si seleziona dal menù di **Processing "Generate Functional Simulation Netlist"**. Quindi, selezionando **Settings > Simulator Settings**, viene configurato il simulatore, impostando il tipo di simulazione che si vuole effettuare (funzionale o temporale) ed associando il file .vwf, creato precedentemente, al progetto.

Dopo questa configurazione, si può iniziare la simulazione selezionando **Start Simulation** dal menù di **Processing**. Si apre quindi una finestra con l'evoluzione dei nodi selezionati in base agli ingressi impostati precedentemente.

3.5.1 Calcoli sui rapporti ingresso/uscita ottenuti durante la simulazione con ingressi variabili

Si effettua ora un calcolo teorico dell'elaborazione digitale di un segnale ad onda quadra +/-500mV a valor medio nullo ($1V_{pp}$) e lo si confronta con le misurazioni effettuate illustrate nelle figure 71 e 72. Gli stessi ragionamenti si possono fare con qualsiasi altro ingresso analogico.
Questo segnale ha una frequenza di 12kHz, la stessa del segnale di sincronismo utilizzato. Il SYNC e l'ingresso ad onda quadra sono in fase. Nel semiperiodo positivo, il convertitore A/D campiona un valore costante di 500mV a una frequenza di 65MHz nella codifica *offset-binary* a 14 bit. Sapendo che l'A/D ha un range di campionatura di $2V_{pp}$, si può facilmente ricavare come venga decodificato un segnale analogico di ampiezza 500mV:



In"*offset-binary*" **1V** viene codificato come 1111 1111 1111 11 = 8191. Quindi **500mV** viene codificato come 8191/2 = 4095 = 1011 1111 1111 11 che, convertito in "*complemento a due*", diventa 0011 1111 1111 11.
Quest'ultimo numero digitale è il valore costante a 14 bit che viene integrato per un semiperiodo di PWM, cioè per (1/2*12kHz) = **41,67µs**. In questo intervallo di tempo vengono sommati algebricamente 41,67µs*65MHz = 2709 campioni digitali che codificano i 500 mV in ingresso.
Quindi, sommando il numero 4095 per 2709 volte, si ottiene il valore 4095*2709 = 11093355 in un registro a 26 bit in "*complemento a due*", in base all'algoritmo descritto nei paragrafi precedenti:

11093355 = 0010 1010 0101 00 010101101011 in "*complemento a due*"

Per codificare il numero in "*offset-binary*", basta invertire il bit più significativo:

11093355 = 1010 1010 0101 00 010101101011 in "*offset-binary*"

Il convertitore D/A in uscita decodifica numeri digitali in ”*offset-binary*” a 14 bit, quindi si effettua un “troncamento” del valore memorizzato nel registro a 26 bit considerando solamente i 14 bit più significativi (cifre in rosso). Per fare questo viene applicato uno shift a destra di 12 posizioni del registro a 26 bit: nel sistema binario questo significa dividere il numero shiftato per $2^{12} = 4096$.

11093355 /4096 = 2708 = 1010 1010 0101 00 in ”*offset-binary*”

Si può valutare l'inevitabile errore di approssimazione compiuto eseguendo il troncamento: se i 12 bit meno significativi vengono tralasciati (cioè come considerarli uguali a 0) su 26 bit viene rappresentato il numero 11091968. L'errore introdotto è pari quindi a
(11093355-11091968)/11093355 = $1{,}25*10^{-4}$ = 0,0125%.
Può quindi essere considerato del tutto trascurabile.
Il segnale analogico decodificato dal D/A è un segnale di corrente da 0 a 20mA:

I numeri positivi sono decodificati in un range di 10mA, tra 10mA e 20mA. Il convertitore ha quindi una risoluzione pari a 10mA/(2^{13}-1) = $1{,}22*10^{-6}$A.
In base a queste considerazioni, il convertitore D/A decodifica il valore digitale
1010 1010 0101 00 = 2708 in un segnale di corrente pari a 2708*$1{,}22*10^{-6}$A = 3,3mA sopra il valore di offset di 10mA. Quindi, alla fine del semiperiodo di PWM, si è ottenuto un segnale analogico di corrente di 13,3mA. Tra il D/A e il connettore BNC della scheda THDB_ADA c'è una resistenza da 50Ω, connessa tra l'uscita del convertitore e massa. In questo modo il segnale di corrente viene trasdotto in un valore di tensione, ottenendo quindi 13,3mA*50Ω = 665mV.

Si riporta ora nuovamente la figura 71 per confrontare le misure effettuate con i calcoli appena svolti:

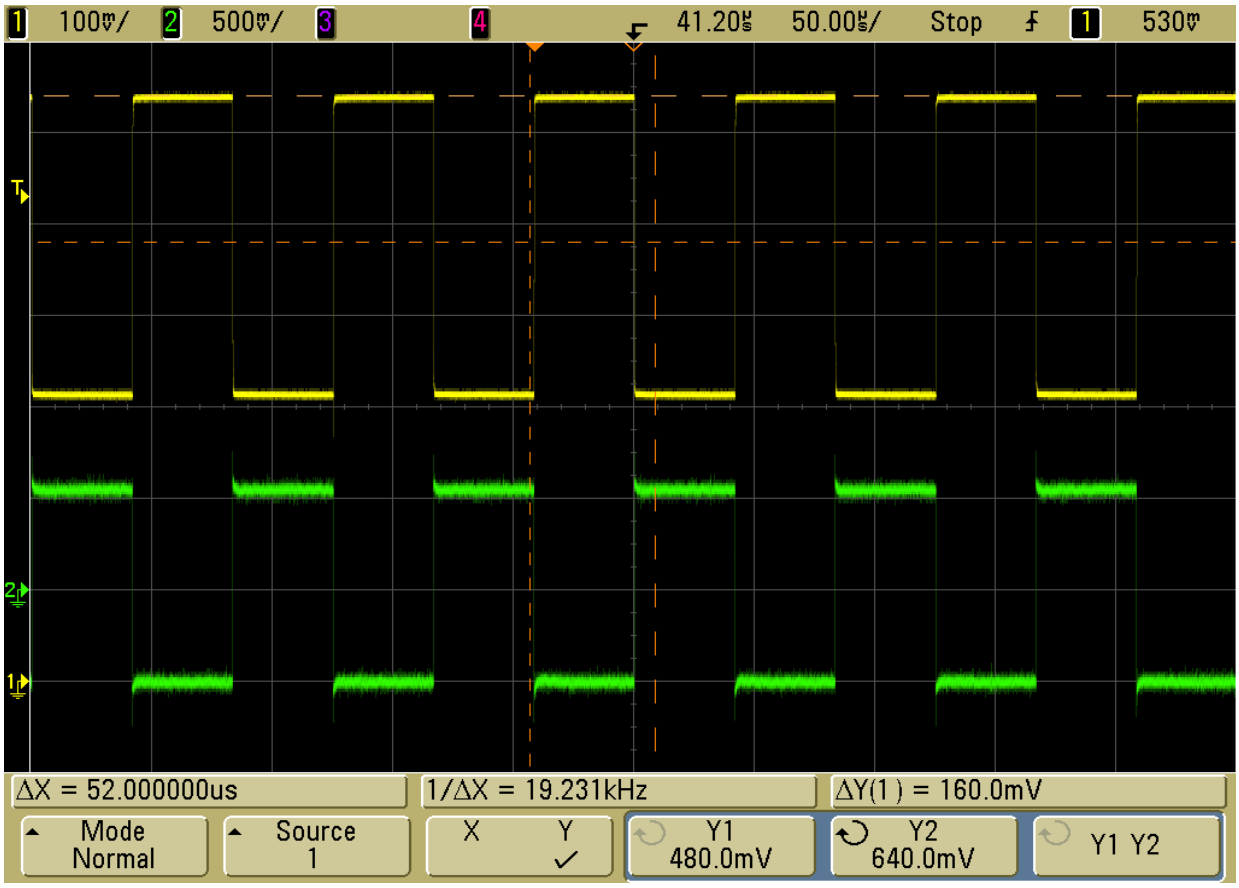


Illustrazione 79: VERDE=ingresso al conv. A/D GIALLO= uscita del conv. D/A

Come si può notare dall'illustrazione 79, l'offset del segnale a onda quadra in uscita è di 480mV, diverso dal valore atteso di 500mV. Questo perché la resistenza da 50Ω, che converte la corrente in uscita dal D/A in un valore di tensione, ha una tolleranza, rispetto al suo valore nominale, del 5%: quindi il suo valore può variare da 47,5Ω a 52,5Ω. In base alle misure riportate, il suo valore è di **48Ω**. Anche il risultato dell'integrazione della semi-onda positiva in ingresso all'A/D è quindi leggermente inferiore rispetto ai 165mV attesi sopra il livello di offset, e pari a 3,3mA*48Ω = 159mV, praticamente identico a quello rilevato con l'oscilloscopio e mostrato nell'illustrazione 79.

3.6 Progetto ed esecuzione delle modifiche hardware alla scheda di acquisizione A/D e D/A

Dopo aver tolto i trasformatori RF posti all'ingresso della scheda THDB_ADA, che filtravano tutte le componenti del segnale da integrare fino a 0,4 MHz, è stata predisposto un nuovo circuito dove si è utilizzato l'integrato **AD8138** della Analog Devices.

Questo componente è un amplificatore operazionale predisposto per la conversione del segnale da single-ended a differenziale, ed è particolarmente adatto per bilanciare l'ingresso differenziale degli ADC, massimizzando le prestazioni del convertitore. Permette inoltre di applicare al segnale in ingresso un fattore di amplificazione e di preservare le sue componenti a bassa frequenza, fino alla continua.

L'azionamento utilizzato per questo progetto ha tensioni di fase del motore che possono arrivare fino a $600V_{pp}$. Lo stadio di riduzione della scheda analogica ha un fattore di attenuazione 1/200, ottenendo quindi un segnale bipolare $3V_{pp}$ (al massimo). Ma il convertitore A/D accetta segnali analogici bipolari in ingresso di ampiezza $2V_{pp}$. Si deve quindi ridurre ulteriormente il segnale che arriva dalla scheda analogica di misura di un fattore **1/1,5,** attraverso un'opportuna configurazione dell'integrato AD8138, come mostrato nell'illustrazione 80:

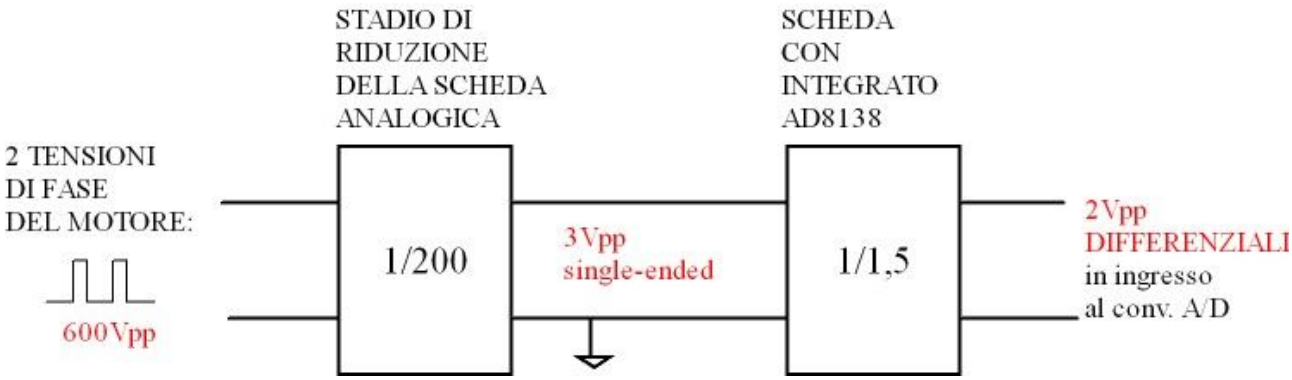


Illustrazione 80: Schema a blocchi di riduzione del segnale d'ingresso

La configurazione dell'AD8138 che più si addice alle richieste di questa progetto è suggerita nel datasheet stesso del componente, dove vengono descritte le più usuali applicazioni dell'integrato e dal quale è stata tratta la figura 81:

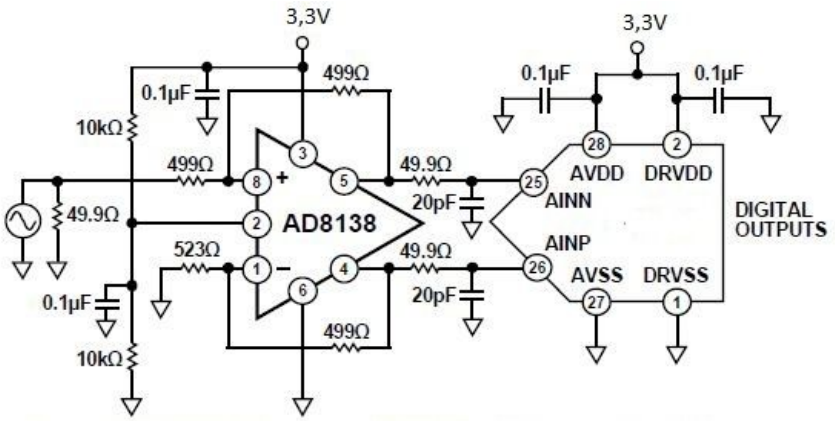


Illustrazione 81: Tipica configurazione dell'AD8138

Il circuito predisposto per configurare correttamente l'integrato AD8138, rispettando le esigenze sopra citate e seguendo le indicazioni dello schema di figura 81, è schematizzato nell'illustrazione 82.

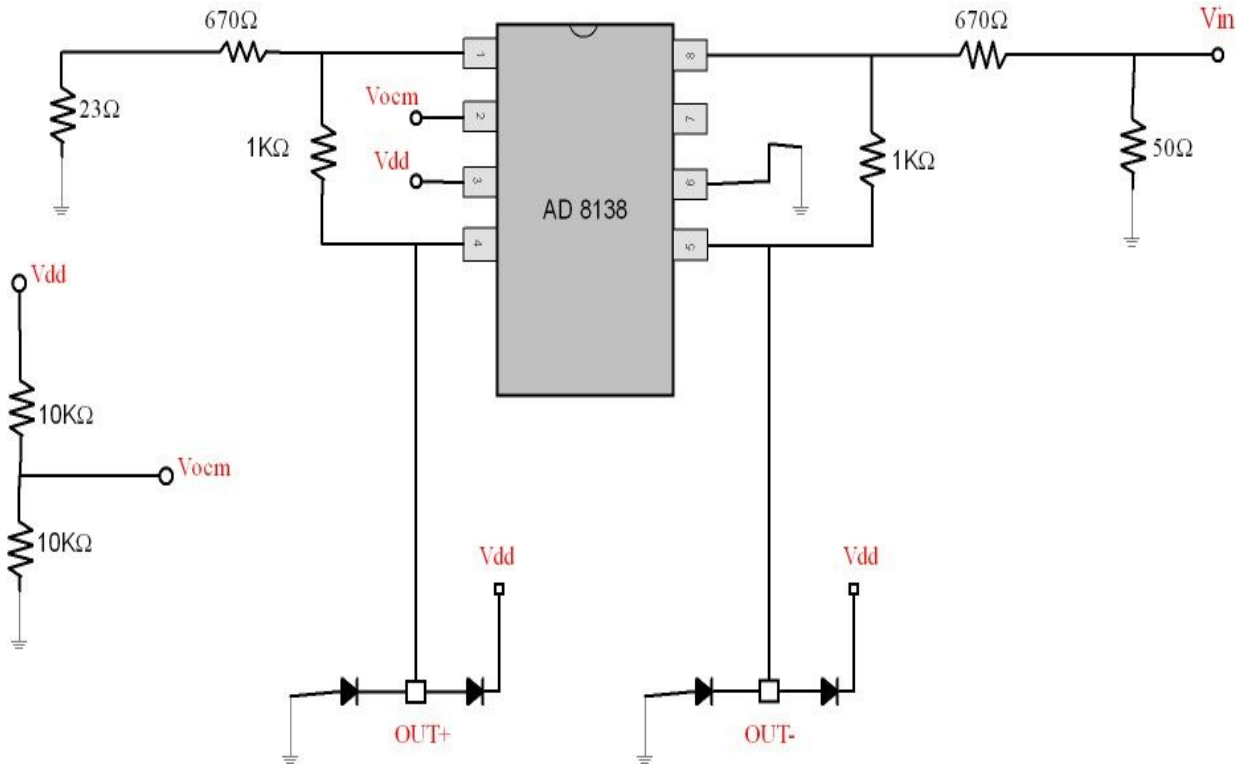


Illustrazione 82: Circuito realizzato con l'integrato AD8138

Il segnale V_{in} rappresenta l'ingresso della nuova scheda realizzata, ed è un segnale bipolare analogico single-ended proporzionale ad una tensione concatenata del motore in alternata.

V_{dd}, l'alimentazione dell'integrato, è ricavata direttamente da un pin della scheda THDB_ADA, ed è pari a 3.3V. La massa del sistema digitale è comune sia a questo nuovo circuito, sia allo stadio di riduzione della scheda analogica. Le uscite dell'AD8138 sono connesse all'ingresso differenziale del convertitore A/D AD9248 (pin 2 e 3), prima di un filtro passa-basso già presente a bordo della THDB_ADA. Nell'illustrazione 83 è messo in evidenza lo schematico dell'ingresso della scheda THDB_ADA, con le modifiche apportate (OUT+ e OUT- sono le uscite dello schema della figura 82):

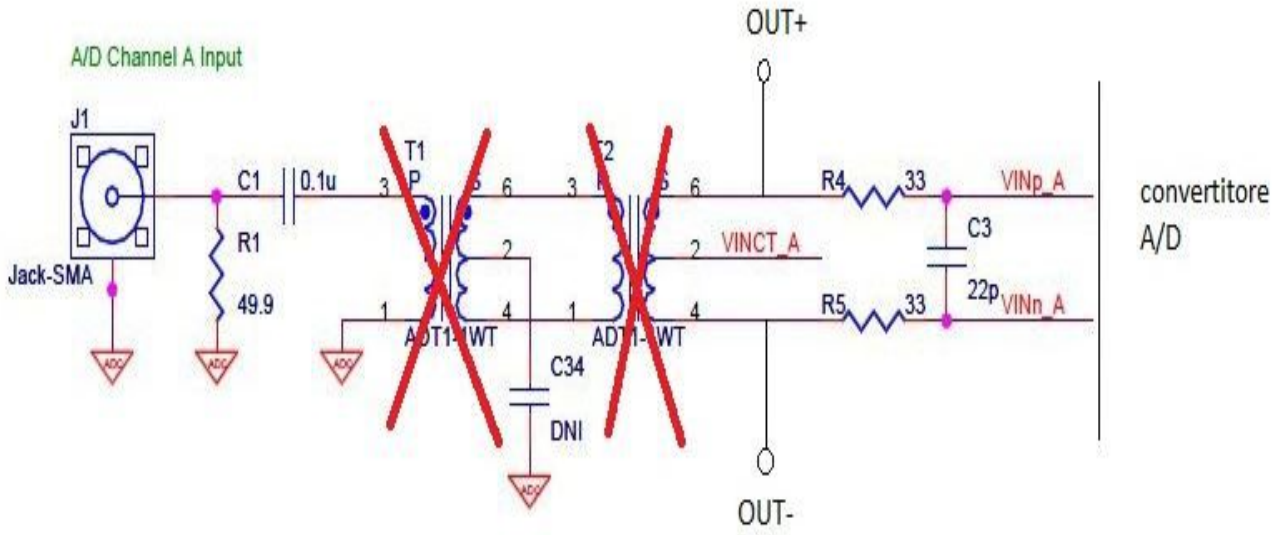


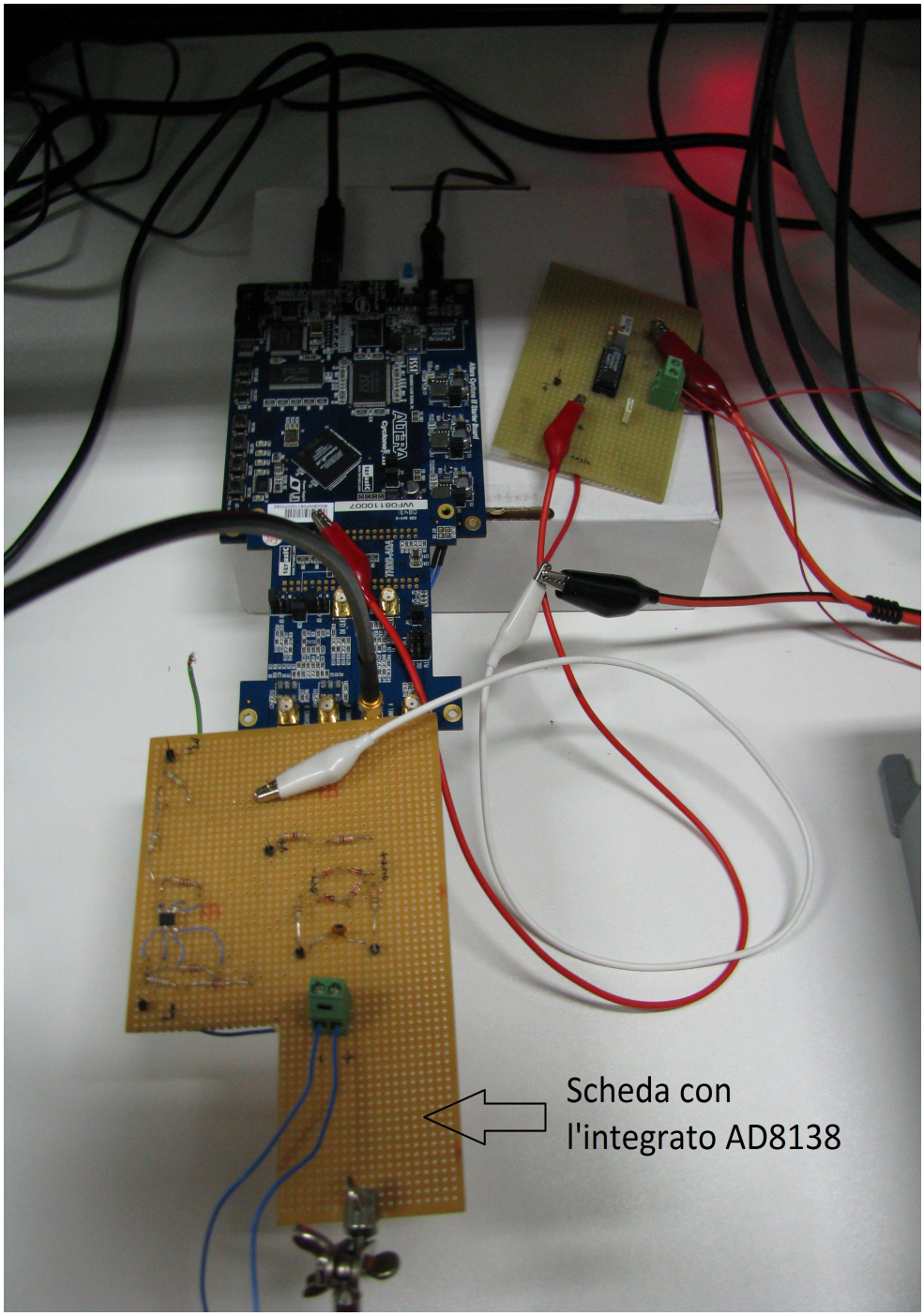
Illustrazione 83: Ingresso della scheda THDB_ADA (in rosso le modifiche effettuate) con OUT+ e OUT- che sono le uscite dell'integrato AD8138

Il guadagno dello stadio di attenuazione è dato dal rapporto delle due resistenze da 1kΩ e 670Ω: 1000/670 = 1,49. Ci sono 2 coppie di resistenze, una per ogni uscita differenziale: i loro rapporti devono essere il più possibile uguali fra loro per il corretto bilanciamento delle uscite digitali.

La resistenza da 23Ω in serie a quella da 1kΩ al pin 1 dell'AD8138 serve a bilanciare il parallelo dell'impedenza d'ingresso a 50Ω, e quella di uscita, sempre di 50Ω, presente all'ingresso del convertitore A/D (50Ω||50Ω = 23Ω).

Nel caso, come quello di questo progetto, in cui la tensione di alimentazione dell'integrato AD81348 e del convertitore A/D sia a 3,3V single-ended, cioè con il pin per l'alimentazione negativa a massa, e in ingresso ci sia un segnale bipolare, cioè sia positivo che negativo rispetto al livello di riferimento di massa, bisogna fornire una tensione di modo comune V_{ocm} al pin 2 dell'AD8138. In questo modo si effettua uno "spostamento" dell'ingresso verso livelli di tensione solamente positivi. Si fornisce quindi una tensione costante, pari a metà della tensione di

alimentazione V_{dd}, attraverso un semplice partitore resistivo al pin 2 dell'integrato. In questo modo l'AD8138 accetta segnali in ingresso bipolari single-ended mantenendo l'accoppiamento DC.
Le uscite OUT+ e OUT- sono poi limitate, attraverso due diodi, tra la tensione di alimentazione V_{dd} e massa per non rischiare di danneggiare gli ingressi dell'integrato che funge da convertitore A/D.
Nella seguente figura si può vedere la scheda realizzata per modificare lo stadio d'ingresso della THDB_ADA e la connessione con la Cyclone III FPGA Starter Board:



Anche lo stadio di uscita dei convertitori D/A della scheda THDB_ADA è stato modificato in base alle esigenze di questo particolare progetto. Come per gli ingressi, è stato tolto il trasformatore RF posto tra il DAC e il connettore BNC, come illustrato in figura 84.

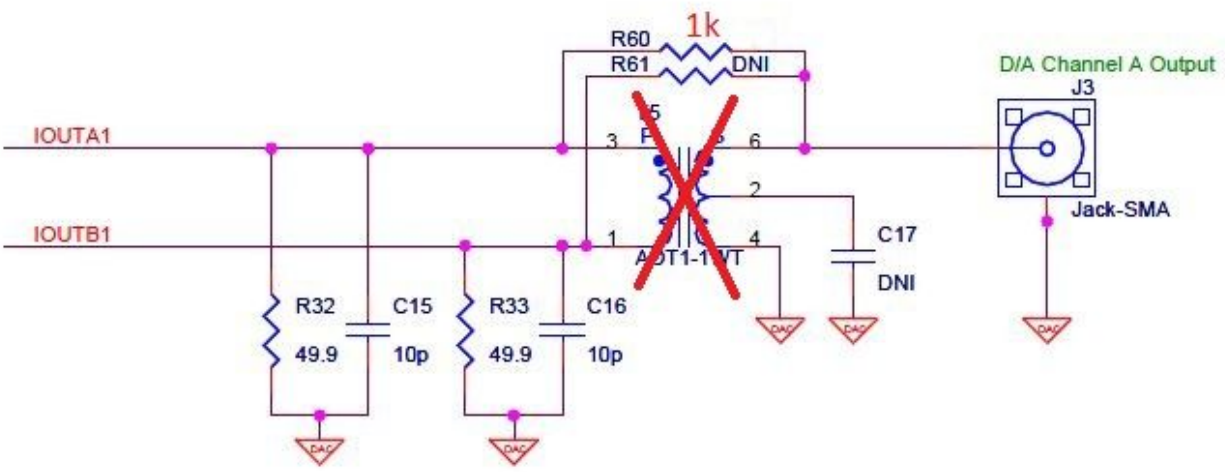


Illustrazione 84: Particolare dello schematico della scheda THDB_ADA con le modifiche apportate (segnate in rosso)

Non è stato necessario in questo caso fare un'altra scheda, in quanto è stata aggiunta solamente una resistenza da 1kΩ al posto di R60 (non montata nello schema originale): in questo modo si può connettere l'oscilloscopio al BNC ed analizzare l'uscita del convertitore D/A convertita da segnale di corrente (da 0 a 20mA) a segnale in tensione (da 0 a 1V) per mezzo della resistenza R32 da 49,9Ω, come mostra lo schema a blocchi dell'illustrazione 85.

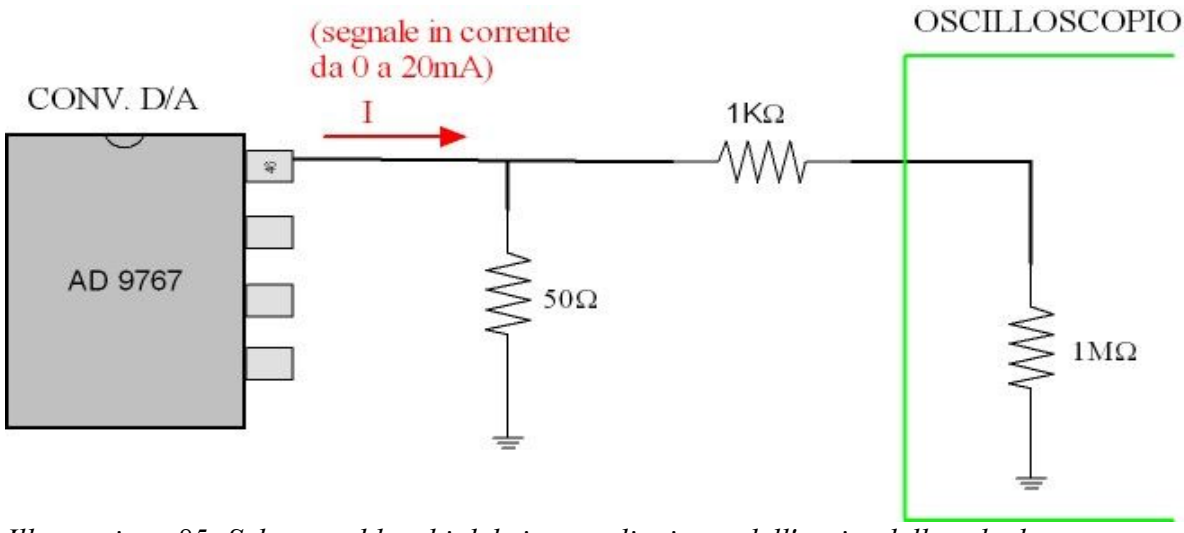


Illustrazione 85: Schema a blocchi del sistema di misura dell'uscita della scheda THDB_ADA

L'ultima modifica hardware alla scheda THDB_ADA, necessaria per connetterla con il resto del sistema, è stata quella di portare a uno dei pin liberi della FPGA (già connesso a un pin del connettore HMSC) il segnale di sincronismo con la PWM dell'azionamento del motore.

4. MISURE E CONFRONTI SUI DUE SISTEMI

Ora che il sistema analogico è stato studiato e analizzato e quello digitale completato, sia dal punto di vista hardware che software, si possono connettere, uno alla volta, all'azionamento e misurare le rispettive uscite, cioè le medie nel periodo di modulazione PWM delle tensioni concatenate del motore sincrono a magneti permanenti isotropo. Sono i motori più usati in tutte le applicazioni che richiedano alte prestazioni dinamiche.

La conversione elettromeccanica che essi attuano si basa sull'interazione fra conduttori percorsi da correnti e campi magnetici creati dai magneti permanenti: i conduttori su cui agiscono le forze sono collocati nella parte fissa (***statore***) mentre i magneti permanenti sono sul ***rotore***. Lo statore e il rotore sono entrambi a forma di corona cilindrica di materiale ferromagnetico laminato e separati da un ***traferro*** in aria. Il termine "isotropo" si riferisce alla disposizione dei magneti permanenti e alla forma del rotore. L'avvolgimento di statore è di tipo trifase; le tre fasi sono reciprocamente sfasate nello spazio di $2\pi/3$, e ciascuna fa capo ad una coppia di morsetti, attraverso i quali è possibile fornire loro alimentazione da una sorgente trifase esterna, cioè l'***inverter***.

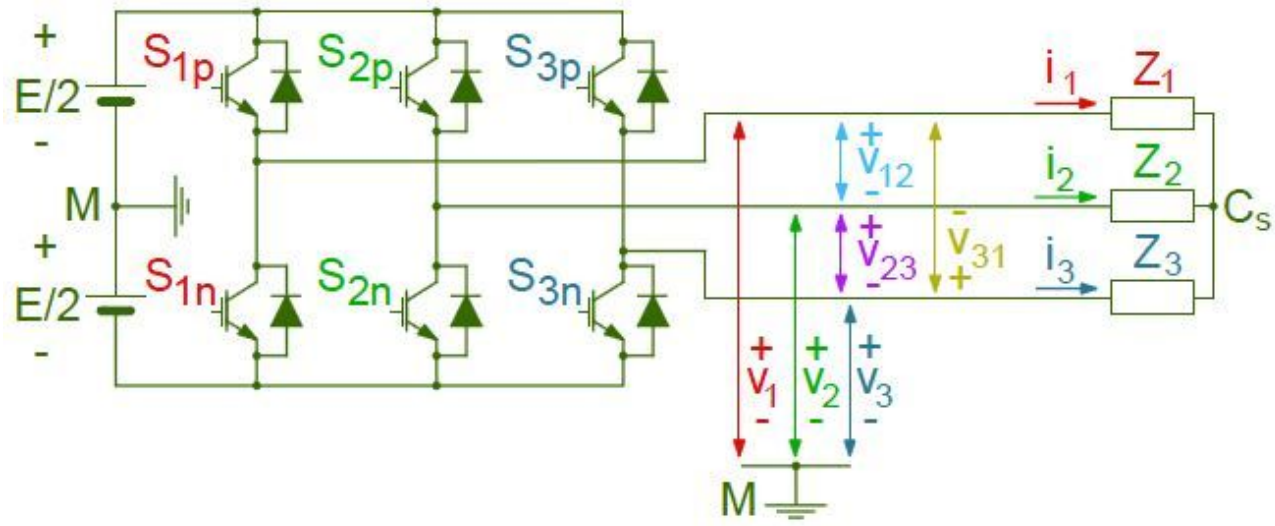


Illustrazione 86: Rappresentazione schematica di un invertitore di tensione trifase

Attraverso gli avvolgimenti di statore si genera un campo magnetico di orientazione e intensità arbitrarie in grado di generare movimento meccanico agendo sui magneti permanenti del rotore. Variando la frequenza di alimentazione varia anche la velocità di rotazione del motore.

Il sistema elettronico che completa l'azionamento elettrico (in questo caso il sistema FCPS e l'inverter) controlla la posizione del motore e usa questa informazione per alimentare le fasi statoriche in modo che il flusso generato sia sempre ortogonale all'asse magnetico rotorico.

Tenendo gli assi magnetici ortogonali (in “quadratura”), la coppia è sempre massima e dipende solo dalla corrente statorica (in modo proporzionale ad essa).

Le tre tensioni di fase che alimentano il motore dovrebbero essere delle sinusoidi di uguale ampiezza e sfasate tra loro di 120° (=simmetriche). Essendo però generate dall'inverter, sono delle onde quadre con un elevato contenuto armonico, la cui componente fondamentale è la sinusoide corrispondente alla frequenza necessaria per avere una determinata velocità del motore o raggiungere la posizione voluta. Le fasi del motore, rappresentando un carico ohmico-induttivo, filtrano le componenti ad alta frequenza, con il risultato di avere correnti quasi sinusoidali anche a fronte di alimentazione PWM.

Se le tensioni di fase sono simmetriche, risulta che anche le correnti di fase hanno ampiezze uguali e sono sfasate tra loro di angoli uguali (=equilibrate).

Le tensioni concatenate ricavate da quelle di fase assumono le stesse caratteristiche nella forma d'onda e frequenza, come illustrato in figura 87 e 88.

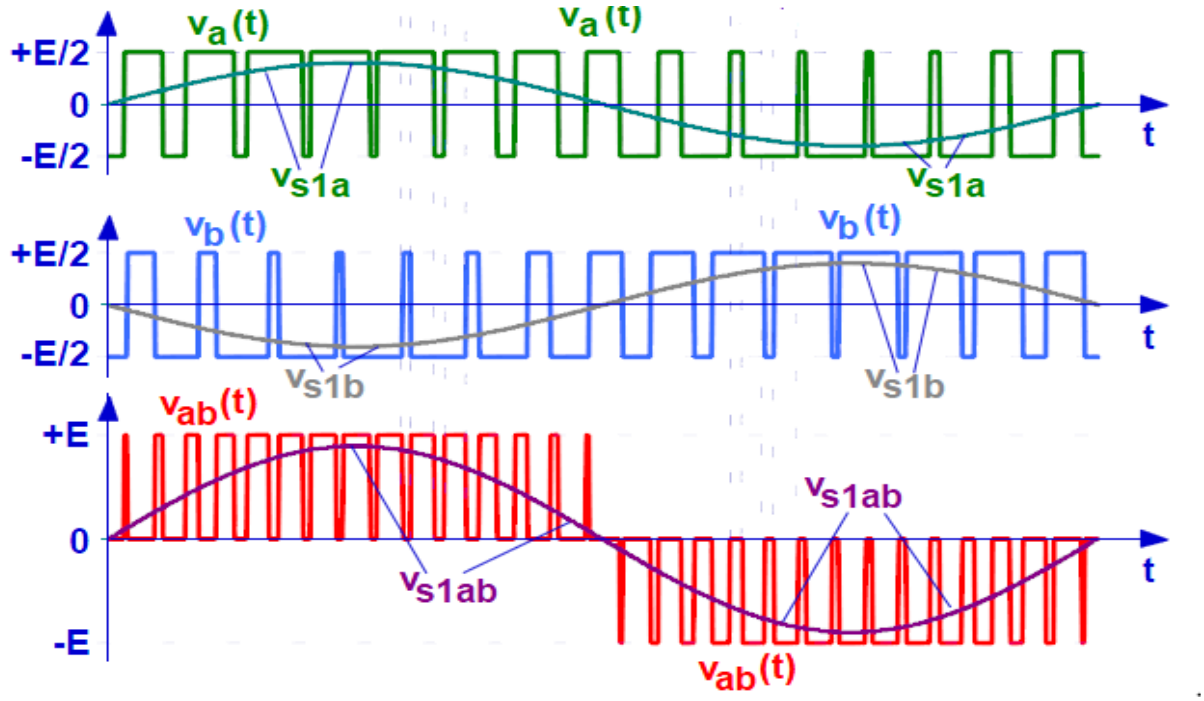


Illustrazione 87: BLU e VERDE= tensioni di fase del motore ROSSO= tensione concatenata del motore. In evidenza, su ogni onda quadra,la componente fondamentale

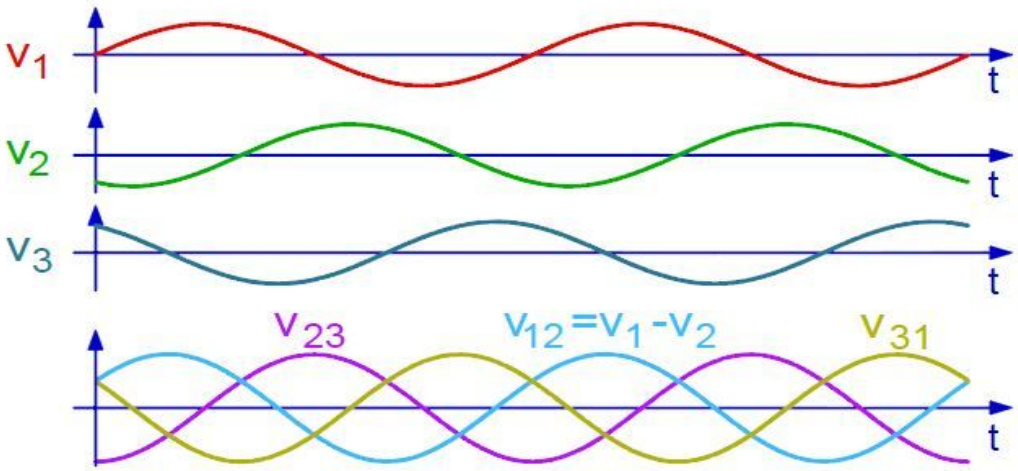


Illustrazione 88: Tensioni concatenate del motoreV_{23}, V_{12}, V_{13} ricavate da quelle di fase

In base a queste considerazioni, connettendo il sistema di misura analogico o quello digitale alle 3 tensioni di fase del motore, si ricavano le forme d'onda differenziali v_{12} e v_{23} illustrate nella figura 88.

Il sistema digitale è stato costruito, in fase di prototipo, per la misura di una sola delle tensioni concatenate del motore.

Attraverso un oscilloscopio sono state effettuate le misurazioni (riportate nelle figure 89,90,91) sull'uscita di un doppio canale di integrazione della scheda analogica e su un connettore BNC di uscita della scheda THDB_ADA per il sistema digitale:

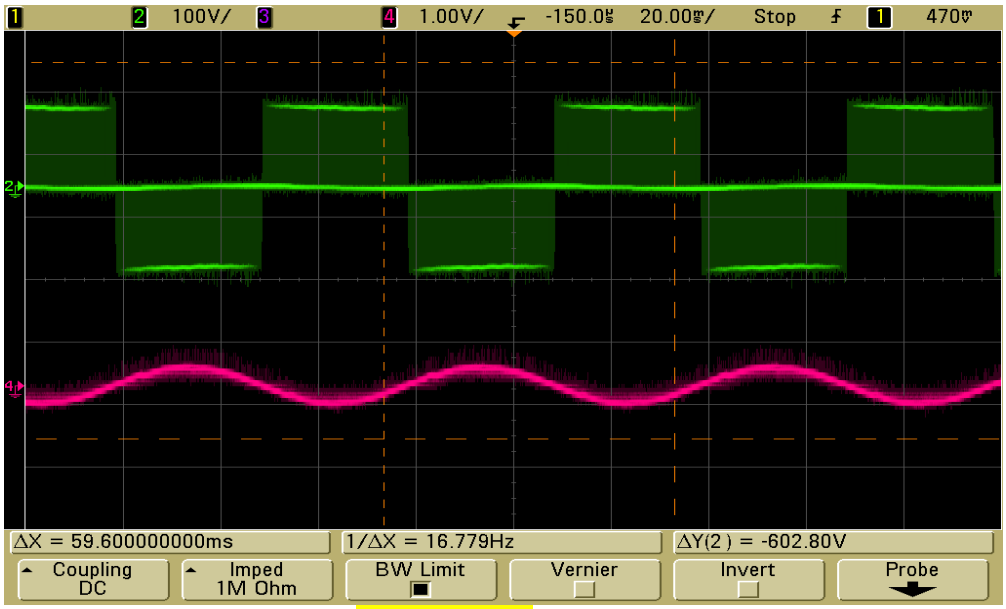


Illustrazione 89: SCHEDA ANALOGICA: VERDE= tensione concatenata del motore ROSA= componente fondamentale della concatenata [1]

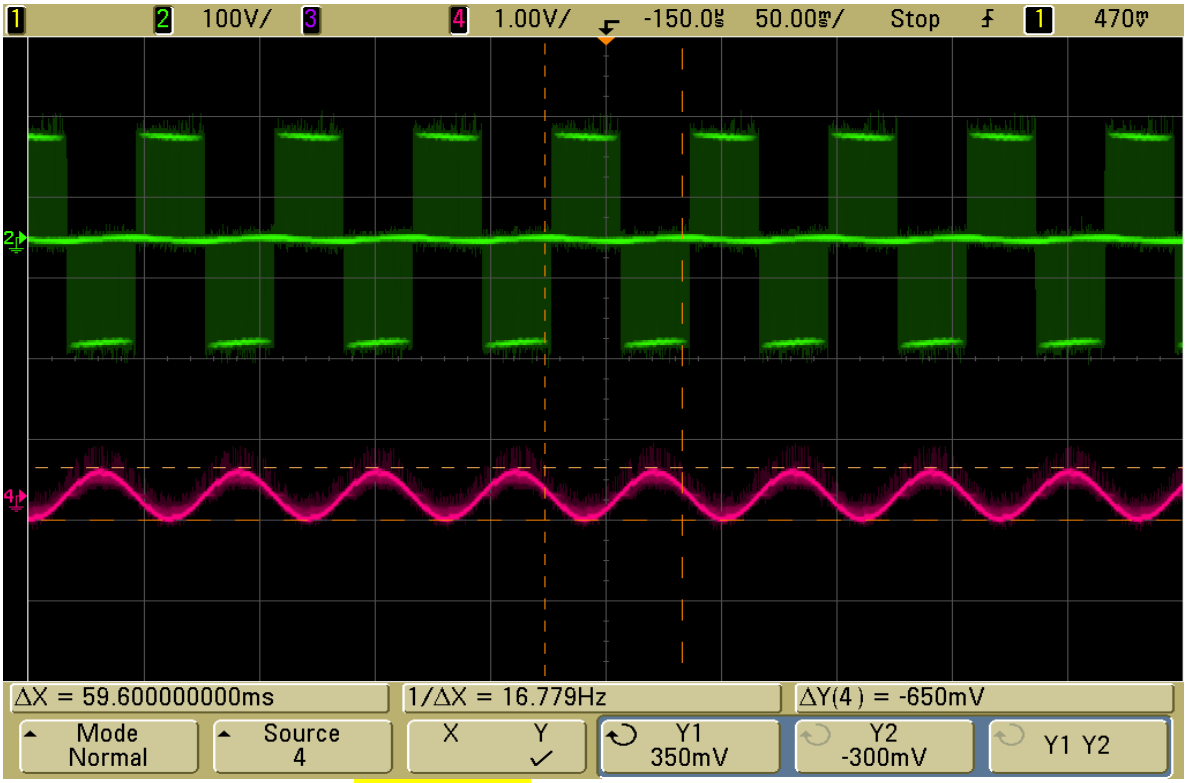


Illustrazione 90: SCHEDA ANALOGICA: VERDE= tensione concatenata del motore ROSA= componente fondamentale della concatenata [2]

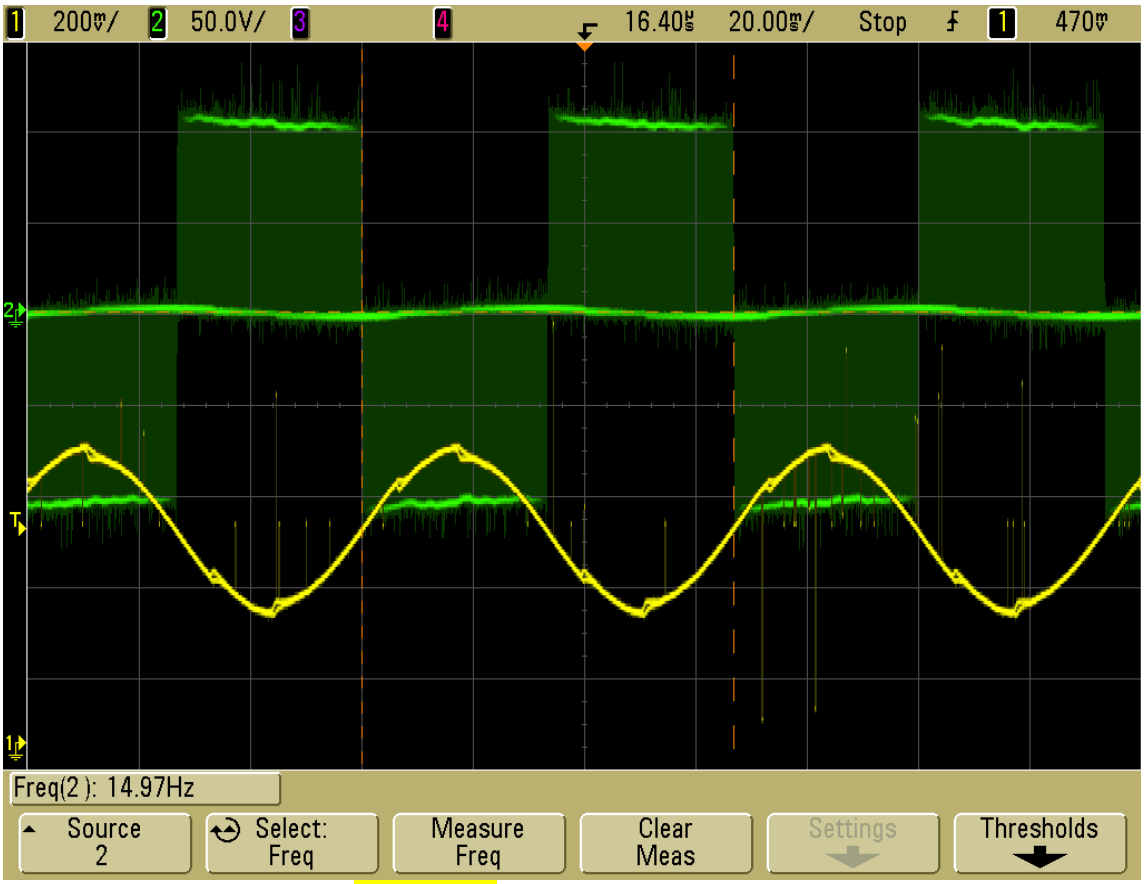


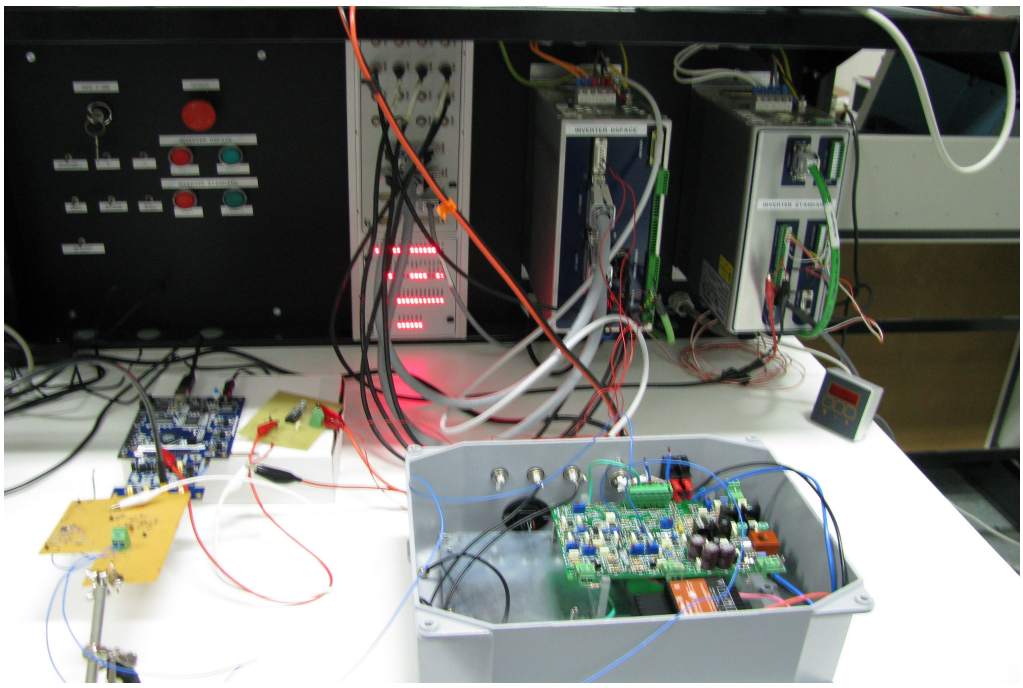
Illustrazione 91: SCHEDA DIGITALE: VERDE= tensione concatenata del motore GIALLA= componente fondamentale della concatenata

Le figure 89 e 90 illustrano una tensione concatenata del motore (ricavata da 2 tensioni di fase in uscita dall'inverter con una sonda differenziale) e la sua componente fondamentale elaborata dalla scheda analogica di misura. La forma d'onda è esattamente quella che ci si aspettava, con i fattori di attenuazione descritti nel capitolo 2.

Anche le misure effettuate con il sistema digitale, illustrate in figura 78, rispettano le considerazioni fatte nel capitolo 3.

In evidenza ci sono le frequenze delle sinusoidi, cioè il valore della componente fondamentale corrispondente alla frequenza necessaria per avere una determinata velocità del motore, impostata dall'interfaccia grafica realizzata per controllare l'azionamento tramite il sistema FCPS.

Nella figura seguente è possibile vedere la connessione dell'azionamento elettrico con i 2 sistemi di misura:



5. CONSIDERAZIONI FINALI E ULTERIORI POSSIBILITA' DI SVILUPPO DEL PROGETTO

L'obiettivo principale dell'intero progetto può ritenersi raggiunto: è stato possibile calcolare le componenti fondamentali delle tensioni concatenate di un motore in corrente alternata attraverso l'utilizzo di un dispositivo logico digitale, l'FPGA, e i risultati ottenuti sono confrontabili con quelli perseguiti attraverso la soluzione analogica [capitolo 4].

Ovviamente ci sono moltissime altre possibilità di sviluppare il lavoro svolto e di migliorare quello che è stato compiuto, approfondendo tutti quegli aspetti che fino a questo punto non sono stati presi in considerazione in maniera approfondita.

L'ottimizzazione di un sistema digitale comporta normalmente un bilancio fra un certo numero di parametri che caratterizzano il circuito. Parametri tipici sono l'***area*** occupata dal circuito (di solito valutata riferendosi al numero di porte logiche utilizzate), il ***ritardo di propagazione***, valutato fra l'istante in cui si presenta un evento, cioè la variazione di un ingresso, e l'istante in cui la risposta all'evento appare manifesta in uscita, la ***potenza dissipata*** dal circuito. Ognuno di questi aspetti può essere rivisto e migliorato per questo specifico progetto.

Si dovrà modificare il software elaborato per programmare l'FPGA per eliminare i ***glitch*** che si presentano periodicamente in uscita ai convertitori D/A nel risultato dell'elaborazione digitale delle tensioni concatenate. Questi sono molto probabilmente dovuti alla sincronizzazione non perfetta fra il latch in uscita e l'accumulatore per ogni canale di integrazione, che sono temporizzati dallo stesso segnale d'ingresso, il SYNC.

Nelle illustrazioni 92, 93, 94 sono riportate alcune immagini ottenute con l'oscilloscopio che evidenziano l'errore nel risultato dell'elaborazione digitale (VERDE=tensione concatenata del motore, GIALLO=media mobile della tensione concatenata, ROSA=SYNC).

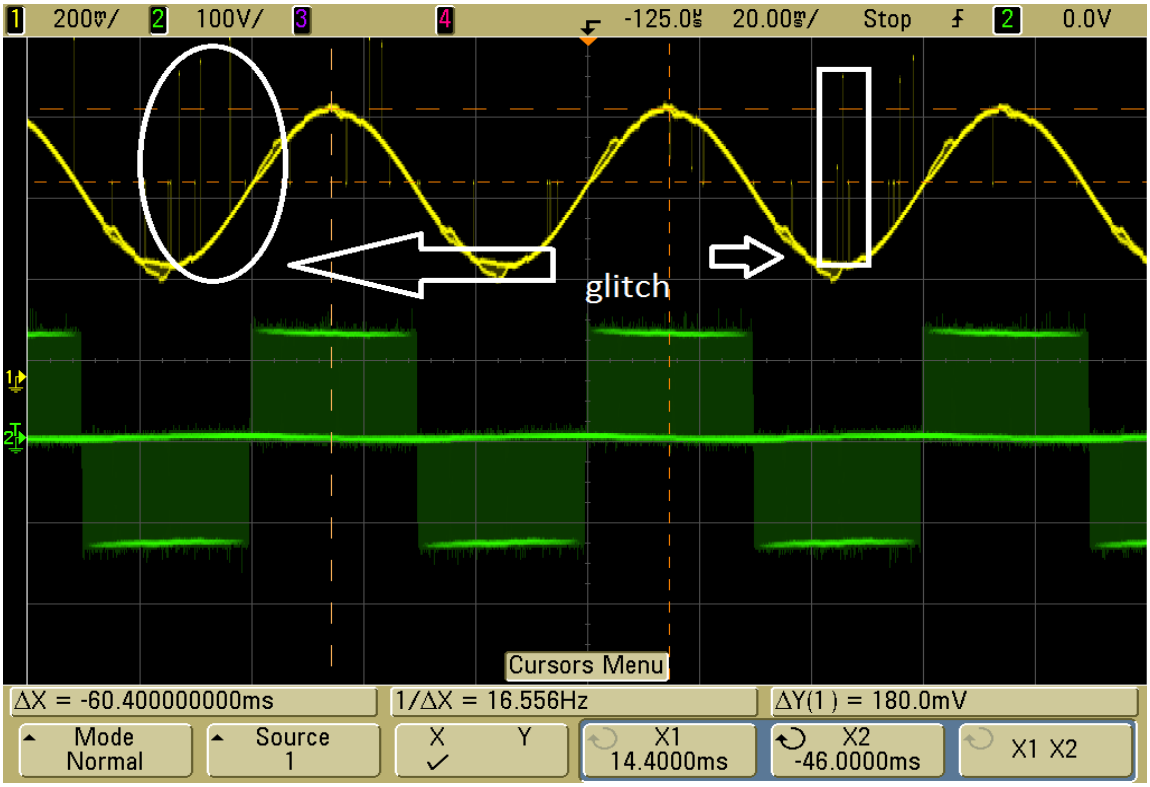


Illustrazione 92: Media della concatenata, in evidenza i GLITCH

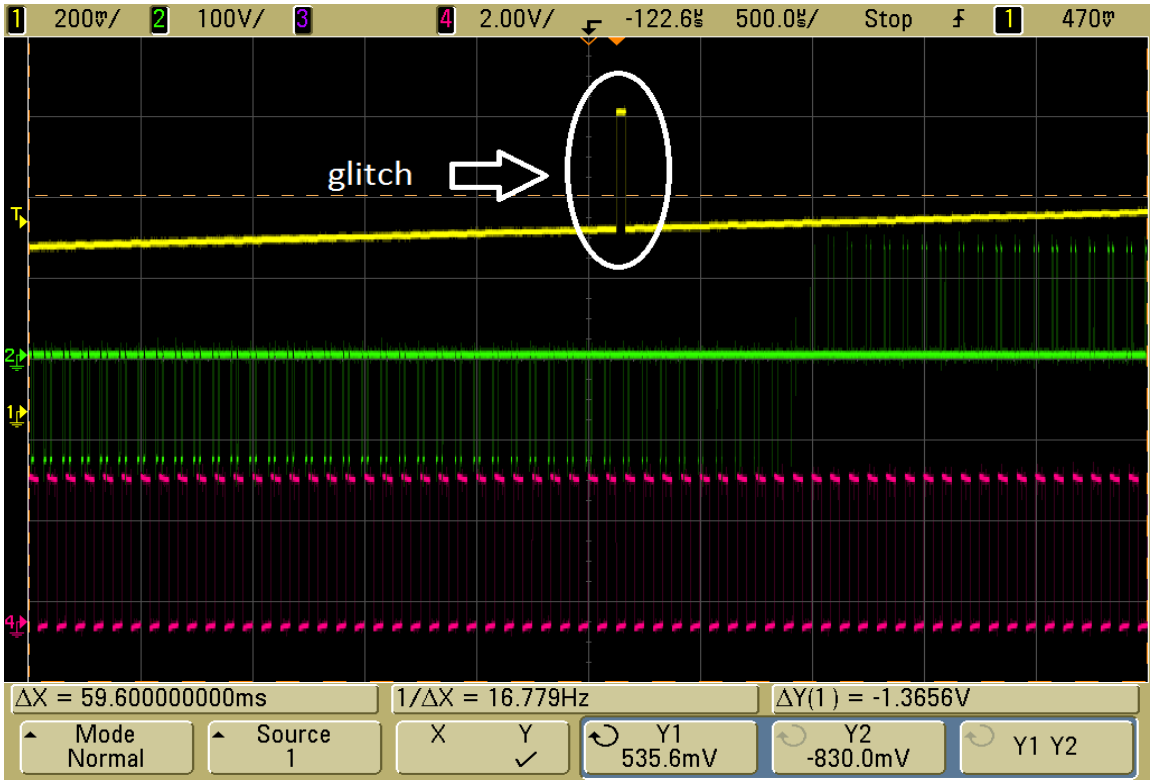


Illustrazione 93: Particolare di un glitch in uscita [1]

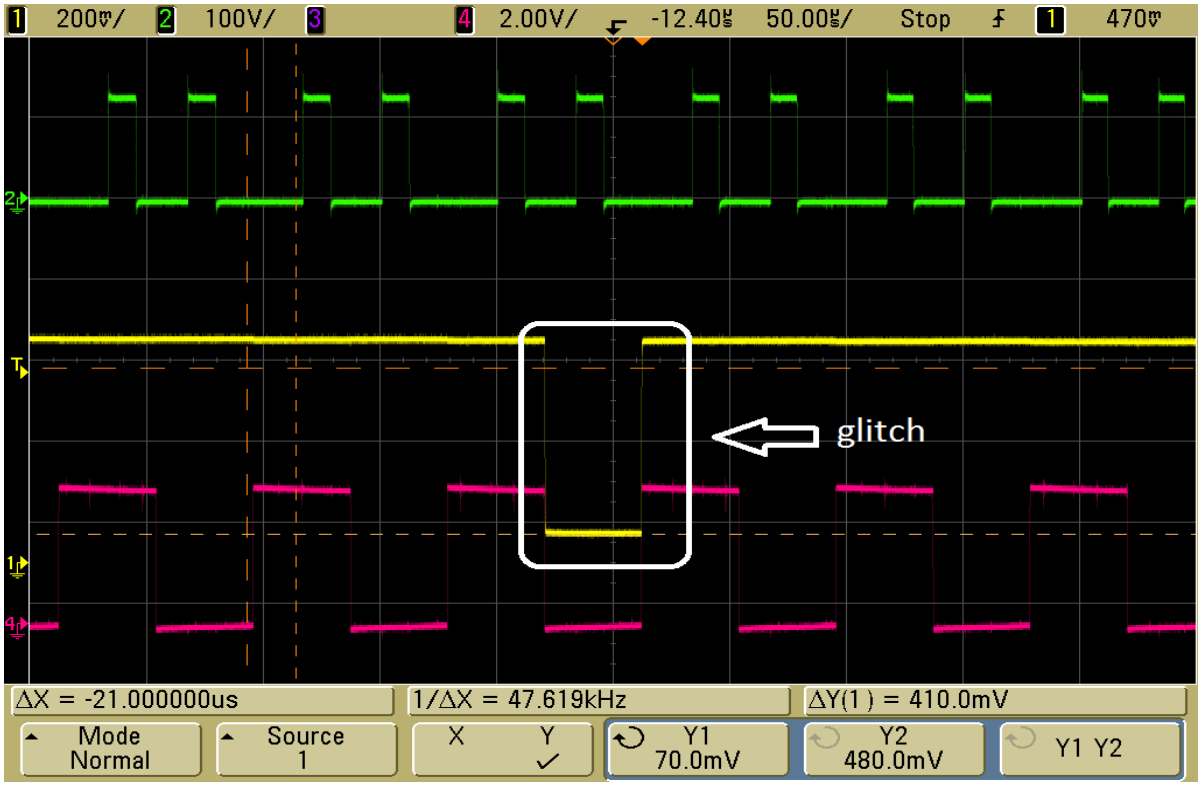


Illustrazione 94: Particolare di un glitch in uscita [2]

Per sopprimere questi glitch le soluzioni sono molteplici:

- effettuare l'integrazione del segnale campionato su un solo canale, eliminando così la sincronizzazione tra i due canali che lavorano alternativamente.
- riformulare interamente l'algoritmo realizzandolo attraverso una MACCHINA A STATI FINITI, ottimizzata da punto di vista dei tempi di propagazione dei segnali attraverso le porte logiche dell'FPGA.
- migliorare le funzionalità temporali del progetto, attraverso il tool messo a disposizione dal **Quartus II TimeQuest Analyzer**.

Per analizzare e modificare gli aspetti temporali di un progetto il Quartus II mette a disposizione il tool denominato "**TimeQuest Analyzer**". Con esso si può ottimizzare la propagazione dei segnali digitali attraverso i singoli blocchi logici della FPGA e sincronizzarli con un segnale di clock fissato per eliminare effetti indesiderati sulle uscite del sistema, come i glitch descritti sopra, dovuti ai diversi ritardi di propagazione dei segnali.

Alternativamente si può riformulare interamente l'algoritmo e realizzarlo attraverso una macchina a stati finiti, gestendo tutti gli aspetti temporali mediante gli algoritmi tipici dell'elaborazione dei

segnali nell'elettronica digitale, come l'algoritmo di Quine-McCluskey, mappe di Karnaugh, minimizzazione mediante metodo analitico.

La scheda con l'integrato AD8138 per l'acquisizione di una tensione concatenata in accoppiamento DC è realizzata su una scheda di prototipizzazione (= millefori). Dovrà essere necessariamente rifatta con 2 integrati per acquisire entrambe le tensioni concatenate che arrivano dal motore: è stato previsto di costruire un circuito stampato (PCB) con componenti SMD. La nuova scheda realizzata viene poi posta nel contenitore metallico con la scheda analogica di misura e la Cyclone II FPGA Starter Board. Quindi anche il contenitore dovrà essere rivisto, in quanto c'è l'esigenza che esso abbia dimensioni maggiori e con nuovi connettori viste le nuove schede al suo interno.

Una volta definito il sistema digitale di misura, si dovrà configurare l'algoritmo per la stima di coppia nel sistema di controllo che utilizza le medie delle tensioni concatenate del motore ad ogni periodo di modulazione PWM. Il codice verrà realizzato all'interno dell'algoritmo di controllo a catena chiusa dell'azionamento. In questo modo si ricava una stima della coppia istantanea del motore utilizzata nell'anello a catena chiusa di controllo, risparmiando così l'utilizzo del sensore esterno.

Bibliografia:

[1] ***Silverio Bolognani***

Dispense del corso di "Azionamenti elettrici 1" tenuto all'università di Padova nell'anno accademico 2007/2008

[2] ***Silverio Bolognani***

Dispense del corso di "Azionamenti elettrici 2" tenuto all'università di Padova nell'anno accademico 2007/2008

[3] ***Luigi Malesani***

Dispense del corso di "Elettronica per l'energia" tenuto all'università di Padova nell'anno accademico 2008/2009

[4] **N. *Mohan, T. Undeland, W. Robbins***

"Elettronica di potenza"

HOEPLI Editore

[5] ***R.C.Jaeger, T. N. Blalock***

"1 Microelettronica – Elettronica Analogica"

2° edizione McGraw-Hill 2005

[6] Manuale di riferimento per il sistema di sviluppo Quartus II

[7] Data-sheet dei dispositivi elettronici utilizzati nel progetto

[8] Manuali di utilizzo per il linguaggio Verilog

[9] ***F. Fummi, M. Sami, C. Silvano***

"Progettazione digitale"

McGraw-Hill

[10] Articolo scientifico di ***T-H. Chin, M. Nakano, T. Hirayama***

"Accurate measurement of instantaneous voltage
for power electronics circuits"

{Proceedings of Power Conversion Conference, {PCC}'97}

[11] ***Mark zwolinski***

"VHDL. Progetto di sistemi digitali"

Pearson – Prentice Hall

[12] ***Charles R. Kime***

"Reti logiche"

Pearson - Addison Wesley