UNIVERSITA' DEGLI STUDI DI ROMA

TOR VERGATA

Facoltà di Ingegneria

Dipartimento di Elettronica

XXII ciclo di dottorato

Tesi di dottorato di ricerca

STRUMENTAZIONE VIRTUALE PER LA MISURA DI GRANDEZZE ELETTRICHE E CALCOLO DELL'INCERTEZZA

Tutor:

Prof. ***Roberto Lojacono***

Prof. ***Marcello Salmeri***

Dottorando:

Dott. Ing. Alfredo Accattatis

Sommario

Indice delle figure

RINGRAZIAMENTI

Giunto alla fine di questa incredibile esperienza, talmente entusiasta da sperare di riuscire a ricominciare di nuovo, ringrazio il prof. Roberto Lojacono e l'amico prof. Marcello Salmeri, grazie ai quali ho potuto far parte, anche se per un tempo troppo breve, del mondo accademico; e parimenti tutti coloro i quali in questo periodo hanno dovuto sopportare il mio entusiasmo, consentendomi di crescere e innamorarmi ancora una volta dell'universo "ricerca". In ultimo, ma non in ordine d'importanza, devo ringraziare la rivista Nuova Elettronica ed in particolare Giacomo Barra e Tiziano Vecchi, che hanno creduto nel mio lavoro e nell'Università, dando vita ad una proficua e duratura collaborazione.

Capitolo 1

Strumentazione Virtuale

1.1 Introduzione

Il termine "Strumentazione Virtuale" è inteso nell'accezione di "riproduzione software" di strumenti di misura normalmente realizzati quasi completamente in hardware. Lo strumento virtuale non è tuttavia completamente software: un componente hardware dedicato acquisisce, converte e memorizza nella RAM dell'elaboratore i campioni rappresentativi del segnale da misurare, ed un programma di calcolo si occupa di implementare via software tutto il resto. In tal modo si ottiene un funzionamento del tutto equivalente ad uno strumento hardware "stand-alone". Questi ultimi nascono come strumenti adatti a misurare una o più grandezze (es.: multimetro, fonometro, luxmetro, etc.) e durante tutto il loro ciclo di vita effettuano solo una determinata classe di misure; e non possono essere modificati se non entro limiti determinati dalla loro struttura fisica (es. aggiunta di sonde attenuate ad un oscilloscopio, circuiti elevatori d'impedenza per multimetri, divisori per un frequenzimetro, etc.). Dunque, per definizione, uno strumento virtuale consente di "sostituire" dell'hardware con del software, ossia il programma di calcolo effettua elaborazioni che negli strumenti "reali" vengono eseguite da circuiti fisici. Una sottocategoria degli strumenti virtuali, qui non considerata, è quella degli strumenti "completamente sintetici", intendendo con tale termine una categoria di strumenti di misura e generazione che non abbisognano di hardware specifici ma usano solamente

un hardware “generico”. Per esempio un personal computer o comunque un hardware di uso generale (“general-purpose”) che non si avvale di organi dedicati, ossia specializzati per una funzione determinata. Si osservi come uno “strumento sintetico” in quanto sottocategoria della più ampia classe “strumenti virtuali” è comunque uno strumento virtuale. L’uso degli strumenti sintetici è nella pratica ristretta ad applicazioni in campo musicale, ed in particolare nell’ambito della sintesi sonora.

Figura 1: esempio di strumento virtuale

Lo strumento virtuale dunque implementa lo strumento di misura a mezzo di un programma di calcolo, una struttura elaborativa generica e hardware dedicato per l’acquisizione e condizionamento dei segnali: una architettura tipica può essere per esempio una piattaforma composta da uno o più DSP (Digital Signal Processor) un microcontrollore, RAM condivisa e convertitori A/D e D/A (alle volte “embedded” nel microcontrollore stesso) più amplificatori, circuiti di condizionamento e protezione (es. diodi) e una interfaccia di comunicazione e trasferimento dati con l’elaboratore (USB, RS232, bus PCI, GPIB, etc.). Sovente per questo tipo di piattaforme sono utilizzati particolari sistemi operativi real-time e multitasking (es.

VxWorks, OSE). Il software preleva dalla memoria i campioni del segnale, ed effettua tutte quelle operazioni necessarie al calcolo della misura (spesso indiretta), visualizzando il risultato a video tramite interfaccia grafica, display alfanumerico od anche tramite diodi led o "barre" degli stessi. In quanto strumento costituito essenzialmente da software, presenta dunque tutti i vantaggi del caso: aggiornamenti, cambi di funzione, miglioramenti, estensioni, modifica dell'interfaccia, vengono tutti realizzati tramite semplici sostituzioni del software (che in questi casi e più corretto definire firmware). Inoltre, l'utilizzo sinergico di sistemi operativi sofisticati e linguaggi di alto livello, consente di sfruttare in maniera capillare le risorse elaborative tramite l'uso intensivo del multitasking e del multi-threading permettendo l'esecuzione simultanea o concorrente di più strumenti (concorrente su macchine con numero di processori inferiore ai task attivi).

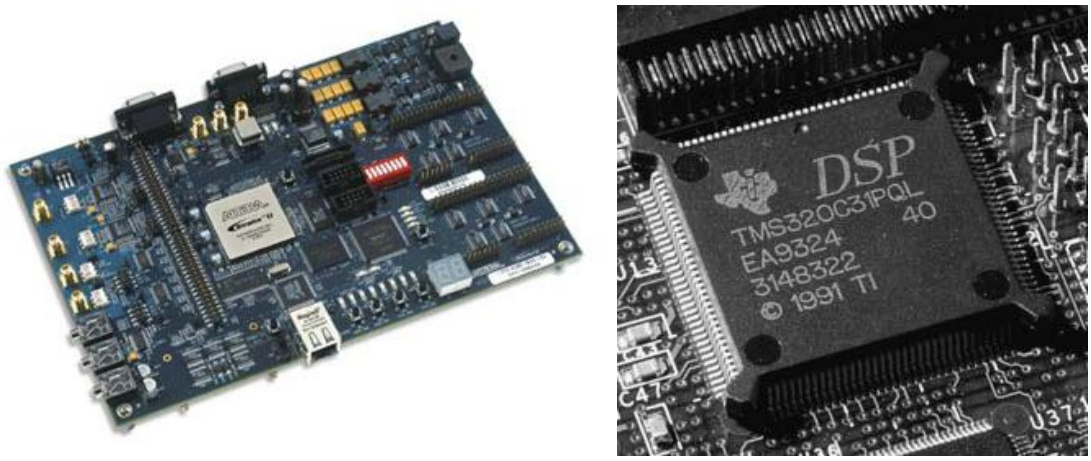


Figura 2: Digital Signal Processor (DSP)

I sistemi operativi che supportano il multitasking sono in generale disponibili per sistemi microprocessore, ma già da alcuni anni sono stati sviluppati sistemi operativi che supportano direttamente piattaforme a DSP multipli (ossia composte solo da DSP), come per esempio OSE di Enea, utilizzato sia in ambito industriale che accademico.

Si deve a questo punto osservare che la tendenza attuale dei costruttori di strumenti di misura (es. Hameg, National, Tektronik, Le Croy, etc.) è quella di produrre

dispositivi costituiti da un vero e proprio Personal Computer “standard” (PC da qui in avanti) con annesso un hardware di acquisizione dati; inoltre sovente viene direttamente utilizzato un Sistema Operativo (SO da qui in avanti) commerciale o freeware tipo Windows o Linux, più che un SO proprietario o specializzato, come i già citati VxWorks ed OSE (e molto tempo fa DOS real-time).

Figura 3: strumentazione virtuale con PC

La continua evoluzione hardware dei PC ha reso infatti possibile l’utilizzo degli stessi in applicazioni un tempo appannaggio esclusivo dei DSP, e dunque rendere parimenti conveniente l’utilizzo di sistemi operativi canonici e diffusissimi alla stregua di quelli di casa Microsoft o di alternative gratuite come Linux. Con l’ulteriore vantaggio di poter implementare strumenti con interfaccia utente nota alla maggior parte degli utenti, con una consequenziale e consistente riduzione dei costi; il tempo necessario per essere immediatamente operativi è sicuramente un fattore non trascurabile a livello industriale e professionale in genere.

Appare chiaro dunque che il termine strumentazione virtuale potrebbe perdere di significato, in quanto le tendenze attuali mostrano come gli strumenti sono “nativamente” realizzati come un programma di calcolo in esecuzione su di una piattaforma hardware/software generica. Anche, si potrebbe dunque affermare che gli strumenti virtuali stanno gradualmente sostituendo gli strumenti classici.

Il lavoro presentato in questa tesi è un software che “virtualizza”, nell’accezione classica del termine, un “set” di strumenti di misura Elettronici. Od in altri termini, a mio avviso più calzanti, esso “realizza” una serie di strumenti di misura. Esso utilizza un personal computer con sistema operativo Windows o Linux, ed un hardware di acquisizione capace di trasformare i segnali elettrici analogici in sequenze di campioni digitalizzati e memorizzati in RAM (tramite meccanismi

hardware/software mediati da bus USB e/o interni al PC). L'hardware di acquisizione può essere dedicato (cfr. cap. IX) oppure è possibile utilizzare persino la stessa scheda sonora (normalmente presente in ogni personal computer). Il secondo caso è stato contemplato per fini didattici e di ricerca, mantenendo una accettabile accuratezza delle misure e con il non trascurabile vantaggio di un costo praticamente irrisorio.

La scheda sonora è infatti un completissimo hardware per l'acquisizione e la sintesi dei segnali elettrici; con un modesto hardware aggiuntivo è possibile usarla alla stregua di una scheda di acquisizione dati a due canali (le schede sonore sono quasi sempre stereofoniche). Le schede sonore più recenti sono dotate di convertitori A/D e D/A che spesso si spingono sino a 24 bit di risoluzione e con frequenze di campionamento che possono arrivare a 192 kHz, ossia a bande passanti dell'ordine di 96 kHz. Quest'ultimo dato non tragga in inganno, perché moltissimi strumenti possono realizzarsi con bande passanti apparentemente così ridotte, che nella realtà permettono un numero consistente di misure anche in ambiti professionali. Di converso, una scheda sonora presenta una limitazione non trascurabile dovuta alla sua funzione primigenia; dovendo infatti essere utilizzata in ambiti musicali e non di misure elettroniche, essa non è opportunamente "caratterizzata metrologicamente" dal costruttore (cfr cap. II, XIII), per il quale ovviamente una simile caratteristica non è di alcuna utilità. Caratterizzazione metrologica significa (anche) specificare tutta una serie di parametri relativi all'hardware, consentendo così ad un eventuale strumento di misura (virtuale e non) di poter calcolare esattamente l'incertezza associata alle misure effettuate nelle varie portate e tipologie di misure. In altri termini, tutti quei parametri di incertezza di "tipo B" (cfr. cap. II) che solo il costruttore, o chi per lui, può fornire. Infatti, essi vengono determinati con procedure complesse che implicano (anche) una analisi statistica da effettuare su un campione statisticamente rappresentativo di schede di stesso tipo, in modo da poter garantire che qualsiasi scheda prodotta fornisca grandezze il cui valore ricada in un intervallo di indeterminazione noto, e di cui si può prendere a modello una opportuna "distribuzione" di probabilità. E' evidente che questo implica uno sforzo economico non giustificato per i costruttori di schede destinate ad uso prettamente musicale e/o

amatoriale. Per contro è da notare come si stia diffondendo in alcuni ambiti - nonostante le considerazioni fatte - l'utilizzo di schede sonore esterne di alta qualità (connesse al PC tramite bus USB), come hardware di acquisizione per strumenti di misura virtuali anche piuttosto sofisticati, ma non ci risulta che per essi ci sia stato alcuno sforzo per ottenere una valida e comprovata dichiarazione dell'incertezza associata.

Il programma presentato in questo lavoro non pretende di essere uno strumento di misura professionale con il calcolo assolutamente esatto e corretto dell'incertezza, campo vastissimo e complesso, di cui spesso non si comprende a fondo l'importanza. Pur tuttavia è uno dei pochi software di questo genere che comprende un calcolo efficace ed in tempo reale dell'incertezza di tipo A e che da la possibilità di tenere di conto della maggior parte dei parametri di tipo B dichiarati dal costruttore delle schede di acquisizione o altri individuati appositamente e ritenuti pertinenti; in tal senso è possibile (per gli strumenti per cui è stata prevista tale funzione, es. voltmetro, ZRLC) calcolare l'incertezza di misura tenendo conto di tutti i parametri conosciuti (nel senso di cui siamo a conoscenza stante alle conoscenze scientifiche attuali) e dunque, tramite scelte opportune e prudenziali (fattori di copertura), essere almeno sicuri di calcolare una incertezza estesa ragionevolmente realistica. Inoltre, per alcuni strumenti è stato implementato, come accenneremo nelle prossime righe, un metodo efficace ed automatico per la riduzione degli "errori sistematici" non identificabili (ovviamente) tramite semplice analisi statistica e/o non dichiarati dal costruttore.

La tesi citata in [1] costituisce la descrizione del software che abbiamo utilizzato come punto di partenza, la cui introduzione originale è riportata in appendice I, con qualche semplice adattamento; il presente lavoro riguarda lo sviluppo di un prodotto software completamente nuovo, che pur partendo da [1] è stato profondamente rivisto nelle sue funzionalità e numero di strumenti realizzati, oltre che nella qualità degli stessi; per alcuni strumenti è stato introdotta la valutazione dell'incertezza, come accennato, effettuata seguendo le linee guida della GUM [22], sia applicando la legge di propagazione delle incertezze che tramite simulazione Monte Carlo (cfr.

cap. XIII, IX), e/o con calcolo in tempo reale dell'incertezza di tipo A (cd. "running statistics"). E stato sviluppato uno strumento per la misura completa delle impedenze, in concomitanza ad un semplice ed economicissimo hardware aggiuntivo, che verrà presentato nel cap. IX. Per esso strumento (battezzato ZRLC) è stato sviluppato un metodo innovativo per la riduzione degli errori sistematici più significativi ed implementato il calcolo in tempo reale dell'incertezza complessiva (A e B, cfr. cap. II e capoverso precedente) che tiene conto della maggior parte dei contributi noti dovuti sia all'hardware che al software. L'uso inoltre di variabili interne in virgola mobile a 80 bit ci ha consentito di rendere trascurabile il contributo che l'errore di arrotondamento (rounding error, cfr. cap. VI) fornisce al calcolo finale dell'incertezza composta (ed eventualmente estesa). A dimostrazione dell'estrema versatilità del prodotto ottenuto, è stata sviluppata una versione che consente la misura automatica dell'impedenza di sensori la cui deformazione é direttamente pilotata dal programma stesso tramite un motore passo-passo, a sua volta controllato tramite bus seriale RS232 in parallelo con il canale di acquisizione dati.

Per rendere l'idea della quantità di lavoro effettuato in questi anni si osservi che il programma è passato dalle originarie 35.000 linee di codice, come citato in [1], alle attuali 300.000 linee (al momento della consegna alle stampe di questo lavoro), tutte scritte in puro C++ Object Oriented, multithreading e, quando necessario per esigenze di tempo reale, facendo uso esclusivo di chiamate alle API di sistema o routine scritte direttamente in linguaggio assemblativo.

Come già accennato si ritiene utile riportare nell'appendice I, con semplici adattamenti, la descrizione originale del programma di partenza, mentre nel prossimo paragrafo è riportata una breve introduzione al programma nel suo stato attuale.

1.2 Il software realizzato: Visual Analyser (VA)

Il software oggetto di questo lavoro è un "pacchetto" che consente di realizzare via software un insieme di strumenti di misura elettronici; esso gira su sistemi operativi della famiglia Windows e tramite opportuni accorgimenti anche sulla maggior parte dei sistemi Linux. Il software è chiamato Visual Analyser, abbreviato in "VA", e consiste in una finestra principale in cui sono subito disponibili due tra i più importanti strumenti disponibili, ossia l'oscilloscopio e l'analizzatore di spettro. Gli altri strumenti compaiono in finestre separate, previa selezione da apposito menù.

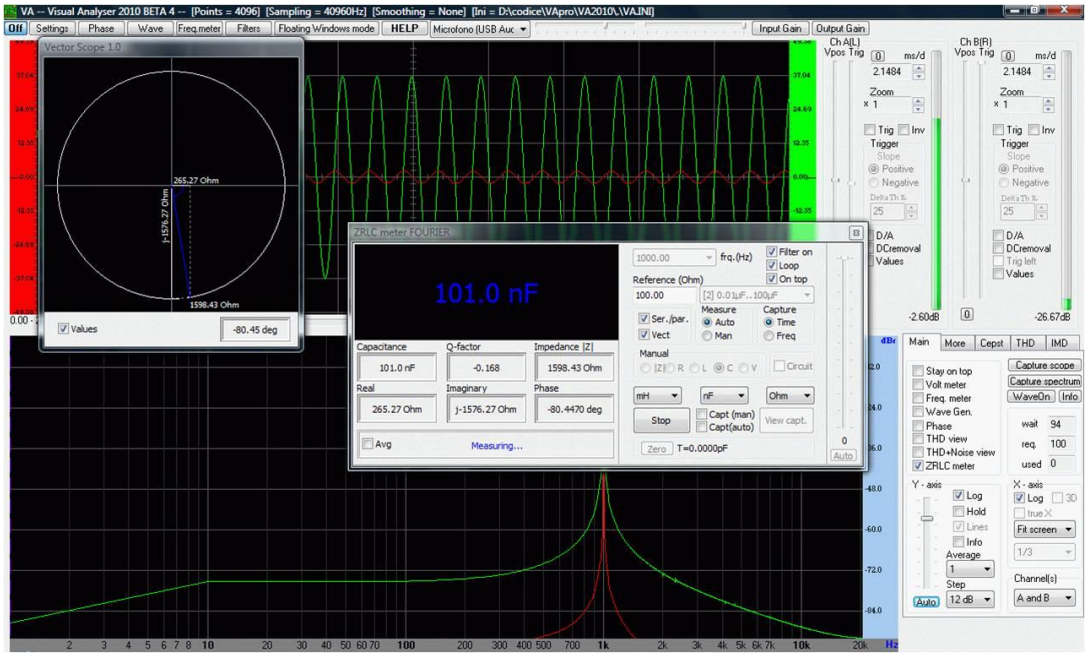


Figura 4: Visual Analyser con strumenti multipli in esecuzione

VA può essere utilizzato fondamentalmente i due modalità generali: a "finestre flottanti" o "standard" (default).

Nella modalità "finestre flottanti", il programma principale si riduce ad una piccola "barra dei comandi" posizionabile liberamente sullo schermo, che consente di selezionare le opzioni volute e tipo di strumenti desiderati; in tale modalità ogni strumento è completamente contenuto in una finestra separata, le cui dimensioni e posizione sono liberamente definibili dall'utente tramite mouse o tastiera. In pratica in questa modalità NON si utilizza, volutamente, l'approccio MDI (Multiple

Document Interface), ossia l'approccio che consiste nell'avere una finestra principale all'interno della quale "vivono" tutte le altre, ossia in essa vincolate.

Nella modalità "standard" (default), il programma gira in una finestra "principale" che contiene i due strumenti base più tutti i comandi necessari per invocare i restanti strumenti (in finestre separate) e selezionare le varie opzioni generali e/o specifiche per ogni singolo strumento.

Tutti gli strumenti possono essere eseguiti simultaneamente; VA è scritto sfruttando il paradigma multithreading, disponibile sia in sistemi operativi Windows che Linux (sebbene tra i due vi siano delle differenze). Su macchine multi-core è dunque addirittura possibile che il sistema operativo assegni un processore hardware per ogni strumento in esecuzione, che diviene in tal caso effettivamente eseguito in parallelismo reale invece che in maniera concorrente. In tal senso, l'uso di schede di acquisizione "intelligenti", ossia dotate di microprocessore proprio, consentono di eseguire alcune funzioni anch'esse in completo parallelismo hardware (per esempio la generazione di forme d'onda può essere interamente devoluta ad esse, VA è in grado di effettuarne la completa gestione).

Inoltre è possibile utilizzare differenti schede per l'acquisizione dei segnali e la loro generazione. In molti casi è tuttavia consigliabile utilizzare una scheda che comprenda sia la scheda di acquisizione che di uscita (ossia il convertitore analogico-digitale ed il convertitore digitale-analogico), in quanto in tal modo viene presumibilmente utilizzato lo stesso segnale di clock per entrambi. Questo fatto consente, quando si effettuano misure che usano più strumenti in contemporanea (esempio generatore di funzioni e oscilloscopio) di avere segnali tra loro perfettamente sincronizzati, nel senso che apparirà chiaro in VII, dove parleremo del concetto di campionamento sincrono e argomenti correlati.

La configurazione del programma è automaticamente salvata alla chiusura, e un apposito meccanismo consente di salvare più configurazioni richiamabili a seconda delle esigenze; la configurazione salvata comprende tutte le scelte effettuate, la posizione e tipo di finestre aperte, le calibrazioni ed i dati acquisiti.

Il programma consente di selezionare il dispositivo di acquisizione dati e di generazione, separatamente, e di configurarlo a seconda delle esigenze e caratteristiche disponibili. Per esempio è possibile selezionare la frequenza di campionamento dei dati, sia in acquisizione che in generazione, e le dimensioni del buffer dei dati (cfr. capitolo architettura) e dunque la risoluzione dello strumento. Più un numero consistente di altri parametri la cui descrizione si trova in [1] e cap. V.

VA consente, tra le molteplici possibilità, di utilizzare anche una semplice scheda sonora come scheda di acquisizione dati (vedi appendice 1), sia interna che esterna (USB) ed in tal caso è possibile utilizzare come ingressi del segnale i canonici "line-in", "microfono" o "aux"; il segnale d'uscita verrà invece prelevato dalle casse/cuffie. Naturalmente con qualche cautela; la sensibilità d'ingresso di una scheda sonora si aggira sull'ordine del volt per gli ingressi generici e di qualche decina/centinaia di millivolt per l'ingresso microfonico. In tal senso è opportuno cautelarsi tramite semplici circuiti di attenuazione e protezione, ossia nella sua forma più semplice tramite partitori resistivi e diodi, tramite i quali implementare un vero e proprio attenuatore calibrato (x1, x10, etc.). Inoltre le impedenze d'ingresso sono dell'ordine delle decine di kΩ fino ad un massimo di qualche centinaio, pertanto in talune applicazioni è consigliabile utilizzare dei semplici amplificatori operazionali usati come adattatori d'impedenza; oppure come amplificatori nel caso di segnali particolarmente deboli. Analoga prudenza per il segnale di uscita: l'amplificatore della scheda sonora è generalmente di potenza modesta e destinato ad impedenze di carico dell'ordine dei 4..8Ω. In entrambi i casi e nella quasi totalità dei casi l'uso di schede sonore non contempla accoppiamenti in continua, ne in ingresso e ne in uscita, sebbene questo sia, per talune particolarissime applicazioni, compensabile via software.

L'uso di VA con schede sonore è contemplato come utilizzo di esso per scopi didattici [2] in quanto in tal modo diventa difficile determinare in maniera efficace l'incertezza di misura dei vari strumenti implementati; tuttavia, anche in tal caso è possibile effettuare una verifica della bontà delle misure effettuate per semplice confronto con strumenti di classe superiore od applicando un'analisi statistica ai

risultati (incertezza di tipo A). In taluni casi è poi possibile desumere ulteriori informazioni utili al calcolo dell'incertezza se si riesce a venire a conoscenza dei componenti elettronici utilizzati (per esempio i convertitori A/D e D/A, circuiti amplificatori e quarzi); in tal caso infatti è facile rifarsi ai "datasheet" dei costruttori che riportano informazioni preziose.

In ultimo, si vuole aggiungere a questa brevissima carrellata sulle principali caratteristiche di VA un accenno alle opzioni di taratura delle scale. VA ha la possibilità di tarare le sue scale direttamente in Volt o suoi sottomultipli, nel senso che il programma contiene una procedura automatica che effettua il confronto con una tensione di valore noto (continua o alternata di cui si conosce il valore efficace o picco o picco-picco) e consente dunque di graduare ogni scala (per esempio dell'oscilloscopio) direttamente in Volt. Si può altresì effettuare una taratura in dB oppure lavorare direttamente in percentuale fondo scala (default), intendendo in tal senso che, data la risoluzione in bit del convertitore analogico-digitale (esempio 16), le scale saranno riportate da 0 a 100%, dove 100% si intende un segnale avente il valore di 2^{16}=65535 intervalli.

1.3 Linguaggi per strumenti virtuali

Il vastissimo mondo degli strumenti virtuali, nel senso precisato nel paragrafo 1.1, vanta un discreto numero di pacchetti software che implementano un vero e proprio linguaggio dedicato per la realizzazione rapida di strumenti virtuali su PC con sistemi operativi Windows, Linux e Mac. In tal senso, è doveroso citare il diffusissimo "LabView", cui recentemente si sono affiancati altri produttori, tra i quali il meno noto ma altrettanto efficace "Measure Foundry". Essi consentono di implementare rapidamente una vastissima gamma di strumenti virtuali con un linguaggio "grafico" costituito da schemi a blocchi e diagrammi di flusso, ma parimenti consentono anche di inserire frammenti di codice (tipicamente in C++, ma anche altri linguaggi) oppure di richiamare DLL esterne scritte in un linguaggio arbitrario. Inoltre essi consentono di gestire i più famosi sistemi standard quali IVI, VISA, SCPI per controllare

strumenti tramite LAN, GPIB, PXI/VXI, LXI, oltre ad avere una imponente libreria di programmi predefiniti. Sono poi provvisti di una libreria di driver adatti a gestire la maggior parte dei moderni strumenti di misura che supportano il controllo remoto (gli strumenti non compresi nella lista si possono comunque aggiungere scaricandoli dal sito del costruttore o dal cd-rom eventualmente in dotazione allo strumento). Molti strumenti di misura "stand alone" consentono infatti di "dialogare" con il PC tramite USB, RS232 o standard di comunicazione più complessi. In tal modo è possibile realizzare pilotaggi remoti di strumenti commerciali dedicati, acquisire i campioni del segnale ed utilizzarli per implementare nuovi strumenti (virtuali) o semplicemente estendere le funzioni quelli esistenti. Si pensi per esempio a banchi di misura che vengono programmati per eseguire misure complesse, e che devono essere protratte per lunghi intervalli temporali oppure in periodi ben determinati in conseguenza di determinati eventi. E dunque alle enormi potenzialità che un sistema coordinato da un PC può fornire.

Il lavoro che presentiamo in questa sede è stato anche "personalizzato" per il pilotaggio / acquisizione dati da alcuni strumenti presenti in vari laboratori di ateneo, ma il punto è un altro. Il fatto di appoggiarsi a pacchetti come LabView implica tenere conto di due fattori importanti. Il primo è ovviamente il fattore costo; un pacchetto evoluto come LabView ha generalmente un costo non trascurabile; il secondo fattore è che necessita di competenze specifiche, che devono essere di volta in volta acquisite da chiunque subentri nella gestione del processo di misura. Un terzo fattore, forse il più importante, è la dipendenza da un prodotto gestito da politiche commerciali spesso avulse da contesti accademici.

Il lavoro che viene qui presentato è invece costituito da puro codice C++, la cui conoscenza dovrebbe essere patrimonio comune di uno studente medio di Ingegneria; la disponibilità del codice sorgente consente una flessibilità senza pari, ed il costo è praticamente irrisorio rispetto a pacchetti commerciali. Inoltre, la possibilità di creare pacchetti personalizzati per studenti e laboratori didattici virtuali [2] è un punto di forza ineguagliabile. Infatti, utilizzando un hardware a basso costo oppure la stessa scheda sonora, è possibile dotare gli studenti di un completo laboratorio di misure

che alla bisogna può essere utilizzato in ogni dove, oppure allestire dei laboratori di misure elettriche/elettroniche a bassissimo costo (in pratica, una sala con qualche PC).

1.4 Gli strumenti implementati

Prima di addentrarci in dettaglio sugli argomenti chiave di questo lavoro, e su i metodi innovativi proposti, preferiamo iniziare con un approccio di tipo "top-down", ossia di seguito si riporta una lista completa di tutte le funzionalità implementate in VA, in maniera "tecnica" e senza riferimento alcuno a come sono state realizzate; esse, come detto, sono in parte funzioni già presenti prima dell'inizio di questo lavoro, ma sono state comunque talmente riviste e modificate da potere essere comunque considerate differenti, più quelle completamente nuove, sviluppate interamente durante questi ultimi anni e con l'introduzione di concetti relativi all'incertezza e metodologie proprietarie per la riduzione dell'entità degli errori sistematici. Nel seguito della trattazione approfondiremo ogni punto ritenuto significativo ed esporremo anche il progetto hardware di un semplice circuito a basso costo da abbinare al programma per la misura di impedenze (cfr cap. IX).

VA implementa i seguenti strumenti:

1) Oscilloscopio, con rilevazione automatica dei principali parametri del segnale (frequenza con zero-crossing, valore medio, efficace, fattore di cresta, fattore di forma, vero valore efficace) e possibilità in tempo reale di conversione D/A su schermo;
2) Analizzatore di spettro, con rappresentazione lineare, logaritmica, a ottave, calcolo correlazione e cross correlazione;
3) Compensazione della risposta in frequenza, tramite applicazione di curva di risposta arbitraria, definita graficamente, e curve standard A,B,C applicabili anche in "parallelo" a quella customizzata;

4) Generatore di funzioni (senza aliasing) con possibilità di generazione di forme d'onda predefinite e personalizzate, queste ultime tramite tool che accetta i coefficienti dello sviluppo in serie di Fourier oppure consente una costruzione "grafica" della forma d'onda stessa; possibilità di generazione "continua" ed in tempo reale della forma d'onda oppure in loop su buffer hardware interno della scheda sonora o di acquisizione in genere; generatore di impulsi, rumore rosa e bianco con possibilità di selezionare tra diverse distribuzioni (gauss, uniforme, t-student); generatore sinusoidale a sweep;

5) Frequenzimetro, con thread a bassa priorità per il calcolo con risoluzione predefinita e calcolo dell'incertezza;

6) Voltmetro AC/DC (DC per le sole schede di acquisizione accoppiate in continua) con rilevazione vero valore efficace (true RMS), picco, picco-picco, medio e livelli in dB;

7) Tarature degli strumenti in tensione (volt, millivolt) o decibel tramite procedura automatica e salvata su file;

8) Filtri digitali: è possibile inserire una serie canonica di filtri digitali (passa basso, alto, elimina banda, notch, notch inverso, allpass, diodo, rimozione componente continua) nel "percorso" del segnale per effettuare misure sui segnali filtrati; possibilità di inviare in uscita, in tempo reale, il segnale filtrato;

9) Cattura segnali nel dominio del tempo e frequenza con stampa e salvataggio; possibilità di cattura di schermate grafiche e salvataggio in clipboard di dati in formato testo e grafico;

10) Cattura dei segnali con threshold e pre-acquisizione, illimitata nel tempo;

11) Distorsiometro (THD, THD+noise) con cattura e algoritmo di compensazione della THD propria della scheda di acquisizione;

12) Rilevazione automatica della risposta in frequenza di dispositivi, tramite uso automatico di (1), (2) e (4);
13) ZRLC misura d'impedenza (resistenza, capacità, induttanza, parte reale e immaginaria, angolo di fase,) con possibilità di visualizzazione grafica dell'impedenza (vettorscopio), valutazione dell'incertezza delle misure, media infinita, calcolo incertezza con metodi statistici, procedura di auto taratura/calibrazione, sweep di misura nel dominio del tempo e della frequenza con acquisizione del grafico e possibilità di salvataggio, impostazione della resistenza di riferimento e tolleranza relativa, uso di modello serie e parallelo, calcolo di fattore di merito Q e D, azzeramento manuale, autodeterminazione dei livelli del segnale;
14) Cepstrum di un segnale;
15) Altri

Tutte questi strumenti sono stati simulati utilizzando otto thread all'interno del più generale processo "Visual Analyser" di cui due sono in esecuzione permanente e sei in esecuzione "on demand", nel senso che sono creati solo al momento in cui è richiesta una specifica funzione (per esempio si attiva il "generatore di funzioni") e cessano di esistere con la chiusura di essa (esempio messa in "off" del generatore di funzioni).

Ognuno degli strumenti simulati è doppia traccia, se si usa una normale scheda a due canali, oppure ad "n" tracce se si usa un hardware di acquisizione multicanale. Quest'ultima modalità richiede la ricompilazione del programma in una versione specializzata.

1.5 Organizzazione dei capitoli

Il lavoro si articola in nove capitoli e tre appendici, oltre a una completa bibliografia e ringraziamenti/conclusioni.

Il primo capitolo presenta una introduzione generica a tutto il lavoro oltre ad una trattazione del concetto di strumentazione virtuale; il secondo capitolo presenta una breve introduzione ai moderni concetti di misura e incertezza riportando le linee fondamentali della GUM [22]. Il terzo capitolo tratta dei sistemi digitali in genere, ed in particolare di quelli utilizzati negli strumenti di misura, oltre a DSP (Digital Signal Processor), microcontrollori e convertitori A/D e D/A, arrivando a definire il concetto di "circuito universale"; il quarto capitolo evidenzia con un certo dettaglio le cause di incertezza più significative delle schede di acquisizione generalmente utilizzate negli strumenti di misura virtuali. Nel quinto capitolo vede la luce una dettagliata descrizione dell'architettura del programma, mentre nel sesto si effettua l'analisi di una causa specifica di incertezza (il rounding error), tipica dei sistemi di misura basati su microprocessori e con variabili rappresentate in registri a dimensione finita. Sempre nello stesso capitolo verrà dimostrato come questa causa d'incertezza sia trascurabile rispetto a molte altre. Il settimo capitolo tratta delle cause specifiche di incertezza introdotte dall'FFT, algoritmo alla base di molti strumenti realizzati nel programma. Il capitolo ottavo descrive ZRLC, uno strumento implementato nel programma per la misura di impedenze, per il quale è stata sviluppata una tecnica proprietaria per l'eliminazione delle principali cause di errori sistematici; successivamente viene descritta l'implementazione di un semplice calcolo dell'incertezza totale. Il capitolo nove, infine, descrive il progetto di un hardware specifico realizzato in collaborazione con la rivista Nuova Elettronica per lo strumento ZRLC.

Nell'appendice 1 viene riportata la descrizione della versione del programma presa come punto di partenza, sviluppato e descritto in [1] (tesi di laurea); l'appendice 2 riporta uno stralcio del "datasheet" di un componente elettronico fondamentale per la realizzazione dell'hardware descritto nel capitolo nove (PCM2902). Infine l'appendice 3 riporta la lista di una serie di funzioni presenti in VA non descritte nei normali capitoli perché non direttamente legate all'argomento oggetto del presente lavoro, ma per esso indirettamente sviluppate come ausilio o comunque ritenute ad esso collegabili in futuri sviluppi.

Capitolo 2

Misure, Incertezza e metodi di valutazione

2.1 Introduzione

In questo capitolo vengono presentati i concetti di base a quella che possiamo definire "teoria della misurazione". Per dettagliate informazioni sull'argomento ed approfondimenti si rimanda a [22] ed alla notevole quantità di materiale che possibile reperire (anche) in rete. Nella stesura di questo capitolo si è fatto riferimento alle dispense del corso "Elaborazione dei segnali di misura" tenuto presso l'Università di Roma "Tor Vergata" [39].

Il concetto di misura è estremamente importante in un numero vastissimo di campi, che spaziano dalle scienze finanziarie sino all'elettronica e alla medicina; nondimeno esso è spesso molto poco ben compreso o comunque spesso inteso in accezioni molto diverse tra loro; situazione che porta alla produzione di risultati, ossia di misure, non comparabili tra loro.

In questo lavoro si è fatto esplicito riferimento ad una guida che è rapidamente diventata uno standard accettato a livello internazionale, i cui elementi essenziali sono riportati in questo capitolo.

Nel 1997 ben 7 organizzazioni internazionali hanno preparato la prima versione della guida nota come GUM (Guide to the expression of Uncertainty in Measurement)

[22] e del VIM (International Vocabulary of basic and general terms in Metrology) [38]. Esse sono:

1) BIPM, Bureau International des Poids et Mesures;
2) IEC, International Electrotechnical Commission;
3) IFCC, International Federation of Clinical Chemistry and Laboratory Medicine;
4) ISO, International Organization for Standardization;
5) IUPAC, International Union of Pure and Applied Chemistry;
6) IUPAP, International Union of Pure and Applied Phisics;
7) OIML, International Organization of Legal Metrology.

Nel 1997 assieme all'ILAC (International Laboratory Accreditation) formano il JCGM (Joint Commitee for Guides in Metrology), composto da due gruppi di lavoro (Working Groups) il WG1 e il WG2.

Il WG1 si occupa dell'espressione dell'incertezza di misura e della normativa associata, il WG2 si occupa di aggiornare costantemente il VIM.

Sul sito della BPM (www.bipm.org) è possibile scaricare gratuitamente la versione Inglese (considerata lingua "originale") di tutti i documenti prodotti e la lista completa di quelli in fase di stesura (all'atto della stampa di questo lavoro). Si veda in particolare:

http://www.bipm.org/en/publications/guides/gum.html

di cui si riporta la lista dei documenti attualmente presenti ed in fase di approvazione/elaborazione:

Evaluation of measurement data – Guide to the expression of uncertainty in measurement JCGM 100:2008 (GUM 1995 with minor corrections)	
Note: JCGM 100:2008 is also available in HTML form from the JCGM portal on ISO's website.	

Ed inoltre (il simbolo del pdf indica una effettiva disponibilità del documento):

↘	***Evaluation of measurement data – An introduction to the "Guide to the expression of uncertainty in measurement" and related documents*** JCGM 104:2009	
↘	***Evaluation of measurement data – Supplement 1 to the "Guide to the expression of uncertainty in measurement" – Propagation of distributions using a Monte Carlo method*** JCGM 101:2008	
↘	*Evaluation of measurement data – The role of measurement uncertainty in conformity assessment*	
↘	*Evaluation of measurement data – Concepts and basic principles*	
↘	*Evaluation of measurement data – Supplement 2 to the "Guide to the expression of uncertainty in measurement" – Models with any number of output quantities*	
↘	*Evaluation of measurement data – Supplement 3 to the "Guide to the expression of uncertainty in measurement" – Modelling*	
↘	*Evaluation of measurement data – Applications of the least-squares method*	

2.2 Concetti di base

Uno strumento di misura, sia esso virtuale o tradizionale, effettua delle misure, ossia risponde alla semplice domanda “quanto vale una determinata grandezza?”; la risposta a questo semplice quesito è tutt’altro che banale. Cercheremo di dare le definizioni essenziali che ci consentiranno di padroneggiare la terminologia di base della materia, e che useremo nel corso del presente lavoro. Per una trattazione esaustiva si rimanda ancora alla GUM e libri su di essa basati.

Tradizionalmente, quando si parla di misure si è portati ad associare ad esse il concetto di errore, sottintendendo dunque che esista un *valore vero* ed un *errore*, ossia uno “sbaglio” che si potrebbe commettere nell’effettuare la misura. Errore per

esempio commesso dallo strumento in se, oppure dall'imperizia di chi effettua la misura, oppure da una cattiva *taratura* dello strumento stesso. Invero, il valore reale di una misura non è noto, ne è possibile conoscerlo realmente proprio in quanto stiamo effettuando la misura. In altre parole, stiamo *perturbando* il sistema che stiamo sottoponendo a misura, sistema che non può sottrarsi al ben noto *principio di indeterminazione di Heisenberg*.

Più correttamente (e modernamente) si tende a parlare di *valore misurato* ed *incertezza* ad esso associato. Diamo le seguenti definizioni, liberamente tratte dalla GUM:

Misura: procedimento con cui si determina il valore di una grandezza fisica *sperimentalmente* mediante uno *strumento di misura;*

Misurando: particolare grandezza soggetta ad una misura;

Risultato di una misura: insieme di valori attribuiti ad un misurando e descritto da: 1) valore del misurando, 2) incertezza, 3) unità di misura;

Incertezza di misura: parametro, associato al risultato di una misura, che caratterizza la dispersione dei valori che possono *ragionevolmente* essere attribuiti al misurando.

Al concetto classico di errore e valore vero si sostituisce pertanto il concetto di incertezza, ossia al modello deterministico, che generalmente presentava una valore del misurando ed una fascia di valori in cui il valore "vero" era sicuramente compreso, si sostituisce il modello probabilistico, in cui si indica al posto del valore vero un valore stimato come il "più probabile" candidato ad essere il valore effettivo ed una fascia di valori che esprimono la distribuzione probabilistica dei valori misurati, ossia l'intervallo in cui è possibile ricada il valore effettivo con un determinato valore di probabilità.

Riportando le esatte parole usate nella GUM diremo dunque che l'obiettivo di una misurazione è quello di determinare il valore del misurando ossia il valore della particolare grandezza o grandezza in senso determinato da misurare. Una

misurazione, pertanto, comincia con una adeguata definizione del misurando, del metodo di misurazione e del procedimento di misurazione.

La **Misurazione** è "l'insieme di operazioni che ha lo scopo di determinare un valore di una grandezza".

La **Grandezza** è l'attributo di un fenomeno, d'un corpo o di una sostanza, che può essere distinto qualitativamente e determinato quantitativamente; in particolare la grandezza può essere riferito ad una grandezza in senso generale (es. tempo, massa, temperatura, resistenza elettrica) oppure determinato (es. lunghezza di una data barra, resistenza elettrica di uno specifico componente elettronico, concentrazione di quantità di molecole di gas in un dato contenitore)

Il **Valore** di una grandezza è l'espressione quantitativa di una grandezza in senso determinato, generalmente in forma di una unità di misura moltiplicata per un numero.

Il **Metodo di Misurazione** è la sequenza logica, descritte in termini generali, usate per effettuare una misurazione.

Infine il **Procedimento di Misurazione** è l'insieme delle operazioni, descritte in termini dettagliati, usate per effettuare determinate misurazioni secondo un dato metodo.

Semplificando, ogni misura (elettronica o no) viene ad essere caratterizzata da un valore misurato, una unità di misura e una fascia di incertezza, ossia un intervallo in cui è probabile (con probabilità assegnata, es. 90%) che cada il valore vero. Il valore vero e l'errore risulta pertanto una semplice e comoda *astrazione,* utile per comprendere altri concetti, ma di fatto inconoscibile. Per esempio, la misura di una tensione (differenza di potenziale) potrebbe essere indicata nella seguente maniera:

$$13{,}4 \pm 0{,}2 \text{ Volt}$$

In cui abbiamo detto che 1) il valore misurato è 13,4 volt 2) abbiamo dichiarato che il valore vero cade nell'intervallo 13.2…13.6 volt ma non ci è dato di sapere dove esattamente. Va da se che la misura è tanto più accurata quanto minore è l'incertezza. L'incertezza è dunque una stima del *dubbio* che abbiamo nel risultato di una

determinata misura. Nel prossimo capitolo cercheremo di chiarire cosa si intenda per accuratezza ed altri importantissimi concetti ad essa associati e frequentemente usati nel corso della presente trattazione.

Si vuole qui ulteriormente chiarire che la modalità di rappresentazione della misura data poc'anzi va intesa in accezioni differenti, a seconda che si tratti di un modello "old-style" di tipo deterministico, oppure di tipo probabilistico. Fermo restando che le modalità di presentazione della misura (valore_misura ± incertezza) possono essere apparentemente simili. Nel modello deterministico classico, è da intendere che il valore misurato riportato è "ragionevolmente" compreso nell'intervallo indicato, ed al limite "certamente" in esso compreso, sebbene anche in questo caso non si sappia esattamente in che punto. Inoltre, non si sa nulla a riguardo delle modalità con le quali il valore del misurando potrebbe ricadere nell'intervallo assegnato, od in altre parole, non si dichiara la *distribuzione di probabilità* ad esso assegnata. che pertanto si è spesso portati a ritenere di tipo uniforme. Ad esempio, l'indicazione del parametro "tolleranza" per una resistenza di tipo commerciale, è in genere riportata come valore percentuale (es. 5%). Si è normalmente portati a ritenere che il valore sia compreso nella fascia indicata dalla tolleranza in maniera "equiprobabile" per ogni valore assunto (così che una resistenza da 1000 ohm e 5% di tolleranza potrebbe assumere un valore qualsiasi compreso tra 975Ω e 1025Ω in maniera paritetica). Il modello deterministico, esprime l'incertezza mediante la seguente relazione:

$$M = (M_0 \pm I)U \tag{2.1}$$

Dove I è l'incertezza assoluta della misura, espressa nella stessa unità di misura U di M_0. L'intervallo 2I rappresenta pertanto la fascia di valore assegnata come misura del parametro. Speso viene fornita l'incertezza relativa percentuale definita come $I_r=I/M_0$ o espressa come percentuale moltiplicandola per il fattore 100.

Nel più moderno modello probabilistico, la misura è espressa nella seguente maniera:

$$x_0 \pm u(x) \quad (\text{ o } u_c(y)) \tag{2.2}$$

la quantità misurata è in questo caso trattata come una variabile aleatoria, a cui è associata una funzione densità di probabilità (ddp) ed il valore assegnato al

misurando è una stima del valore atteso della sua ddp. Dunque, x_0 è il valore atteso e $u(x)$ è la deviazione tipo della ddp, assunta come informazione quantitativa dell'incertezza di misura, ed è detta *incertezza tipo*. In questo caso il significato dell'intervallo individuato è che esso rappresenta l'intervallo in cui il misurando è compreso con una determinata probabilità e distribuzione. Si definisce in tal caso come "intervallo di fiducia" e di definisce *incertezza estesa* $U(x)$ il valore di $u(x)$ moltiplicato per un fattore costante k ($U(x) = ku(x)$) detto fattore di copertura, che consente di estendere l'intervallo di fiducia (nel senso definito poc'anzi) in cui è ragionevolmente compreso il misurando. In altre parole, se la distribuzione di probabilità dei valori misurati è di tipo gaussiano, scegliendo un fattore di copertura $k = 3$ otteniamo una probabilità del 99.7% che il misurando sia compreso entro l'intervallo di fiducia, mentre rimarrebbe del 68.3% circa nel caso in cui $k = 1$.

2.3 Errori casuali e sistematici

La suddivisone in errori sistematici e casuali, alla luce dei nuovi approcci, potrebbe apparire poco opportuna; invero, essa permane, anche se talvolta talune incertezze di tipo sistematico, a seconda dei contesti, possono esser fatte ricadere nella categoria opposta. Sostanzialmente, gli errori *casuali* sono dovuti a variazioni casuali del misurando, o delle grandezze d'influenza; essi sono tipicamente valutati e diminuiti mediante un processo di misure ripetute: l'errore casuale ha media nulla. L'errore *sistematico* è relativo ad una quantità "fissa" che si viene ad aggiungere (o sottrarre) alla misura, dovuta alle cause più disparate, e comunque non evidenziabile tramite misure ripetute perché a media generalmente non nulle. Come vedremo nei prossimi paragrafi, l'errore sistematico deve essere eliminato prima di effettuare qualsivoglia analisi statistica, o quantomeno noto a priori per poter poi essere eliminato dai risultati ottenuti. Quindi l'errore si dice *sistematico* se nella misura esiste una componente di errore fisso, ossia eliminabile con una semplice operazione algebrica. Per esempio, se misuriamo una resistenza con dei puntali che sono dotati di fili molto lunghi, aggiungeremo al valore misurato la resistenza dei fili stessi; per riportare la

misura al valore corretto dovremo sottrarre il valore della resistenza dei cavi alla misura effettuata. Diamo le definizioni rigorose dei due tipi di errori:

Errore casuale

La variabile aleatoria che descrive l'errore casuale (o random) è data dalla differenza fra la singola misura e la media di infinite misure ottenute sotto condizioni di ripetibilità [v. par. 2.6] (che tende al valore atteso o valore vero).

Errore sistematico

La differenza fra il contributo di errore totale e l'errore casuale è detto errore sistematico. Esso è spesso detto anche polarizzazione (bias) o distorsione, perché ha un carattere deterministico e dipende in maniera diretta o indiretta dal misurando stesso. Questo tipo di errore deve essere corretto mediante un fattore di correzione che deve essere almeno stimato.

2.4 Accuratezza, precisione, risoluzione

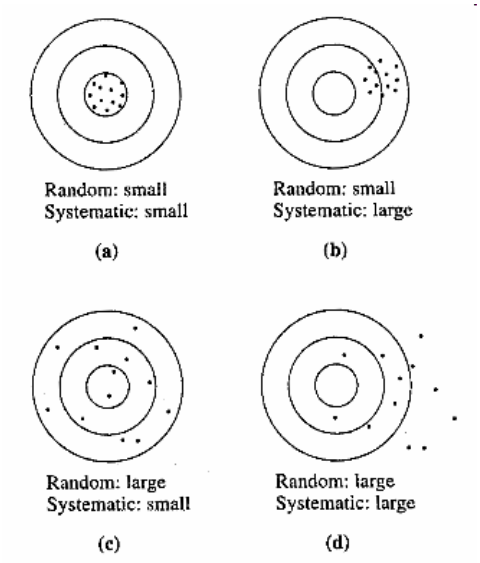
Accuratezza di una misura è un concetto molto diverso da *precisione*. Sovente i due termini vengono confusi, o usati come sinonimi. Con un po' di esercizio mentale ci si convincerà che i concetti sono profondamente diversi.

Per *accuratezza* di una misura intendiamo quanto una determinata misura è tanto più vicina al valore "convenzionalmente" vero, ossia usando una terminologia antiquata, quanto minore è l'errore commesso. Quindi eventuali ripetizioni della misura vedono i valori misurati concentrarsi attorno al valore vero, in maniera tanto maggiore quanto più accurata è la misura.

La *precisione* indica invece quanto una misura (anch'essa ripetuta "n" volte) si sparpaglia attorno al valore medio che non necessariamente coincide con il valore convenzionalmente vero (presenza di errore sistematico). In pratica una misura molto precisa vede i valori della misura concentrarsi attorno ad un valore che non è necessariamente quello vero, per la presenza di un errore sistematico. Relativamente agli strumenti di misura, la precisione di uno strumento di misura può vedersi come il

potere risolutivo dello strumento, per esempio numero di cifre nel caso di uno strumento digitale o finezza della scala graduata nel caso di strumento analogico. Come detto sovente si fa confusione tra il concetto di accuratezza e precisione; è possibile allora usare al posto del termine precisione il termine *risoluzione*, intendendo per essa la "più piccola variazione del misurando che provoca una variazione nel valore indicato".

Ricapitolando, uno strumento può essere molto preciso, ossia avere un'ottima risoluzione, ma in effetti essere poco accurato. Cioè quindi, può fornire una estrema finezza di scala, ma dare delle misure comunque lontane dal valore vero del misurando. Oppure può essere preciso ed accurato, nel qual caso è dotato di "finezza scala" è fornire misure molto vicine al "valore vero". Riportiamo, in ultimo, una figura che può sintetizzare in maniera efficace i concetti appena esposti; essa è stata riportata dal testo J. Taylor "An Introduction to Error Analysis" (il valore "vero" ideale è nel centro dei cerchi).



Nella figura (a) è riportata una misura con errore sistematico piccolo, e dunque precisa, ed un errore casuale piccolo (e dunque accurata). Le misure dunque si addensano verso il valore vero, e sono poco "sparpagliate". Viceversa, nel caso (b) abbiamo ancora un errore casuale piccolo, e si vede perché i valori sono addensati attorno al valore medio, che tuttavia è sbagliato per la presenza di un errore sistematico elevato. E dunque abbiamo una misura precisa (errore casuale piccolo) e

poco accurata (errore sistematico elevato) . I casi (c) e (d) sono conseguentemente di immediata interpretazione.

2.5 Misure dirette e indirette

I metodi di misurazione si dividono in due grandi famiglie, dirette ed indirette. Nel corso del presente lavoro faremo quasi esclusivamente riferimento a misure di tipo indiretto, come avremo modo di notare nei capitoli successivi.

Metodi di misurazione diretti

Essi si basano sul confronto tra la quantità misurata ed una grandezza della stessa specie generata da un campione locale a da esso memorizzata. Tipica è la misura eseguita con il metodo a lettura diretta, ossia assegnando il valore del misurando dopo aver letto l'indicazione dello strumento al cui ingresso è applicato il misurando. Questo tipo di misura è generalmente effettuata previa operazione di taratura (eseguita una-tantum ma per taluni strumenti anche appena prima di ogni misura); intendendo per essa l'applicazione della relazione $M = f_T(L)$ dove L è l'indicazione fornita dallo strumento M è il risultato della misura ed f_T è il diagramma di taratura, ossia quella relazione che fa corrispondere alla grandezza L la grandezza M. Una altro tipico esempio di metodo di misurazione diretta è la classica misura ottenuta per confronto: si pensi al confronto tra campioni lunghezza (misura di lunghezza con un righello).

Metodi di misurazione indiretti

Ricadono in questa tipologia un numero enorme di misure effettivamente realizzate; la maggior parte degli strumenti moderni sono dotati di un trasduttore, ossia un dispositivo capace di tradurre una grandezza in un altro tipo di grandezza, poi effettivamente misurata e messa in relazione con la prima.

Si pensi ad esempio ad un voltmetro analogico, la cui lettura sullo strumento graduato (galvanometro a bobina mobile solidale con un indicatore a lancetta) fornisce una posizione; quest'ultima è messa in corrispondenza biunivoca con il

valore della tensione elettrica (differenza di potenziale) per confronto con una scala graduata. Nel caso di un siffatto strumento una eventuale operazione di "taratura" non può ovviamente consistere in un ridisegno della scala quanto un eventuale aggiustaggio del valore di "offset". Nel caso di un voltmetro digitale la curva di taratura può invece essere contenuta direttamente nelle memorie interne dello strumento.

La definizione rigorosa asserisce che un metodo di misurazione è indiretto quando la misura del parametro di un sistema è assegnata mediante un calcolo che coinvolge altri parametri del sistema stesso o di altri sistemi con essi interagenti. La stima di questi parametri è tuttavia ottenuta mediante metodi di misurazione diretta.

In tal senso questo tipo di misurazione presuppone l'esistenza di un modello matematico che esprime in maniera esplicita il legame tra il misurando e le altre grandezze ottenute tramite misurazione diretta. Il modello matematico è genericamente espresso dalla seguente relazione:

$$M_I = f(M_{D1}, M_{D2}, M_{D3} \ldots M_{DN}) \tag{2.3}$$

Dove M_I rappresenta il misurando ottenuto e con M_{Di} le grandezze misurate per via diretta.

Nei prossimi paragrafi analizzeremo come sia possibile, a partire da misure ottenute per via indirette e dunque per tramite di un modello matematico come quello indicato nella 2.3, calcolare l'incertezza sulla misura M_I a partire dalla conoscenza delle varie incertezze sulle M_{Di} (legge di propagazione delle incertezze). Prima di fare questo è tuttavia necessario enunciare altri importanti concetti.

2.6 Ripetibilità e riproducibilità

Riportiamo le definizioni che troviamo nella GUM e in [39]; per definire questi due concetti, si devono dare prima delle definizioni preliminari indispensabili, che sono le "condizioni di ripetibilità" per la ripetibilità, e le "condizioni di riproducibilità" per la riproducibilità.

Condizioni di Ripetibilità: per condizioni di ripetibilità di misurazione si intende una condizione di misura in cui si mantengono fissi l'operatore, il sistema di misura, la stessa procedura di misurazione e le condizioni operative, e si effettuano misure ripetute in un intervallo temporale piccolo.

Per *Ripetibilità* di una misurazione intendiamo dunque lo sparpagliamento dei risultati della misurazione in condizioni di ripetibilità.

Condizioni di Riproducibilità: per condizioni di riproducibilità della misurazione si intende il caso in cui si possono cambiare le condizioni di misura, fra cui il principio di misurazione, il metodo di misurazione, l'osservatore, lo strumento, il luogo, i campioni di riferimento, le condizioni operative e l'intervallo temporale.

Per *Riproducibilità* intendiamo dunque il grado di concordanza tra le misure dello stesso misurando effettuate in condizioni di riproducibilità.

Queste due definizioni sono molto importanti, perché in sostanza forniscono due importantissimi concetti: semplificando, nel primo caso (ripetibilità) abbiamo a che fare con misurazioni condotte in un determinato luogo e con una determinato strumento (e relative "condizioni al contorno") in tempi brevi. Nel secondo caso (riproducibilità) stiamo definendo un concetto più ampio, che cerca di quantificare il grado di accordo tra misure effettuate in luoghi differenti, persone diverse e magari differenti strumenti.

2.7 Tipi di incertezza (A e B)

Diamo ora un cenno alle due grandi categorie in cui sono suddivise le tipologie di incertezze, così come definite dalla GUM. Essenzialmente una prima categoria è relativa alle incertezze determinabili tramite metodi statistici, e quindi applicabili a errori che vengono definiti "casuali" ossia la cui natura ben si presta ad essere quantificata (appunto) con metodi statistici; questo tipo di incertezze è detta di *tipo A*.

Si osservi che gli errori sistematici, ossia non casuali, non possono essere messi in evidenza da analisi statistiche, in quanto comportano uno spostamento del valore atteso della misura (valore più vicino al "valore vero" della misura) che non può essere in alcun modo evidenziato; nondimeno, come vedremo, anche per questo tipo di errori si può giungere ad una descrizione di tipo probabilistico per motivi di praticità ed omogeneità di calcolo.

Il secondo tipo di incertezza è detto di *tipo B*, e non deriva da analisi statistica, anche se ai fini del calcolo complessivo dell'incertezza vengono anch'esse descritte spesso da variabili aleatorie e distribuzioni di probabilità.

Incertezza di tipo A

Si definisce media di una serie di misure ripetute $x_1 \ldots x_N$, la seguente grandezza (media empirica):

$$\bar{x} = \frac{1}{N} \sum_{k=1}^{N} x_k. \tag{2.4}$$

Supporremo che le misure siano state eseguite in condizioni di ripetibilità, e che siano tra loro non correlate. La media così definita è a sua volta una variabile aleatoria che ha una distribuzione di probabilità asintoticamente gaussiana, come è facile dedurre dal teorema del limite centrale. Ancora, se le singole osservazioni hanno a loro volta una distribuzione di probabilità di tipo gaussiano, la media empirica avrà distribuzione gaussiana anche per un numero di osservazioni finito (dunque non solo asintoticamente). La varianza delle osservazioni è stimata mediante la varianza empirica definita dalla seguente relazione:

$$s^2(x) = \frac{1}{N-1} \sum_{k=1}^{N} (x_k - \bar{x})^2. \tag{2.5}$$

Si noti come un gruppo di misure ripetute, eseguite nelle condizioni di ripetibilità, dette anche "nominalmente uguali", può essere a sua volta ripetuto, per ottenere una serie di medie anch'esse caratterizzabili con una nuova media (media delle medie) ed una varianza. Quest'ultima si dimostra essere approssimabile con la quantità:

$$s^2(\bar{x}) = \frac{s^2(x)}{N} \tag{2.6}$$

al crescere del numero delle osservazioni della singola serie di misure fatte in condizioni di ripetibilità. Lo "scarto tipo sperimentale della media", ossia l'indice che ci fornisce la dispersione di diverse stime della media empiriche rispetto al valore "sperato" è dato dunque dal seguente valore:

$$s(\bar{x}) = \frac{s(x)}{\sqrt{N}} \tag{2.7}$$

In sintesi dunque, per un valore sufficientemente grande di osservazioni, possiamo considerare come una serie di misure effettuate in una medesima sessione di misure approssimi una "serie di serie" di misure effettuate in momenti diversi. L'indice calcolato dalla 2.7 quantifica la "bontà" del valore medio come rappresentativo del valore vero della misura; come incertezza *u(x)* assumiamo il valore calcolato dalla 2.7 che al tendere all'infinito di *N* tende a zero.

L'incertezza di tipo "A" è dunque rappresentata dal calcolo della media ottenuta con la 2.4 e dell'incertezza con la 2.7.

Incertezza di tipo B

Questo tipo di incertezza è calcolata sulla base di informazioni fornite da terze parti; come riportato dalla GUM esse possono essere ricavate da:

- dati di misurazioni precedenti;
- esperienza o conoscenza generale del comportamento e delle proprietà dei materiali e strumenti d'interesse;
- specifiche tecniche del costruttore
- dati forniti in certificati di taratura o altri;
- incertezze assegnate a valori di riferimento presi da manuali.

Pari valore ha un'incertezza di tipo B rispetto a quella di tipo A. Nel calcolo complessivo dell'incertezza finale (composta) potranno essere presenti contributi valutati mediante analisi statistiche, e dunque di tipo A e di tipo B. In tal caso

bisogna sommare tra loro quantità omogenee (ossia incertezze) ed a tale scopo si procederà, sulla base della conoscenza di parametri di tipo B, ad assegnare a questi ultimi una opportuna distribuzione di probabilità e conseguente scarto tipo della media, in maniera da poterle sommare tra loro (secondo regole che definiremo a breve, e comunque con diverse modalità a seconda che i vari contributi siano relativi a grandezze correlate o meno).

Per esempio, se abbiamo a che fare con un dato fornito dal costruttore (che per esempio dice che un determinato componente ha un certo valore ed una incertezza ad esso associata) al fine di far rientrare questa nel computo finale dell'incertezza, assegneremo alla rosa di valori che può effettivamente essere assunta dal componente una distribuzione di probabilità (esempio: resistore e tolleranza dichiarata dal costruttore, distribuzione supposta uniforme) e calcoleremo di conseguenza lo scarto tipo della media. In taluni casi saremo costretti a valutare l'incertezza esclusivamente con i dati relativi all'incertezza di tipo B, per esempio quando non sia fisicamente possibile effettuare più di una misura (lettura singola).

In figura 5 si riporta una tabella che riassume le distribuzioni più frequentemente utilizzate per descrivere incertezze di tipo B.

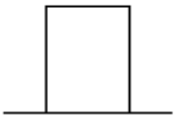
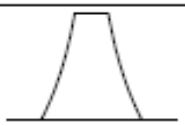
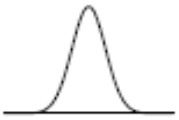
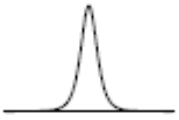
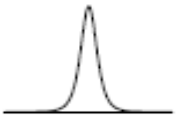
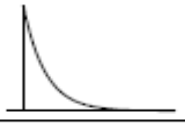
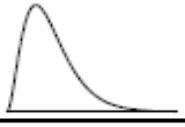
Available information	Assigned PDF and illustration (not to scale)	
Lower and upper limits a, b	Rectangular: R(a, b)	
Inexact lower and upper limits $a \pm d$, $b \pm d$	Curvilinear trapezoid: CTrap(a, b, d)	
Sum of two quantities assigned rectangular distributions with lower and upper limits a_1, b_1 and a_2, b_2	Trapezoidal: Trap(a, b, β) with $a = a_1 + a_2$, $b = b_1 + b_2$, $\beta = \lvert(b_1 - a_1) - (b_2 - a_2)\rvert / (b - a)$	
Sum of two quantities assigned rectangular distributions with lower and upper limits a_1, b_1 and a_2, b_2 and the same semi-width ($b_1 - a_1 = b_2 - a_2$)	Triangular: T(a, b) with $a = a_1 + a_2$, $b = b_1 + b_2$	
Sinusoidal cycling between lower and upper limits a, b	Arc sine (U-shaped): U(a,b)	
Best estimate x and associated standard uncertainty $u(x)$	Gaussian: N(x,$u^2(x)$)	
Best estimate x of vector quantity and associated uncertainty matrix $\boldsymbol{U}_{\boldsymbol{x}}$	Multivariate Gaussian: N($\boldsymbol{x}$,$\boldsymbol{U}_{\boldsymbol{x}}$)	
Series of indications $x_1,\ldots,x_n$ sampled independently from a quantity having a Gaussian distribution, with unknown expectation and unknown variance	Scaled and shifted t: $t_{n-1}\left(\overline{x}, s^2/n\right)$ with $\overline{x} = \sum_{i=1}^{n} x_i/n$, $s^2 = \sum_{i=1}^{n} \left(x_i - \overline{x}\right)^2 / \left(n-1\right)$	
Best estimate x, expanded uncertainty U_p, coverage factor k_p and effective degrees of freedom ν_{eff}	Scaled and shifted t: $t_{\nu_{eff}}(x, (U_p/k_p)^2)$	
Best estimate x of non-negative quantity	Exponential: Ex(1/x)	
Number q of objects counted	Gamma: G(q + 1, 1)	

Figura 5: distribuzioni tipiche da assegnare ad una incertezza di tipo B

2.8 Calcolo incertezza complessiva

Come detto, il calcolo dell'incertezza complessiva può essere effettuato generalmente dovendo sommare tra loro i contributi di incertezza determinati con metodi statistici (ossia di tipo A) e incertezze di tipo B. In generale, vale la regola della somma in quadratura:

$$U_{tot} = \sqrt{U_A^2 + U_{b1}^2 + \cdots U_{bm}^2} \tag{2.8}$$

nel prossimo paragrafo ci riferiremo al caso più generale di misure indirette ed eventualmente con correlazione o meno, e dunque considereremo il caso in cui, pur valendo sempre la 2.8, avremo a che fare con grandezze aggiuntive e calcoli che implicano la "propagazione" delle incertezze attraverso relazioni che "modellizzano" in maniera più o meno complessa la misura. L'incertezza definita nella 2.8 è detta incertezza tipo combinata.

Essa è in genere espressa dal termine $u_c(y)$ quando l'incertezza di y, stima di un misurando *Y* ossia risultato di una misurazione, è ottenuta mediante composizione opportuna delle incertezze tipo delle stime d'ingresso $x_1, x_2, \ldots x_N$. in altri termini, con tale notazione si vuole indicare che l'incertezza è relativa a misurazioni funzionalmente dipendenti da una relazione che lega le grandezze *x* al risultato *y*.

Incertezza complessiva in casi semplici

Facciamo riferimento al caso semplice di somma algebrica di grandezze affette da incertezza. Avendo a che fare con due misure *x* e *y* con relative incertezze δx e δy, ossia la seguente situazione:

$$x = x_0 \pm \delta x \qquad \text{e} \qquad y = y_0 \pm \delta y \tag{2.9}$$

Evidentemente potremo sicuramente scrivere:

$$x_0 + y_0 - \delta x - \delta y \le x + y \le x_0 + y_0 + \delta x + \delta y \tag{2.10}$$

Si potrebbe intuitivamente asserire che l'incertezza $\delta(x + y)$ sia uguale a $\delta x + \delta y$. In tal caso stiamo considerando un caso effettivamente lontano dalla realtà, per esempio se

le grandezze non presentano correlazione. In tal senso infatti è improbabile che le due incertezza "varino nella stessa direzione" ossia che si sommino dando una somma che coincida proprio con i due valori massimi e aventi lo stesso segno. E' pertanto molto più ragionevole porre in essere una somma in quadratura, ossia la 2.8 che ci consente di scrivere:

$$\delta(x+y)=\sqrt{(\delta x)^2+(\delta y)^2} \tag{2.11}$$

Nel caso invece in cui si sospetti una correlazione, allora potremo scrivere la relazione "peggiore" prima considerata, ossia:

$$\delta(x+y)=\delta x+\delta y \tag{2.12}$$

E vale sicuramente che:

$$\delta(x)+\delta(y)>\sqrt{(\delta x)^2+(\delta y)^2} \tag{2.13}$$

Medesimi risultati valgono nel caso della sottrazione (basta invertire i segni) mentre una relazione simile si può ricavare per il rapporto ed il prodotto tra due grandezze. Omettiamo le relazioni relative a questi ulteriori casi, dato il carattere puramente introduttivo del capitolo (si veda [47]).

2.9 Legge di propagazione delle incertezze

Considereremo in questo ultimo paragrafo una delle relazioni più importanti definite nella GUM, meglio nota come "legge di propagazione delle incertezze". Essa è di tipo generale, e qui riporteremo la definizione principale relativa al caso delle misure ripetute e misure a lettura singola. Riporteremo la trattazione relativa al più moderno modello probabilistico, che differisce concettualmente (e operativamente) dal vecchio modello deterministico, anche se può a prima vista sembrare simile (si veda

[22][39]). Detta con *y* la generica quantità misurata, legata alle *M* grandezze x_i dalla seguente relazione:

$$y = f(x_1 \dots x_H, x_{H+1}, \dots, x_M) \tag{2.14}$$

dove le *H* misure sono ottenute con metodi diretti a letture ripetute, mentre le rimanenti *M* - *H* grandezze sono ottenute con metodi diretti a letture singola oppure fornite da terze parti. Per ciascuna delle *H* grandezze abbiamo una serie di osservazioni, e dunque possiamo ricorrere alle formule viste in 2.7 per il calcolo della media e scarto tipo. Per le restanti *M* - *H* si dispone di un solo valore (ed eventualmente della conoscenza delle incertezze di tipo B). Il valore nominale y_0 della quantità misurata è dunque:

$$y_0 = f(\bar{x}_1, \cdots, \bar{x}_H, x_{(H+1)0}, \cdots, x_{M0}). \tag{2.15}$$

Supponendo che la relazione funzionale f sia nota in forma esplicita (caso talvolta difficile da verificare nella pratica) la GUM suggerisce di approssimare la stessa con un polinomio ottenuto tramite sviluppo in serie di Taylor troncato al primo ordine, cioè dunque di linearizzare *f* nell'intorno del valore nominale (valor medio). Se non si sospetta correlazioni tra le variabili in gioco, possiamo sicuramente far valere la seguente relazione per il calcolo dell'incertezza tipo combinata:

$$u_c(y) = \sqrt{\sum_{i=1}^{M} \left(\frac{\partial f}{\partial x_i}\right)^2 \Big|_{\vec{x_0}} u^2(x_i)} \tag{2.16}$$

dove il vettore x_0 è il vettore dei valori centrali (x_{10}, ..., x_{M0}) e $u(x_i)$ è l'incertezza tipo della generica grandezza x_i stimata con tecniche relative ad incertezza di tipo A o B.

Se invece alcune delle grandezze non sono indipendenti, vale la relazione più generale:

$$u_c(y) = \sqrt{\sum_{i=1}^{M} \left(\frac{\partial f}{\partial x_i}\right)^2 \Big|_{\vec{x_0}} u^2(x_i) + 2 \sum_{j=1}^{M-1} \sum_{l=j+1}^{M} \frac{\partial f}{\partial x_j}\Big|_{\vec{x_0}} \frac{\partial f}{\partial x_l}\Big|_{\vec{x_0}} u(x_j, x_l)} \tag{2.17}$$

Dove la correlazione è tenuta di conto dal termine $u(x_j, x_l)$ detta covarianza tra la coppia di grandezze indicate, che esprime il grado di dipendenza statistica tra la coppia di variabili. La relazione che generalmente è usata per quantificare la dipendenza statistica di due variabili aleatorie è un parametro (adimensionale) detto *coefficiente di correlazione* ed espresso come:

$$\rho(x_j, x_l) = \frac{u(x_j, x_l)}{u(x_j)u(x_l)}. \tag{2.18}$$

Nella GUM è possibile trovare una formula approssimata per il calcolo della $u(x_j, x_l)$ con i valori delle misure effettuate. (formula 17, paragrafo 5.2 di GUM [22]). Per finire, si osservi che il coefficiente di correlazione assume valori compresi tra -1 e +1 e vale zero se non c'è correlazione (variabili statisticamente indipendenti). Se $\rho(x_j, x_l) = +1$ allora si parla di correlazione lineare positiva tra le stime delle due grandezze e si può scrivere:

$$u_c(y) = \sqrt{\left[\sum_{i=1}^{M} \frac{\partial f}{\partial x_i}\Big|_{\vec{x}_0} u(x_i)\right]^2} = \sum_{i=1}^{M} \frac{\partial f}{\partial x_i}\Big|_{\vec{x}_0} u(x_i) \tag{2.19}$$

2.10 Calibrazione e taratura

Le operazioni di taratura e calibrazione sono spesso tra loro confuse, e sebbene intimamente connesse sono comunque operazioni differenti. Da notare altresì come il termine italiano calibrazione è derivato dall'Inglese "calibration" che invero andrebbe tradotto in italiano con il termine taratura. Il termine italiano calibrazione, è a sua volta tradotto con il termine inglese "adjustment", ossia l'operazione che in italiano si intende con il termine calibrazione è "aggiustamento".

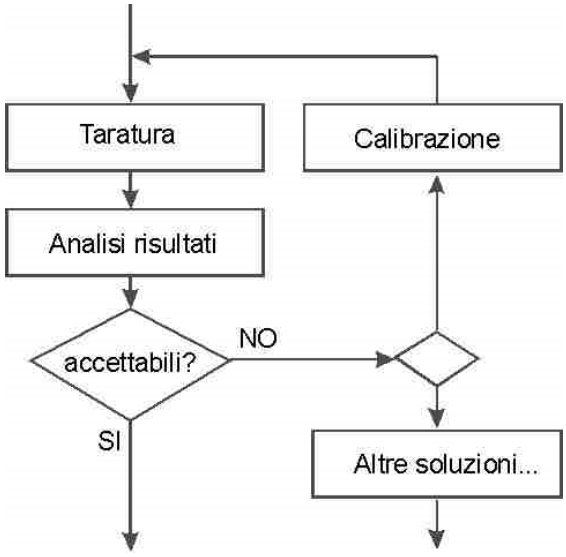
La taratura, come da definizione ufficiale, “è un'operazione che permette di definire le caratteristiche metrologiche di uno strumento, allo scopo di definirne la precisione” [38].

La calibrazione “ha come obiettivo rendere lo strumento più accurato e spesso, conseguentemente, migliorarne la precisione” [38].

La prima comprende pertanto una serie di procedure generali più macroscopiche e fondamentali, mentre la seconda è più particolare e riguarda il miglioramento delle caratteristiche di uno strumento eventualmente già tarato.

Per maggiori dettagli, inutili da elencare esaustivamente in questa sede, si rimanda a [38]. La figura riportata nella pagina successiva, reperibile in rete, mostra una esemplificazione della interdipendenza delle due differenti procedure.

La taratura comprende dunque anche la procedura che concorre a definire la “curva di taratura” dello strumento, che in Visual Analyser è inserita è definita come “calibration” (Visual Analyser, per motivi di massima generalità, esiste solo nella versione Inglese) che appunto è correttamente tradotta in italiano con il termine taratura.



Interdipendenza tra procedura di taratura e calibrazione

Pagina lasciata intenzionalmente bianca

Capitolo 3

Sistemi digitali di misura

3.1 Introduzione

In questo capitolo si vuole porre l'enfasi su alcuni aspetti fondamentali dei sistemi elettronici e informatici che sono alla base di un moderno strumento di misura (virtuale e non) di tipo digitale. Come ampiamente descritto nel capitolo primo, la struttura fondamentale di un "sistema digitale di misura" è costituita da un hardware dedicato all'acquisizione dei segnali da misurare, e da un sistema di elaborazione (a microprocessore) che consente la trasformazione dei segnali (amplificazione, filtraggio, etc.), l'effettiva implementazione dello strumento di misura (algoritmi specifici, ad esempio FFT per lo strumento analizzatore di spettro) con annessa rappresentazione della misura effettuata (a video, su carta, etc.). Gli elementi che metteremo in evidenza nei successivi paragrafi sono:

- La struttura generale di un sistema di acquisizione dati digitale (cosiddetto "circuito universale");
- I teoremi alla base della trasformazione dei segnali analogici in numerici (teorema di Shannon-Nyquist o teorema del campionamento);
- Digital Signal Processor (DSP), microprocessori specializzati per l'elaborazione numerica dei segnali in tempo reale;

- Convertitori A/D e D/A, per l'applicazione pratica del teorema del campionamento e conseguente trasformazione di segnali analogici in segnali digitali e viceversa.

3.2 Il circuito universale

La moderna elettronica nasce con l'introduzione massiccia di una famiglia di microprocessori nota come DSP che significa "Digital Signal Processor". Lo stesso acronimo è genericamente utilizzato (e spesso confuso) con il termine "Digital Signal Processing" che è fondamentalmente la denominazione anglosassone della materia "Elaborazione Numerica dei Segnali", anche se da taluni intesa in un'accezione più restrittiva. Quest'ultima si occupa dello studio degli algoritmi per il filtraggio ed elaborazione dei segnali digitali, mentre la materia nota come "Analisi Numerica" si occupa dell'adattamento "pratico" di algoritmi (teoricamente sviluppati per il mondo del "continuo") al mondo degli elaboratori, e dunque in parte connessa con l'Elaborazione Numerica dei Segnali.

Lo sviluppo di processori DSP sempre più performanti, di pari passo con la sintesi di algoritmi di enorme importanza ed efficienza (si pensi alla FFT) hanno fatto si che molti circuiti normalmente specializzati per singole funzioni siano confluiti in strutture circuitali "generali" che cambiano funzione in ragione di un diverso programma di calcolo su di essi eseguito. Ossia dunque una sorta di "circuito universale" che a grandi linee può essere schematizzato con la figura seguente:

Figura 6: il circuito universale

In essa possiamo notare come il segnale analogico venga convertito in segnale digitale tramite un convertitore Analogico-Digitale (ADC, A/D), elaborato in tempo reale da un microprocessore di tipo DSP (ed eventualmente coadiuvato da

microcontrollori o microprocessori di altro tipo), successivamente riconvertito in forma analogica da un convertitore Digitale-Analogico (DAC, D/A). L'elaborazione del segnale è effettuata tramite un programma di calcolo eseguito dal microprocessore dedicato di tipo DSP, e dunque la funzione di trasferimento complessiva del circuito visto come una "black-box" (dall'ingresso all'ADC all'uscita del DAC) varia in funzione esclusiva del programma di calcolo. Infatti, esso potrebbe implementare un filtro passa basso oppure un filtro passa alto o magari implementare il calcolo completo dello spettro del segnale tramite una FFT (Fast Fourier Transform) eccetera.

La struttura universale appena descritta è attualmente utilizzata in molteplici apparecchiature elettroniche e parimenti affiancata ad un altro tipo di microprocessore estremamente diffuso: quelli normalmente presenti in personal computer di qualsiasi fascia. A partire dagli anni 90 infatti la potenza di calcolo dei microprocessori presenti in sistemi di elaborazione dati "personali" (i cosiddetti Personal Computer o PC) è stata sufficiente per implementare gli algoritmi sino ad allora normalmente appannaggio esclusivo dei DSP. Microprocessori, tra le varie cose, anche particolarmente costosi (ai tempi). L'esistenza stessa di un programma di calcolo quale quello descritto in questo lavoro è stata possibile solo in virtù dei livelli prestazionali dei moderni PC e del loro relativo bassissimo costo (e conseguente diffusione capillare). E' parimenti necessario osservare che l'evoluzione hardware dei personal computer si è affiancata con l'evoluzione dei sistemi operativi ad essi associati e alla consequenziale capacità di "elaborazione concorrente" nelle sue varie forme (per esempio multi-threading). Gli strumenti di misura virtuali sfruttano pesantemente la concorrenza per implementare molteplici funzioni contemporanee.

Lo scenario attuale vede i Personal Computer e sistemi operativi annessi (Windows, Linux, OS-Leopard, etc.) talmente diffusi da poter considerare il loro costo virtualmente pari a zero, essendo uno strumento normalmente presente ed acquistato per altri scopi; in tal senso il costo di uno strumento virtuale viene considerato al netto della spesa necessaria per l'acquisto di un PC. Parimenti, anche il costo di un DSP è attualmente talmente basso da rendere la sua diffusione altrettanto capillare,

sino ad invadere il campo dell'elettronica di consumo standard (per esempio elettrodomestici) e persino a livello amatoriale (sono per esempio molto diffusi i "sistemi di sviluppo" basati su DSP e microcontrollori programmabili in C per applicazioni dilettantistiche).

3.3 Il teorema del campionamento

La struttura circuitale "general purpose" sommariamente descritta nel precedente paragrafo, si basa sull'applicazione dei convertitori analogico-digitali e digitali-analogici a loro volta basati su un teorema noto come Teorema di Shannon-Nyquist. Di esso daremo gli elementi fondamentali, data la sua importanza, e anche perché esplicitamente utilizzato anche nel programma stesso in talune funzioni aggiuntive. La descrizione del teorema attinge a piene mani da [1]. In linea introduttiva esso asserisce che, dato un segnale analogico tempo continuo a banda limitata, esso può essere completamente rappresentato (ed eventualmente ricostruito) da un segnale tempo discreto da esso derivato per tramite di un'operazione di campionamento. Ciò premesso veniamo all'enunciazione rigorosa del teorema di Shannon, per poi discuterla in termini pratici:

Ipotesi:

- sia *X(t)* un segnale a *banda limitata*, ossia la trasformata di Fourier *X(f)* sia uguale a zero per | f | > B (dove B è la banda del segnale). Ossia, sia *X(t)* un segnale il cui contenuto armonico vari da zero a un massimo (per esempio da 0 a 20000 Hz)
- sia la *frequenza di campionamento* maggiore od almeno uguale al doppio di B, ossia si prelevino *campioni* del segnale con una frequenza almeno doppia di quella della massima frequenza presente nel segnale (ossia B).

Allora (tesi):

- il segnale *X(t)* è rappresentato completamente dai suoi campioni

- il segnale può essere ricostruito con un filtro passa-basso avente frequenza di taglio Ft tale che B<Ft
- il segnale *X(t)* può essere ricostruito a partire dai suoi campioni con lo sviluppo in serie definito dalla seguente relazione:

$$x(t) = \sum_{n} x(nT_c) \sin c(\pi \frac{t - nT_c}{T_c})$$

(4.1)

Teorema la cui dimostrazione è omessa per semplicità. Detto in altre parole, il teorema asserisce che se da un segnale analogico preleviamo alcuni "campioni" (ossia il valore istantaneo del segnale ad un determinato valore di tempo t) e con una determinata cadenza, essi saranno sufficienti a ricostruire il segnale in maniera completa, ossia lo rappresentano esaustivamente.

Il segnale così ottenuto non è ancora comunque un segnale digitale; esso è solo un segnale che ricade nella categoria dei cosiddetti "segnali tempo discreti", ossia quei segnali la cui variabile indipendente (il tempo) assume un valore intero (discreto). Invero, la variabile dipendente è ancora una variabile tempo continuo, ossia una variabile il cui valore può assumere virtualmente infiniti valori. La digitalizzazione avviene se facciamo un passo ulteriore, ossia associamo un numero (intero) ad ogni campione (cd. quantizzazione). La quantizzazione associa, secondo un determinato criterio, uno dei numeri interi (e dunque discreti) disponibili ad un valore continuo della variabile dipendente. Ossia *approssimiamo* il livello reale del campione con il numero intero ad esso più vicino. Così facendo abbiamo commesso un errore, tanto più piccolo quanto più la profondità di bit della variabile intera è elevata (ossia la "grana" è più fine).

Chiariamo ulteriormente questi concetti con dei grafici. La figura 7 rappresenta il processo di campionare un segnale analogico, immaginandolo come ottenuto dalla moltiplicazione di un treno periodico di impulsi per il segnale da campionare.

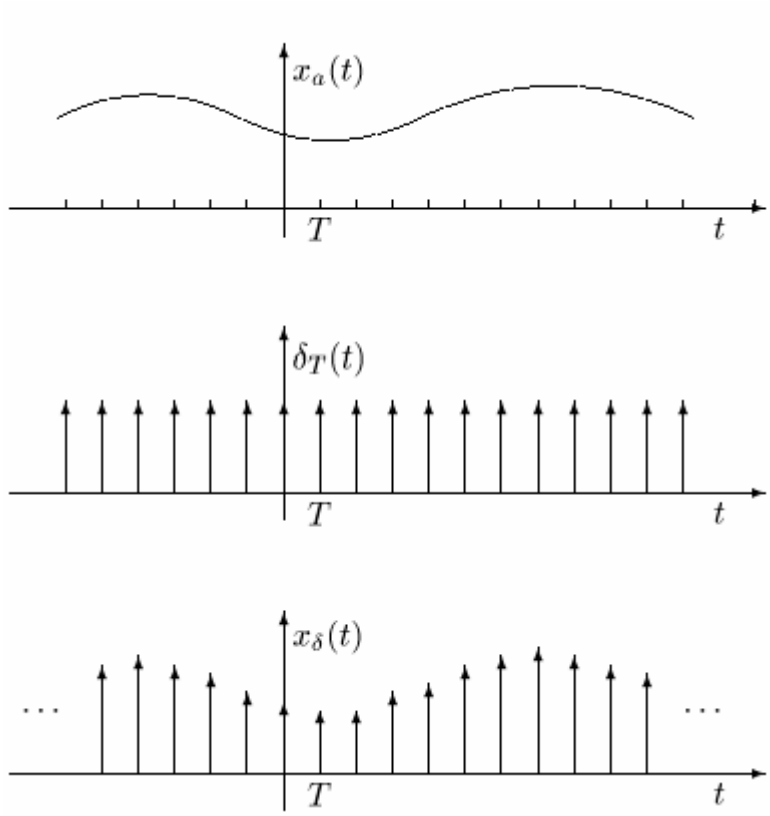


Figura 7: campionamento

Nella prima parte della figura è rappresentato un generico segnale analogico x(t); nella parte immediatamente inferiore il treno di impulsi e nell'ultima parte il segnale ottenuto moltiplicando i due, ossia il segnale campionato. Ricordiamo che l'impulso è un segnale che vale "1" ad un determinato istante di tempo e zero altrove. Il segnale ottenuto è ancora un treno di impulsi il cui inviluppo segue il segnale originale tempo continuo. A partire da questo segnale sarà possibile effettuare una quantizzazione, nel senso descritto prima, e quindi ottenere un segnale completamente digitale, oppure applicare la (4.1) e ricostruire esattamente il segnale.

Il segnale digitale calcolato, a meno di un errore di approssimazione di cui discuteremo in un apposito capitolo, può essere nuovamente riconvertito in un segnale analogico per tramite di un convertitore digitale-analogico (DAC), che agisce in maniera inversa rispetto all'ADC, anch'esso introducendo errori di approssimazione (l'associazione numero livello analogico è fatta con un livello analogico diverso da quello del segnale originale, essendo quest'ultimo non noto esattamente per effetto dell'errore di quantizzazione).

Ricapitolando, il teorema di Shannon afferma che è possibile rappresentare esaustivamente e senza errore un segnale analogico con una sequenza tempo-discreta; commettendo un errore di approssimazione, è possibile associare alla

sequenza tempo-discreta una sequenza completamente numerica che si presta ad essere “compresa” da un sistema di elaborazione a microprocessore, ottenendo un segnale memorizzabile nella RAM di un elaboratore. Usando pertanto un circuito che trasforma un segnale analogico in segnale digitale ed il suo inverso, e interponendo tra essi un elaboratore digitale, possiamo effettuare delle operazioni matematiche sul segnale digitale a mezzo di microprocessori, e dunque equivalentemente effettuare delle trasformazioni sul segnale analogico d’ingresso, trasformazioni che dipendono esclusivamente da un programma di calcolo. Le approssimazioni commesse nell’effettuare queste operazioni sono molteplici. In primis, come detto, nella stessa quantizzazione, e in generale in operazioni associate a circuiti “reali” (per esempio introduzione di rumore, errori di linearità, jitter, etc.); in secondo luogo errori ed approssimazioni dovute alla lunghezza di parola associata con il sistema a microprocessore utilizzato (cd. Rounding Error). Per finire, errori dovuti a cattivi algoritmi o cattive implementazioni degli stessi (cd. algoritmi mal condizionati). In apposite sezioni e capitoli si tratta dettagliatamente di queste problematiche, sino ad arrivare all’implementazione di routine per il calcolo “esatto” delle approssimazioni introdotte, quando ritenute significative.

3.4 Digital Signal Processor

Il processore “specializzato” per segnali digitali, o in termini anglosassoni “Digital Signal Processor”, nasce esplicitamente pensando ad una struttura elaborativa ove sia necessario effettuare un consistente numero di operazioni matematiche più o meno complesse in tempo reale; ossia, tipicamente, nel tempo che intercorre tra due campioni (digitali) adiacenti, in uscita da un convertitore analogico-digitale.

La struttura standard di un microprocessore si basa su la cd. Architettura di “von Neumann” la cui caratteristica principale è quella di leggere istruzioni e dati dalla RAM in tempi differenti (se non presente memoria cache). Questo significa che per effettuare (ad esempio) una somma tra due numeri saranno necessari accessi multipli alla stessa memoria (prelievo istruzione, prelievo dati), e dunque maggior tempo

elaborativo complessivo richiesto. Questo senza contare il tempo necessario all'esecuzione dell'istruzione stessa ed eventuale memorizzazione dei risultati. Ricordiamo che, in linea generale, le fasi relative all'esecuzione di una generica istruzione prevede tre passi: *fetch*, *decode* ed *execute*. La prima è relativa al "prelievo" dell'istruzione, la seconda alla sua "interpretazione" o "decodifica" e la terza alla sua esecuzione. La fase di fetch o prelievo deve essere pertanto ripetuta sequenzialmente per caricare istruzione ed operandi dalla stessa memoria. L'architettura utilizzata da un tipico DSP è invece di tipo Harvard, così chiamata perché sviluppata per la prima volta presso l'omonima università. Essa prevede la memorizzazione di istruzioni e dati in memorie fisicamente differenti ed accedute parallelamente in un singolo ciclo elaborativo. E dunque, rispetto all'architettura classica di von Neumann, in tempi teoricamente almeno dimezzati. Inoltre, l'architettura Harvard è dotata di diversi "address space" per dati e istruzioni, arrivando dunque a poter definire una diversa lunghezza di parola per l'indirizzamento di dati e istruzioni.

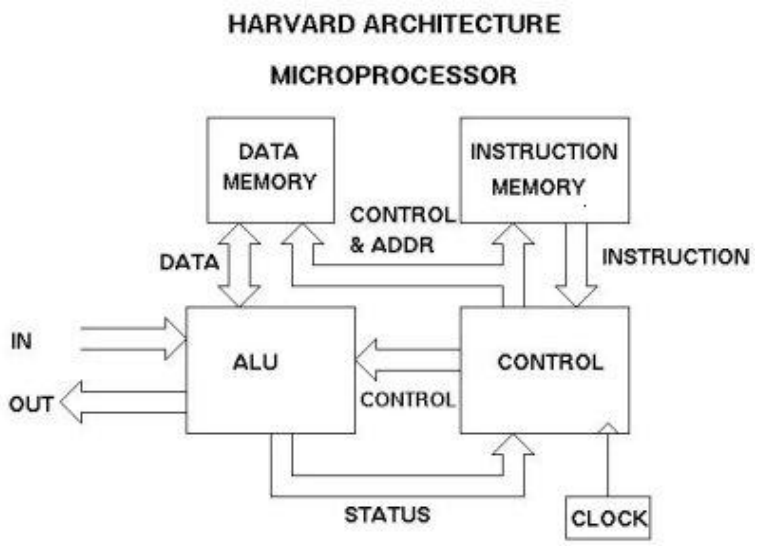


Figura 8: architettura Harvard

Nella figura 8 è riportato uno schema semplificato di un microprocessore ad architettura Harvard. In esso è messo in evidenza come la memoria dei dati sia fisicamente separata dalla memoria delle istruzioni e acceduta da bus distinti (e dunque potenzialmente al medesimo istante) oltre a poter essere indirizzata con

"risoluzioni" differenti. In generale, la separazione tra le due aree dati e istruzioni può estendersi anche oltre; generalmente infatti le istruzioni possono essere memorizzate direttamente in memorie non volatili, visto che durante l'esecuzione del programma non è generalmente richiesto che esse debbano essere scritte oltre che lette; cosa che ovviamente è invece normalmente possibile per i dati. Questo giustifica dunque in generale caratteristiche completamente diverse per i due address space, che possono spingersi anche a caratteristiche più dettagliate, come tempi di accesso, larghezza di parola, tecnologie implementative. Per chiarire ulteriormente le cose, si riporta nella figura successiva la sequenza delle fasi di esecuzione di tre istruzioni contemporanee [46], temporalmente scandite dal segnale di clock per l'architettura Neumann (fig. 9 bis) e Harvard (fig. 9 ter).

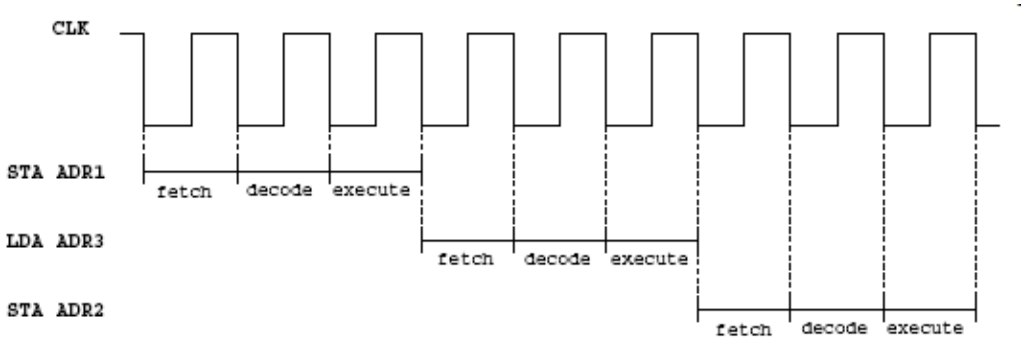


Figura 9 bis: esecuzione di tre istruzioni nell'architettura von Neumann

Ed invece per l'architettura Harvard:

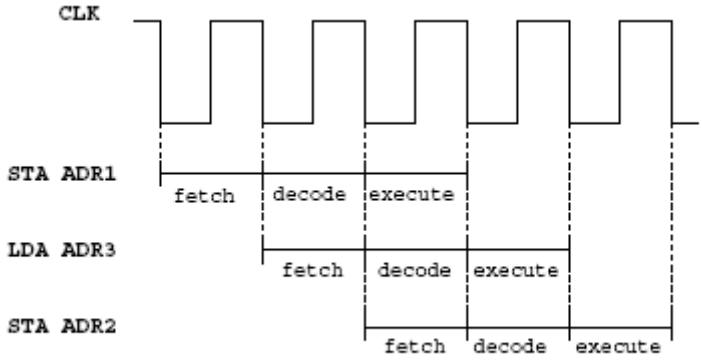


Figura 9 ter: esecuzione di tre istruzioni nell'architettura Harvard

Come è facile osservare dalle figure precedenti il vantaggio dell'architettura Harvard è immediatamente evidente. Per contro, gli svantaggi di quest'ultima sono il costo maggiore, la maggiore difficoltà di rilocazione delle due aree di memoria distinte, maggiori vincoli nell'esecuzione di istruzioni parallele, maggiore complessità nello scrivere programmi. Invero questi svantaggi sono quasi del tutto compensati dalla particolare efficienza dei moderni compilatori.

Nella realtà, le cose si sono evolute in maniera estremamente più complessa, coesistendo attualmente in microprocessori commerciali entrambe queste tecnologie e architetture (più molte altre, per esempio RISC) che mirano essenzialmente ad un abbattimento dei tempi di calcolo. Invero, esistono numerose soluzioni "orientate" alle applicazioni cui sono destinati (microprocessori dedicati all'elaborazione di segnali video, audio, applicazioni software standard, archiviazione dati, etc.). Molti di essi presentano come caratteristica la coesistenza di molteplici microprocessori di diversa tipologia fisicamente integrati all'interno di uno stesso chip; per esempio l'oramai non più giovane C6000 della Texas Instruments è costituito da ben quattro DSP e un RISC, e nasce per gestire flussi video, sebbene utilizzato per molte altre applicazioni (esempio in centrali telefoniche Ericsson). Oppure, diverse soluzioni architetturali convivono nel medesimo microprocessore, come per esempio l'architettura Harvard integrata con molteplici unità elaborative parallele.

Il concetto chiave che si vuole porre in risalto in questa sede è che un microprocessore di tipo DSP presenta un'architettura che lo specializza per il trattamento "veloce" di dati che sono normalmente rappresentativi di segnali analogici e che necessitano di elaborazioni in tempi comparabili con la velocità evolutiva del processo sotto osservazione (si pensi a un segnale cui si vuole applicare un filtro e che questo debba avvenire in tempo reale, ossia per esempio in un elettrocardiografo mentre si stanno rilevando i segnali dalle varie derivazioni standard).

Come accennato nell'introduzione nel primo capitolo, ben presto la tecnologia relativa ai personal computer ha consentito di utilizzare al posto dei DSP dei normali PC, anche se solo per determinate classi di applicazioni; alcuni autori sostengono che il DSP sia stato soppiantato dall'uso dei personal computer, e questo è certamente vero per molteplici applicazioni, ma è parimenti vero che esistono campi in cui è necessario utilizzare dispositivi dedicati e integrati nello stesso sistema da gestire; si pensi ad esempio a normali centraline (ECU, Electronic Control Unit) utilizzate nelle moderne automobili ad iniezione elettronica, oppure a strumenti di misura "stand-alone" o computer Avionici.

In ogni caso la possibilità di usare (per talune applicazioni) i normali PC al posto dei DSP ha consentito l'utilizzo di algoritmi un tempo dominio esclusivo dei DSP stessi in programmi Windows standard (ed anche Linux o Mac) consentendo dunque (anche) la realizzazione di strumentazione virtuale a basso costo nel senso più volte specificato nel corso della presente trattazione. Ossia utilizzando il PC come un "pezzo" significativo dello strumento virtuale che oltre ad essere usato per la visualizzazione dei dati collabora fattivamente all'elaborazione numerica dei segnali.

3.5 Convertitori A/D e D/A

I convertitori analogico-digitali e digitali-analogici consentono dunque la pratica applicazione del teorema del campionamento e sono dunque fondamentali in tutte quelle applicazioni dove sia necessario "trasformare" segnali analogici in segnali memorizzati in RAM, e dunque utilizzabili da un elaboratore. Su di essi esiste vasta letteratura [40] e ad essa si rimanda per l'aspetto circuitale. Quello che si ritiene invece di porre in evidenza in questo paragrafo sono le approssimazioni introdotte da essi; ossia tutti quegli elementi che introducono errori (o meglio incertezza) nell'operazione di conversione, e dunque riuscire a quantificare quanto il processo di conversione reale si discosta dalla conversione teorica prevista dal teorema di Shannon Nyquist. Intanto una curiosità; sovente, i convertitori A/D utilizzati nelle apparecchiature audio (per esempio lettori CD) presentano una frequenza di

campionamento di 44100 kHz. Essa consente una banda passante teorica di 22050 kHz (la banda audio è generalmente considerata tale sino a 20kHz) dunque si potrebbe pensare (e taluni autori lo confermano) che la scelta di porre un limite superiore della banda passante maggiore dei limiti reali della banda audio sia dovuto a precise scelte tecniche; in particolare per consentire al filtro anti-aliasing di lavorare al meglio. Infatti, il filtro anti-aliasing è un filtro passa basso implementato in hardware, e come tale presenta una caratteristica non ideale, ossia per esempio una banda di transizione graduale e non istantanea. In altre parole, la risposta in ampiezza non è assolutamente un gradino ma piuttosto una retta con pendenza finita, e la risposta in fase non è anch'essa ideale. Iniziando a "tagliare" intorno ai venti kHz, e garantendo comunque un contenuto armonico originale rigorosamente limitato a 20 kHz (filtrando il segnale prima della conversione) pur campionando con una banda passante teorica di 22050 Hz si è sicuri che il filtro lavori in maniera ottimale e le frequenze sino a 20 kHz siano riprodotte con distorsioni di fase e ampiezza che risentono molto meno della non idealità del filtro. In altre parole, si inizia a "tagliare" a 20 kHz un segnale di per se già privo di componenti al di sopra di questa frequenza ma campionato in maniera tale da avere una banda potenziale più estesa, sicchè il comportamento del filtro sopra i 20 kHz è trascurabile.

Questo fatto è decisamente vero, ma la reale ragione è (anche) un'altra; le prime apparecchiature digitali per le registrazioni di segnali audio sfruttavano dei registratori a larga banda commerciali normalmente disponibili ai tempi, ossia i registratori video (analogici). In particolare, lo standard video americano NTSC si basava su 30 frame di 525 linee al secondo, di cui 490 effettivamente utilizzabili per la visualizzazione. Lo "spazio" per la memorizzazione di una linea venne utilizzato per memorizzare in sicurezza 3 campioni audio.

E dunque, avendo 30 frame al secondo e 490 linee per frame, ognuna delle quali contenente 3 campioni si ha:

$$Campioni = 30*490*3 = 44100\ (Hz)$$

Ed ecco spiegato il motivo del numero 44100. Lo stesso si ottiene per lo standard europeo PAL, il quale prevede 25 frame di 625 linee di cui 588 utilizzabili. Con

analogo calcolo si può facilmente verificare che si ottiene un numero di campioni al secondo pari a 44100.

3.5.1 Il convertitore A/D, definizioni principali

Enunciamo brevemente i parametri principali che caratterizzano un convertitore Analogico Digitale. Esso è caratterizzato intanto dal "numero di bit" ossia dal numero di cifre binarie con cui viene convertito il valore di tensione d'ingresso. Esso è in pratica corrispondente all'ampiezza in bit del numero intero utilizzato per la rappresentazione del valore approssimato del valore di tensione. Dato questo parametro (valori tipici 8, 16, 20, 24 bit) è possibile risalire al numero di livelli possibili di tensione rappresentabile. Per esempio un numero di cifre binarie n=8 bit implica 2^n livelli differenti ossia 256 differenti livelli di tensione.

Un'altro parametro importante è la "*tensione di fondo scala*" (V_{fs}) definita come il doppio di quella tensione in ingresso il cui valore è rappresentato da un valore numerico con il bit più significativo posto a 1. In parole più semplici, è il valore di tensione in ingresso massimo accettabile.

Dato il parametro "n" e "V_{fs}" è immediato ricavare un altro parametro noto come "risoluzione" e dato dal rapporto tra la seconda e la prima, ossia:

$$R = \frac{V_{fs}}{2^n} \tag{4.2}$$

Che rappresenta dunque la minima tensione rappresentabile od il "passo" di tensione globalmente rappresentabile con quel convertitore (e dunque la risoluzione).

Date queste principali ed importanti definizioni, analizzeremo in un successivo capitolo le principali cause di incertezza introdotte da un convertitore analogico digitale connesse con tutta l'elettronica presente sulla scheda di acquisizione. Infatti la presenza di circuiti di condizionamento, amplificazione e attenuazione, oltre alla presenza di circuiti aggiuntivi di supporto (generatori di tensioni di riferimento, generatori di clock) così come anche lo stesso layout del circuito (il circuito stampato) e l'eventuale convertitore AC-DC (il comune alimentatore) possono dare

un contributo (anche significativo) all'incertezza introdotta dalla scheda di acquisizione.

3.5.2 Cause di incertezza dei convertitori A/D

Nel prossimo capitolo descriveremo le cause d'incertezza più significative connesse con l'uso dei convertitori A/D, le quali si possono riassumere sin da ora nel seguente elenco:

- Errore di Quantizzazione
- Errore di Offset
- Errore di Guadagno
- Errore di linearità (differenziale ed integrale)
- Omissione di codice
- Jitter
- Noise

Come detto nelle note conclusive del precedente paragrafo, questi errori possono essere considerati come generati non solo dal convertitore in se (che comunque già di per se contribuisce in maniera significativa) ma in generale da tutta la scheda di acquisizione (compreso il cablaggio stesso).

Capitolo 4

Cause di incertezza specifiche e loro valutazione

4.1 Introduzione

In questo capitolo si vuole porre l'attenzione sulle cause di incertezza presenti in uno strumento virtuale "PC-based", e parimenti darne una valutazione quantitativa. Come verrà dimostrato più avanti, le cause d'incertezza più significative di Visual Analyser sono concentrate nella scheda di acquisizione. In particolare esse vengono in larga misura determinate dal convertitore Analogico Digitale (e circuitazione di supporto ad esso associata), sulla cui funzione ci siamo soffermati nel precedente capitolo, dando anche le definizioni dei principali parametri d'interesse. In questo capitolo descriveremo le incertezze introdotte da questo componente e parimenti un semplice modello matematico che ci consentirà di tenere conto di esse nel nostro strumento virtuale, e/o tramite simulazione Monte Carlo. Come già accennato, le cause di incertezza che descriveremo sono senza alcun dubbio generate dal convertitore A/D ma decisamente peggiorate dall'accoppiamento "sinergico" con tutte le altre componenti circuitali di una generica scheda di acquisizione, e dunque sono da considerare come "globali" alla scheda di acquisizione, anche laddove, per semplicità di linguaggio, venga fatto riferimento al solo convertitore A/D. Ancora, si vuole far notare come la "caratterizzazione metrologica" della scheda, intesa nel senso spiegato nel capitolo VIII, sia fisicamente realizzabile in laboratori specializzati che effettuano analisi statistiche su un campione statisticamente

rappresentativo della scheda e in tutte le condizioni d'uso possibili. In particolare, un'analisi accurata dovrebbe comprendere i test di ripetibilità e riproducibilità. Queste considerazioni verranno riprese nel capitolo relativo alla progettazione della scheda di acquisizione dati progettata per lo strumento "ZRLC" (Impedenzimetro) la cui caratterizzazione metrologica non è stata effettuata per mancanza di mezzi tecnici; né è possibile derivare essa dalle caratteristiche dei costruttori fornite per i singoli componenti, sia perché spesso non note, ma in particolar modo perché la scheda, nel suo insieme, ha un comportamento diverso dalla somma dei comportamenti dei singoli componenti (dal punto di vista dell'interazione delle cause d'incertezza).

Parimenti, la possibilità che il programma fornisce di impostare i parametri relativi alla caratterizzazione metrologica dello strumento, fa si che esso sia di massima generalità e che dunque, quando in possesso di tali parametri (per esempio utilizzando altre schede di acquisizione commerciali e caratterizzate) si possano ottenere risultati corretti.

4.2 Cause di incertezza dei convertitori A/D

Il convertitore A/D è un dispositivo capace di fornire in uscita un valore numerico proporzionale alla tensione del segnale d'ingresso, ma "a meno della sua risoluzione"; ossia introducendo un'incertezza che dipende dal numero di bit utilizzati per rappresentare numericamente il segnale.

Le considerazioni si basano su [33, 34]. La figura [9] mette in evidenza che cosa si intenda per quantizzazione e la figura [10] mette in evidenza l'andamento e l'ampiezza dell'incertezza associata alla quantizzazione stessa.

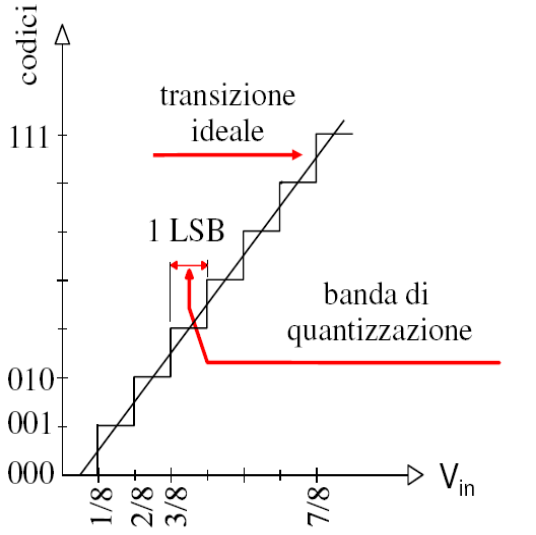


Figura 9: processo di quantizzazione

In questa figura si prende ad esempio un convertitore A/D con n=3 bit; in ascissa è riportato il valore della tensione d'ingresso ed in ordinata il codice numerico assegnato al valore di tensione V_{in}. La caratteristica "di trasferimento" mette in evidenza come ad un "intervallo di tensione" d'ingresso (ossia ad un insieme di valori di tensione) corrisponda il medesimo codice d'uscita, e dunque si introduca evidentemente una incertezza sul valore digitalizzato. La figura 10 mette meglio in evidenza il fenomeno noto come "errore di quantizzazione" ε.

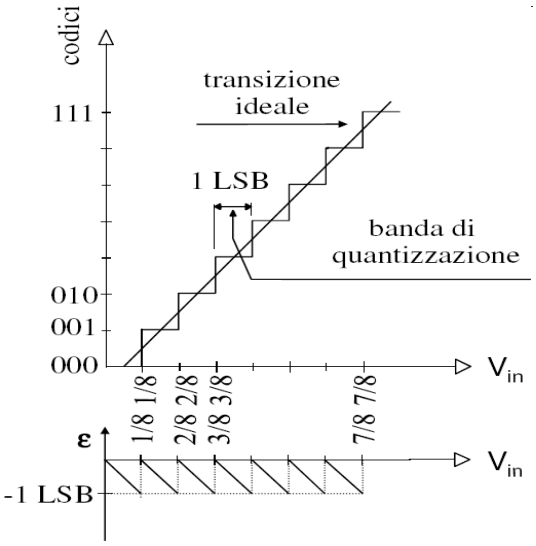


Figura 10: incertezza di quantizzazione (anche detto errore di quantizzazione)

Esso è dunque pari *al più* 1 LSB (Least Significant Bit, bit meno significativo). Generalmente è preferibile aggiungere un offset di ½ LSB per avere un'incertezza che varia tra ±½ LSB.

Accanto all'incertezza "classica" di quantizzazione (essenzialmente un'incertezza di tipo "rounding error", ossia da "arrotondamento") si accompagnano una nutrita schiera di altre cause, più o meno gravi e quantificabili, che derivano tutte essenzialmente dallo scostamento della forma della funzione di trasferimento da quella ideale rappresentata dalla retta della figura 9. Per esse manterremo la denominazione di "errore" che generalmente si trova in letteratura, sebbene esse siano da intendere nel senso moderno di "incertezza" come esplicitamente chiarito nel capitolo II e spesso ribadito in più punti nel presente lavoro. Essi sono essenzialmente i seguenti, oltre all'errore di quantizzazione:

- Errore di offset
- Errore di guadagno
- Errore di linearità (differenziale ed integrale)
- Omissione di codice
- Jitter
- Noise

Da notare, e metteremo ben in evidenza la cosa in molti punti della trattazione, che alcuni di essi si possono considerare "sistematici" ed altri "casuali". In tal senso, gli errori sistematici possono essere in parte compensati, sia a livello hardware che software. Quelli casuali poi, qualora si effettui una determinazione dell'incertezza complessiva del sistema di misura che tiene conto dei contributi A e B (cfr. cap. II), verranno automaticamente compresi nella procedura di determinazione dell'incertezza di tipo A e dunque non dovranno essere esplicitamente tenuti di conto nel calcolo dell'incertezza di tipo B. Parimenti, in generale, alcuni errori sistematici che non è possibile compensare in alcun modo (oppure per libera scelta) verranno considerati casuali e "modellizzati" come variabili casuali dotate di opportuna distribuzione.

Errore di offset

L'errore di OFFSET è rappresentato nella figura 11; come è facile evincere dal grafico, esso è dato da uno scostamento della caratteristica ideale sotto forma di traslazione della stessa di un fattore costante; nella figura esso si discosta dalla caratteristica ideale (che è una retta con coefficiente angolare di 45 gradi) di un "offset" negativo che sposta la retta in basso di una determinata quantità, detta appunto "errore di offset".

Tutti i codici prodotti sono pertanto modificati nella stessa misura; essa è compensabile mediante calibrazione.

Taluni convertitori prevedono una apposito "pin" di calibrazione che consente di applicare al circuito interno una tensione di offset. Questo errore, o meglio questa causa d'incertezza, ricade dunque nella categoria degli errori sistematici.

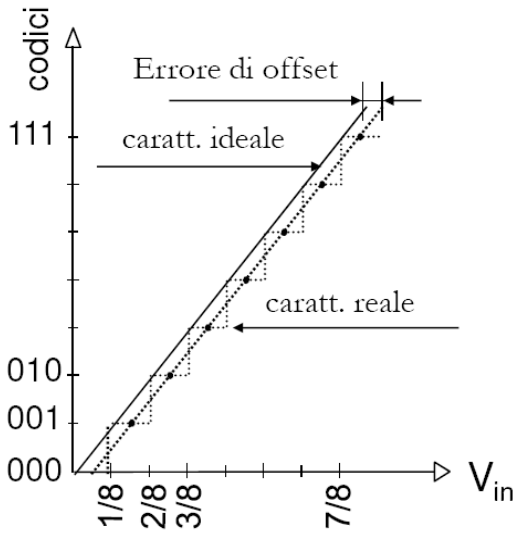


Figura 11: errore di offset

Errore di guadagno

In maniera simile all'errore di OFFSET l'errore di GUADAGNO altera la produzione di codici rispetto alla situazione ideale, come mostrato nella fig. 12.

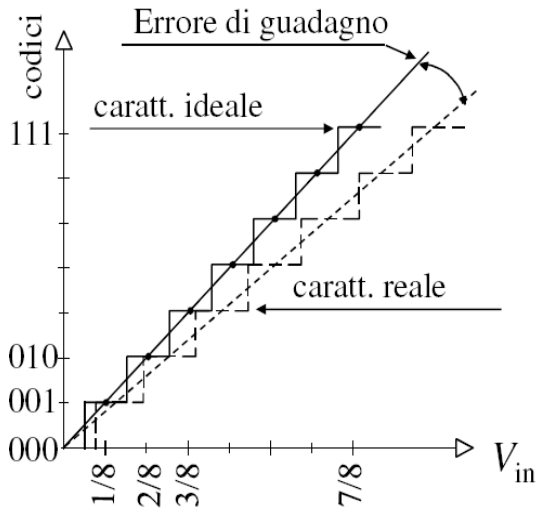


Figura 12: errore di guadagno

In questo secondo caso la retta parte dall'origine, ma ha una diversa pendenza rispetto a quella ideale. In questo caso tutti i codici "prodotti" sono *moltiplicati* per la stessa quantità (nel caso precedente ad ogni codice aveva *sommata* una medesima quantità).

Errore di linearità

Un altro errore, molto più "generale" dei precedenti è l'errore di linearità, di cui distingueremo le due varianti "integrale" e "differenziale". Si veda la figura 13.

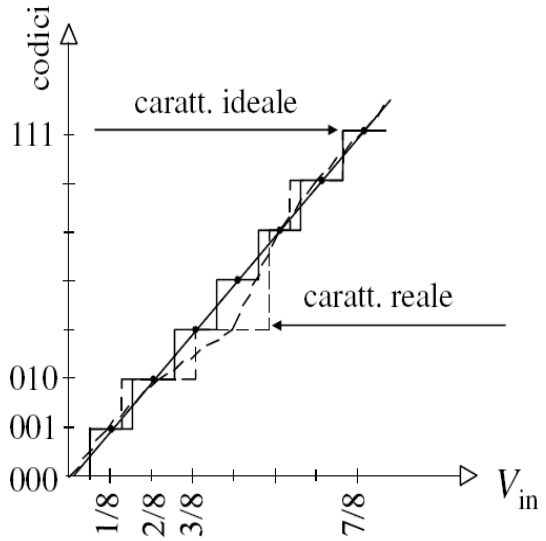


Figura 13: errore di linearità

Esso dunque è relativo ad uno scostamento che può essere visto come una sorta di generalizzazione dei due precedenti errori; semplicemente, la "funzione di trasferimento" non è più una retta ma una curva qualsiasi "contenuta" tra due rette parallele alla funzione ideale; ed inoltre strettamente crescente.

Detta in altre parole, l'ampiezza delle bande di quantizzazione non è più costante e pari ad un LSB, ma può variare ad ogni passo.

L'errore di non linearità può essere descritto in due modi, dando origine a due differenti definizioni:

- DNL (Differential Non Linearity) ossia errore di linearità differenziale. Esso è definito come la differenza tra la larghezza della singola banda effettiva e quella ideale, ossia 1 LSB (fig. 14 riportata da materiale National [33]);
- INL (Integral Non Linearity) ossia errore di linearità differenziale, è definita come la massima deviazione della funzione di trasferimento reale (congiungente i punti centrali dei gradini reali di quantizzazione) dalla retta ideale (v. fig. 15). Un'altra definizione è: il massimo scostamento della "migliore retta interpretativa" (determinata per es. con i minimi quadrati) da quella ideale, ed un'ultima è: deviazione della linea caratteristica di trasferimento dalla retta ideale al termine della caratteristica di trasferimento. Molti autori esortano a distinguere attentamente a quale delle tre ci si riferisce quando si interpretano i parametri di un convertitore [34] forniti dal costruttore; nel corso della presente trattazione faremo spesso riferimento alla prima definizione.

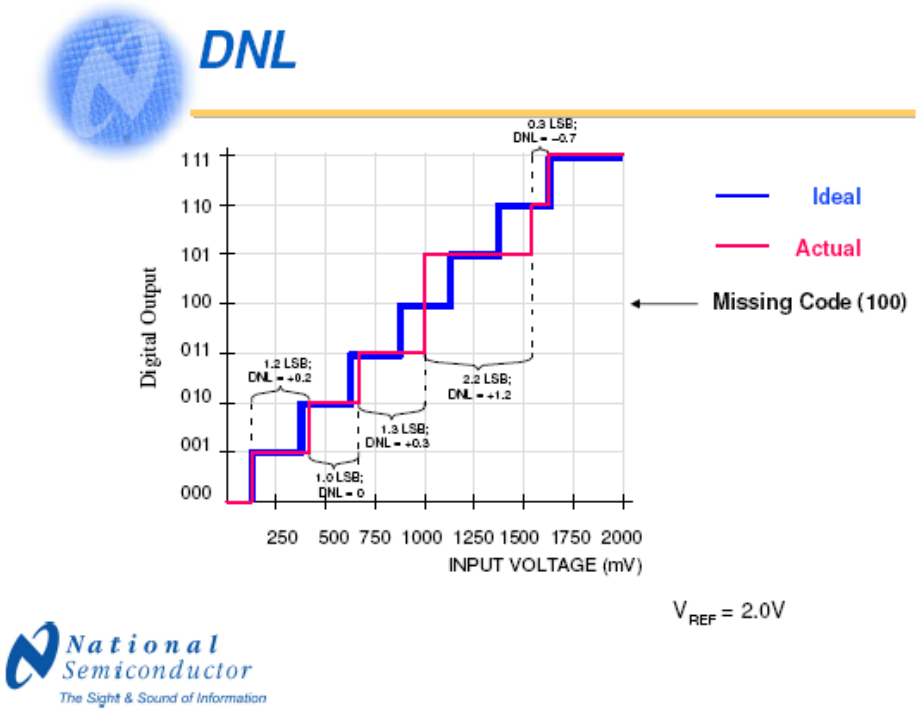


Figura 14: linearità differenziale

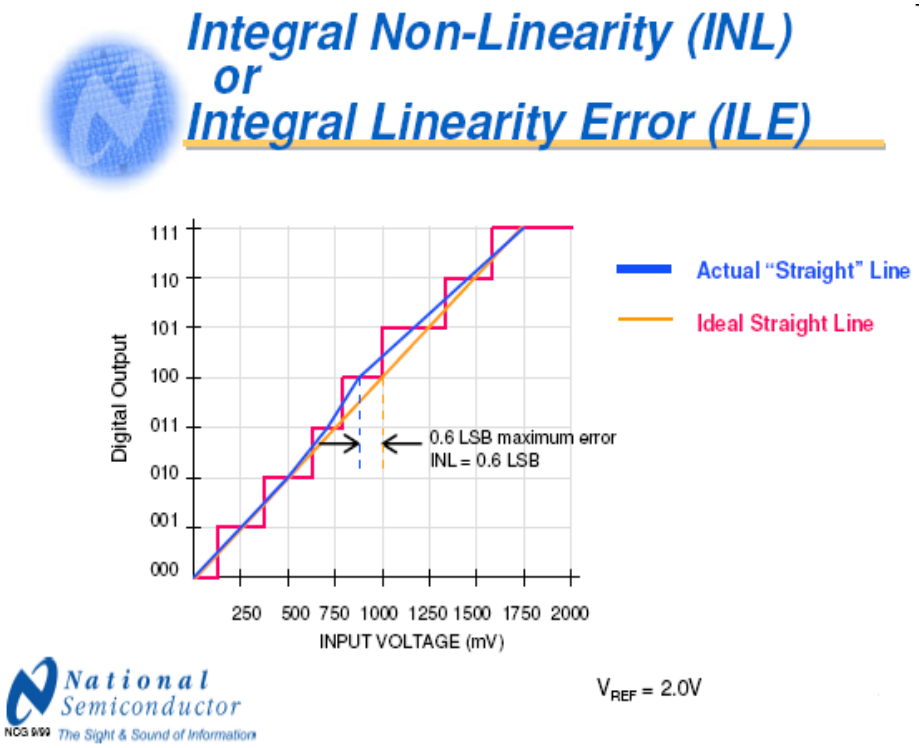


Figura 15: linearità integrale

I due indici testé definiti (linearità differenziale ed integrale) danno informazioni utili sul comportamento del convertitore e l'incertezza da esso introdotta; in particolare un elevato valore della linearità integrale indica che gli errori di non linearità deformano la caratteristica di trasferimento nella stessa direzione; e un basso valore di linearità integrale, accompagnato ad un elevato valore di linearità differenziale indicano che gli errori di non linearità hanno valore elevato e verso opposto nei differenti punti della caratteristica [32, 33]. La INL può essere misurata in due modi; uno detto "End Point" ed un secondo detto "Best Fit". Non entriamo in dettaglio in questi metodi, perché esulano dagli scopi del presente paragrafo e della trattazione in generale.

Omissione di codice

L'errore di OMISSIONE DI CODICE è direttamente legato agli errori di linearità, ed in particolare della linearità integrale. Infatti, se per una banda l'errore di linearità supera 1 LSB si ha la perdita di un codice, ossia un codice associato ad un livello di tensione NON viene generato (v. fig. 16). Questo errore non verrà esplicitamente considerato perché frutto di una causa il cui solo valore eccessivo ne determina l'esistenza; in generale faremo conto di riuscire a "contenere" l'errore di linearità integrale entro limiti moderati che non causano omissione di codice.

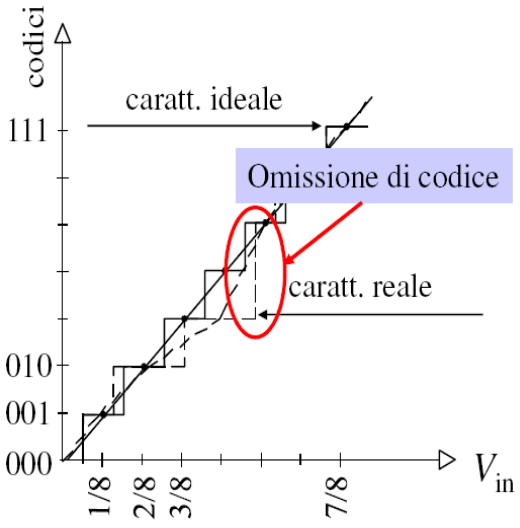


Figura 16: omissione di codice

Errore di Jitter

Veniamo ora ad un errore particolarmente importante, detto "JITTER". Esso è direttamente legato alla bontà del segnale di clock utilizzato per pilotare il circuito di conversione A/D (e D/A). Il segnale può presentare delle variazioni casuali nella sua frequenza, fase e duty-cycle, determinando un'incertezza dell'istante effettivo in cui si ha il campionamento del segnale; esso infatti avviene in corrispondenza di uno dei due fronti del clock. Le caratteristiche matematiche del modello usato per caratterizzare questo errore sono state trattate esaustivamente in VI. La figura 17 mette in evidenza la genesi dell'errore di Jitter (intesa come variazione indesiderata delle caratteristiche del segnale di clock) e la figura 18 l'effetto del clock sull'istante di campionamento. Nella medesima figura viene riportato il calcolo del massimo valore che il Jitter può avere sul codice emesso per un segnale sinusoidale (le variazioni infatti dipendono dal segnale da campionare, ossia dalla sua pendenza in quell'istante ed in un suo intorno scelto opportunamente). Queste variazioni possono essere causate dal circuito che genera il clock, da un cattivo layout (circuito stampato), interferenza con altri segnali oppure disadattamenti d'impedenza.

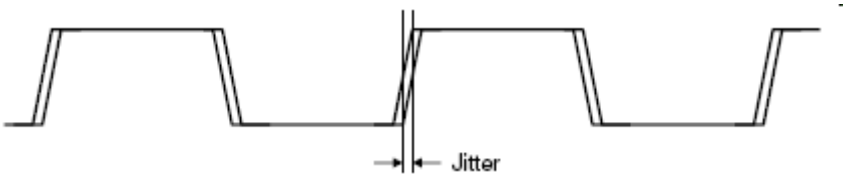
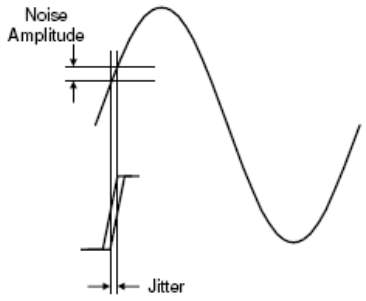


Figura 17: jitter



$$\text{Max Jitter} = V_{IN} / (2^{(n+1)} \ \pi \ V_{FS} \ f_{IN})$$

Figura 18: effetto del segnale di clock sull'istante di campionamento e calcolo del massimo valore

Il Dithering

Per finire, citiamo una tecnica che può essere utilizzata per cercare di compensare gli errori di linearità; essa è nota sotto il nome di "Dithering" e contribuisce a diminuire in generale tutti quegli errori dovuti a campionamenti e quantizzazioni dei segnali. Il termine "Dither" deriva dal termine anglosassone "didderen" che significa tremare, e consta essenzialmente nell'aggiungere al segnale da campionare una certa quantità di rumore.

Risale ai tempi della seconda guerra mondiale la scoperta che vibrazioni aggiuntive somministrate a computer ad ingranaggi, usati su bombardieri americani, rendevano gli stessi più accurati e precisi nelle operazioni di calcolo. La spiegazione fu presto chiara; spiegazione che ha dato origine alla tecnica appunto detta di "Dithering" oggigiorno applicata a segnali che devono essere trasformati in segnali numerici. Prendiamo ad esempio, per semplicità, l'errore di quantizzazione che si ottiene campionando e quantizzando un segnale sinusoidale. Il processo di campionamento e quantizzazione introduce, tra i vari errori, anche una ulteriore forma di errore dovuto alla ciclicità dell'errore di quantizzazione.

Il che si traduce nell'aggiunta di ulteriori armoniche, ossia distorsione del segnale; la ciclicità dell'errore è dovuta al fatto che l'errore di quantizzazione presenta la stessa ampiezza in corrispondenza dei medesimi valori assunti dal segnale d'ingresso.

E dunque se in generale è vero che l'errore varia continuamente e casualmente al variare del segnale d'ingresso, e altrettanto vero che se quest'ultimo è a sua volta periodico esso si ripeterà con la medesima ampiezza in corrispondenza dei medesimi valori dell'ingresso.

L'errore di quantizzazione non sarà quindi completamente casuale, ma si ripeterà ciclicamente, dando origine pertanto ad un nuovo segnale. In altre parole, a titolo esemplificativo, prendiamo il segnale il cui spettro è rappresentato in figura 19 (fonte [35]), relativo ad un segnale sinusoidale a 500 Hz e quantizzato con un convertitore a 16 bit di risoluzione.

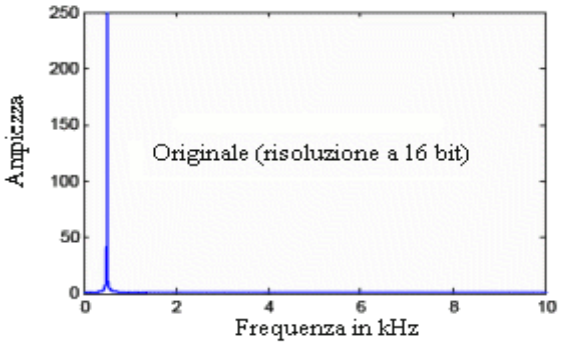


Figura 19: spettro di un segnale sinusoidale quantizzato a 16 bit

Per rendere più evidenti i segnali spuri introdotti da una quantizzazione spinta, effettuiamo un troncamento a 6 bit del segnale; il segnale risultante è rappresentato in figura 20.

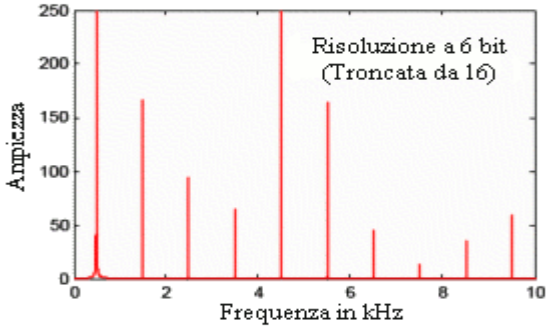


Figura 20: segnale troncato a 6 bit

A questo punto aggiungiamo al segnale originale un segnale casuale (rumore), ed otteniamo:

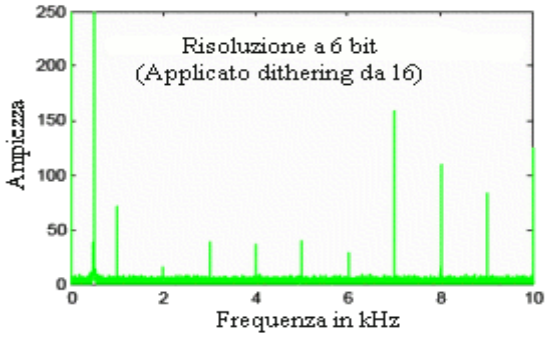


Figura 21: segnale di figura precedente con Dithering

Il Dithering, come detto, consiste nell'aggiunta al segnale originale di un rumore casuale (con determinate caratteristiche di distribuzione ed ampiezza) che rende altrettanto casuale l'errore di arrotondamento eliminando in buona misura le armoniche aggiuntive dovute alla periodicità dell'errore; e vale per tutte quelle forme d'errore che provocano errori periodici, come la non linearità differenziale. Il prezzo da pagare è la comparsa di un rumore aggiuntivo sovrapposto al segnale originale, che tuttavia si può pensare di eliminare con varie tecniche (es. filtraggio).

4.3 Modellizzazione delle incertezze

Le cause d'incertezza messe in evidenza nel paragrafo precedente devono essere quantificate per poter essere valutate. Lo strumento virtuale proposto effettua un calcolo della incertezza complessiva in tempo reale, che tiene conto di queste molteplici cause (per ogni strumento implementato e naturalmente per ogni scheda di acquisizione usata); è inoltre possibile effettuare un'analisi Monte Carlo, utilizzando i medesimi modelli matematici definiti per tutte queste cause di incertezza ma in senso opposto, ossia per la generazione simulata delle variabili d'ingresso e incertezze associate. L'analisi Monte Carlo verrà effettuata con meccanismi di generazione interna delle grandezze (ossia con apposite routine che "sintetizzano" le variabili da misurare opportunamente "perturbate") e facendo uso delle medesime procedure per il calcolo in tempo reale dell'incertezza complessiva. In altre parole, è possibile attivare un generatore interno che "alimenta" lo strumento misura con grandezze elettriche simulate in accordo ai parametri che rappresentano la scheda di acquisizione.

Si noti che, si ribadisce, non si sta parlando di una generazione di segnali tramite hardware, ma di segnali fittizi generati via software a livello esclusivamente numerico, e il cui andamento segue il modello matematico (stocastico) ricavato dai parametri che "caratterizzano metrologicamente" la scheda reale di acquisizione.

Per ogni incertezza presa in considerazione è infatti necessario definire un modello matematico, il quale essenzialmente si limita a determinare le caratteristiche della

variabile aleatoria associata alla causa di incertezza, sia che essa sia relativa ad un errore sistematico che casuale. Infatti, gli errori sistematici che possono essere eliminati sono appunto (almeno in gran parte) non più presenti, mentre quelli che non possono essere compensati vengono comunque modellizzati e tenuti di conto tramite una variabile casuale. E quindi, in definitiva, si tratta di definire il tipo di distribuzione ed i parametri ad essa associati (media, varianza, deviazione standard, etc.).

Da tenere in considerazione che alcune delle incertezze qui considerate sono invero automaticamente stimate dalla procedura che calcola le incertezze di tipo "A", ovvero l'incertezza statistica (qualora calcolata per lo strumento specifico: allo stato attuale il calcolo è facoltativamente implementato quasi per tutti). Come è noto (v. capitolo II) questo calcolo consente di valutare le incertezze dovute ad errori casuali; ossia relative a quelle incertezze "catturate" da un processo di analisi statistica che valuta media e deviazione standard su misure ripetute.

Le misure ripetute, data la tecnica utilizzata dal programma per acquisire i dati, sono facilmente ed automaticamente implementabili (è il default). Infatti, ogni strumento virtuale implementato opera sul segnale acquisito a blocchi di "n" campioni, blocchi continuamente rinnovati ad intervalli di tempo predefiniti (i buffer vengono passati continuamente e senza interruzioni ai thread allo scopo preposti dei vari strumenti implementati). Di conseguenza, la misura viene effettuata sui campioni del buffer corrente, ossia sulla porzione di segnale "contenuta" nel buffer, e nuovamente ripetuta ad ogni nuovo buffer. Il processo di misure ripetute deve essere interrotto manualmente. In tal senso, la misura ripetuta lo è per default, ossia in un certo senso "viene a costo zero".

Date le cause d'incertezza analizzate nei paragrafi precedenti, almeno per esse si può procedere ad una valutazione di tipo B e di tipo A. Per quella di tipo A rimandiamo ad un apposito capitolo/paragrafo, per quelle di tipo B, ossia ottenibili con metodi diversi dall'analisi statistica (basate per esempio sui parametri che fornisce il costruttore) essa dipende dal tipo di misura effettuata. Come visto nel precedente capitolo, le incertezze inerenti le varie variabili presenti nella relazione di tipo

$Y=f(X_1, X_2, ..., X_n)$ (nel caso, relativamente diffuso, che si stia effettuando una misura di tipo indiretto, e dunque "modellizzata" con una relazione di questo tipo che per semplicità chiameremo "relazione costitutiva") si "propagano" sino ad arrivare alla variabile *Y* (la misura) in maniera che dunque dipende dalla relazione costitutiva in se. Per esempio, nel caso dello strumento "impedenzimetro" presentato in un apposito capitolo di questo lavoro, la relazione che lega la grandezza sotto misura è perfettamente definita (v. cap. VIII), ed i vari contributi dell'incertezza (Jitter, Gain, etc.) si propagano sino alla misura finale (impedenza, capacità, induttanza, etc.) e sono quantificabili in maniera relativamente semplice calcolando i "coefficienti di sensibilità" e tenendo di conto eventuali correlazioni (legge di propagazione delle incertezze). Entrambe le quantità (coeff. di sensibilità ed eventualmente di correlazione) possono essere calcolati dalla relazione costitutiva.

Lo stesso procedimento si può applicare a tutti gli strumenti di misura la cui relazione costitutiva è nota. Anche nel caso dell'analizzatore di spettro essa è perfettamente quantificabile per tramite della legge di propagazione delle incertezze, il cui calcolo è stato effettuato (per alcune grandezze) nel cap. VIII.

Per contro si osservi che alcuni degli strumenti implementati godono di particolari caratteristiche implementative che rendono più conveniente l'applicazione di altre metodologie di calcolo, che verranno descritte caso per caso (ossia strumento per strumento; si veda come è stato p. es. affrontato il caso del calcolo dell'incertezza dell'impedenzimetro, ed in particolare di come si è tenuto di conto del contributo dato dalla resistenza di riferimento).

Quello che dunque metteremo in evidenza in questi ultimi paragrafi è la scelta di come caratterizzare le variabili aleatorie associate ad ogni causa di errore presente nella scheda di acquisizione, e come queste possono venir utilizzate per costruire una "variabile sintetica" che rappresenti fedelmente la variabile reale e che eventualmente possa venir utilizzata per impostare una analisi automatica di tipo Monte Carlo.

4.5.1 Errore di quantizzazione

Come accennato in altre sezioni, questo errore viene a generarsi per effetto dell'associazione effettuata tra i numeri (a cardinalità finita) disponibili per rappresentare una grandezza analogica, che per sua natura é ad infiniti valori. Si veda a tal proposito la figura 9 in questo stesso capitolo. Se il convertitore analogico-digitale è a "k" bit, mentre l'intervallo analogico di tensione del convertitore è di Δ_v Volt, allora l'intervallo di quantizzazione è pari a:

$$q = \frac{\Delta v}{2^k} \tag{4.1}$$

Ossia, la più piccola variazione che possiamo apprezzare con il nostro convertitore è di "*q*" Volt, che costituisce dunque anche il valore rappresentato dal bit meno significativo del numero a "k" bit che rappresenta la grandezza analogica. Per esempio, se abbiamo un convertitore con un delta d'ingresso di 10 Volt ed un convertitore A/D ad 8 bit, allora il LSB (Least Significant Bit = bit meno significativo) è rappresentativo di 10/256 = 0,0390625 Volt. Questo significa che il numero "1" (00000001) rappresenta 0.039 Volt mentre il numero 2 (00000010) rappresenta 0.039x2=0.078 Volt eccetera. Dunque una tensione che cade tra le due sarà comunque rappresentata da uno dei due valori disponibili (ossia dal valore 1 o 2); e questa infatti è l'essenza dell'errore di quantizzazione: diversi valori di tensioni rappresentati con il medesimo codice. Si viene pertanto a perdere parte dell'informazione originale, e dunque si introduce una distorsione nel segnale acquisito, che va stimata accuratamente per poter valutare l'incertezza di misura.

L'errore di quantizzazione è dunque compreso tra zero e *q*, ma al fine di avere una riduzione della potenza associata al disturbo di quantizzazione si preferisce fare in modo che l'errore di quantizzazione risulti centrato rispetto al valore nominale in maniera da avere che esso risulti compreso tra ± ½ *q*. In tal modo possiamo dire che, usando ancora il concetto di errore, l'errore che si commette effettuando la quantizzazione di un segnale analogico è pari alla differenza tra il valore associato alla tensione ed il suo valore reale (che comunque non è noto, ma è una comoda astrazione per meglio comprendere i concetti esposti in questa sede). Questo equivale

a dire che un valore di tensione fornito da un convertitore A/D corrisponde a infiniti valori della tensione analogica "reale" e dunque appare evidente come sia possibile adottare una descrizione probabilistica del fenomeno. In tal senso è ragionevole supporre che la variabile aleatoria ξ che associeremo all'errore di quantizzazione sarà di tipo uniforme, ossia la funzione densità di probabilità sarà pari a *1/q* nell'intervallo chiuso *–q/2* e *+q/2* e 0 altrove (vedi fig. 22) [36]

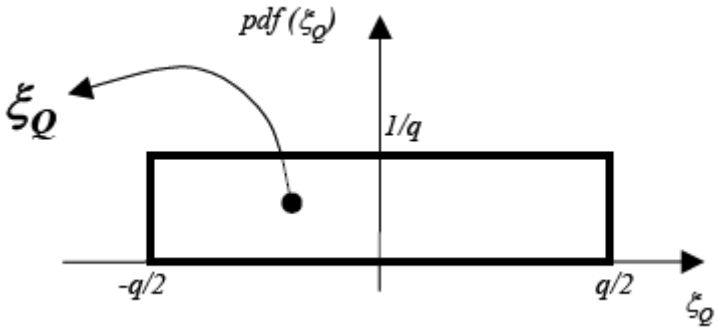


Figura 22: distribuzione uniforme associata all'errore di quantizzazione

In generale, per questa distribuzione, la varianza è pari a:

$$\sigma_Q^2 = \int_{-\infty}^{+\infty} (\xi_q - \mu_q)^2 pdf(\xi_q) d\xi_q = \frac{1}{q} \int_{-\frac{q}{2}}^{+\frac{q}{2}} (\xi_q - \mu_q)^2 d\xi_q = \frac{q^2}{12} \tag{4.2}$$

Dove σ^2 è la varianza e μ_q è il valor medio, che in questo caso supporremo nullo. Associando a questo tipo di errore il tipo di distribuzione e i parametri fondamentali che la descrivono (varianza e valor medio) abbiamo ottenuto il modello matematico dell'errore.

4.5.2 Errore di offset

L'errore di offset è stato descritto in 4.2 ed in particolare nella figura 10. In pratica la traslazione della caratteristica ideale verso l'alto o verso il basso fa si che i codici corrispondenti ad una determinata tensione siano generati in corrispondenza a "punti

di scatto" non più pari a kq (k intero) ma a valori corrispondenti a $kq \pm e_o$ dove con il simbolo e_o indichiamo il contributo dell'errore dato appunto dall'offset. In tal senso possiamo vedere l'errore di offset come un segnale analogico generato internamente al convertitore, che si sovrappone al segnale d'ingresso. Quest'ultima considerazione è invero valida per la maggior parte degli "errori" che consideriamo in questo capitolo.

L'errore di offset, come accennato in altri paragrafi, può essere compensato da meccanismi previsti negli stessi convertitori, essendo apparentemente un errore completamente sistematico; essi comunque dipenderebbero da altri componenti (hardware) a loro volta soggetti a derive ed invecchiamenti e dunque a variazioni non prevedibili. E dunque in tal senso presenta caratteristiche anche di non sistematicità. In generale, ogni errore sistematico che non si riesce a compensare può comunque essere modellizzato come un errore casuale di cui sono note le caratteristiche (intese come parametri della variabile aleatoria che lo caratterizza). Per l'errore di offset è opportuno scegliere una variabile aleatoria continua distribuita nell'intervallo ±o/2 il cui parametro "o" è scelto in base ai parametri forniti dal costruttore o noti per altre vie.

4.5.3 Errore di guadagno

Questo tipo di errore è descritto in 4.2 è rappresentato esaustivamente nella figura 11. Esso è dovuto al convertitore A/D ma precipuamente anche alla circuitazione analogica presente nella scheda di acquisizione, per esempio i circuiti di condizionamento e amplificazione/attenuazione del segnale. Esso dipende dalle tolleranze dei componenti e dalla loro sensibilità alle condizioni di temperatura, umidità e pressione atmosferica, dall'imprecisione dei valori della tensione di riferimento.

Esso inoltre, a differenza dell'errore di offset, interessa i campioni in funzione del valore che essi stessi assumono, ossia in particolare e tanto più marcato quanto più ci si avvicina al fondo scala del convertitore. In generale, si preferisce modellizzare

questa causa d'incertezza come una variabile aleatoria continua distribuita uniformemente nell'intervallo ±g/2 dove la quantità "g" è scelta in base ai parametri forniti dal costruttore.

4.5.4 Errore di linearità

Esso è descritto in 4.2 e rappresentato nella figura 13. In sintesi, esso semplicemente consiste nella "incostanza" del passo q dovuto al fatto che la funzione di trasferimento non è una retta ma si discosta da essa secondo una curva continua e variabile con continuità Dunque, i codici emessi non variano più proporzionalmente al valore di tensione applicato, ossia non variano linearmente con essa. Accenniamo nuovamente alle definizioni di base dei due tipi di non linearità che si possono presentare, che sono poi quelle adottate in questo lavoro. La linearità integrale è data dalla differenza, per ogni parola di codice, tra il valore effettivo della tensione analogica in ingresso e quella della retta di migliore approssimazione. Come è facile evincere da quanto esposto sopra, questo valore varia (sia in termini assoluti che percentuali) nei diversi punti della scala: le specifiche fornite dai costruttori allora forniscono il valore massimo lungo tutta la caratteristica (non specificando in quale punto si verifica).

La linearità differenziale considera due codici adiacenti; essa è relativa alla differenza di "passo" q tra due codici adiacenti; essa è misurata in LSB, e dunque un convertitore A/D ideale dovrebbe avere un errore di non linearità differenziale di 0 LSB, a significare che il passo rimane costante e pari a q.

Per esempio (consideriamo un caso tipico) [36] un convertitore con errore di linearità differenziale di 0.5 LSB implica che si ha una variazione di un codice digitale ogni qual volta il segnale analogico varia di 1±0.5LSB. In altre parole, lo "scatto" può avvenire in corrispondenza di soglie diverse da quelle che ci si aspetta per il convertitore ideale (più piccole o più grandi).

La modellazione di questo tipo di errori è più complessa dei precedenti; si veda per maggiori dettagli [36] da cui si è preso deliberatamente spunto. La modellazione

delle non linearità richiede di formulare alcune ipotesi; alcuni autori suggeriscono di utilizzare una relazione tra l'errore di non linearità e il segnale d'ingresso con una relazione di secondo grado, ossia, dato e_{NL} (errore di non linearità) possiamo scrivere:

$$e_{NL}(x_{in}) = ax_{in}^{2} + c \tag{4.2}$$

Altra ipotesi che facciamo è considerate che il valore medio su tutta la caratteristica sia nullo e che in corrispondenza degli estremi sia pari al valore dichiarato dal costruttore, rappresentato dalla variabile ENL. Di conseguenza possiamo scrivere l'equazione seguente:

$$\begin{cases} \int_{-FSR}^{+FSR} e_{NL}(x_{in})d(x_{in}) = \int_{-FSR}^{+FSR} (ax_{in}^{2} + c)d(x_{in}) = 0 \\ e_{NL}(FSR) = aFSR^{2} + c = ENL \end{cases} \tag{4.3}$$

Dove FSR è l'intervallo massimo del segnale analogico d'ingresso accettato dal convertitore A/D, ENL il valore dell'errore dichiarato dal costruttore (in accordo alle definizioni date in questo paragrafo) ed "a" e "c" delle costanti che possiamo ricavare dal seguente sistema di equazioni:

$$\begin{cases} \int_{-FSR}^{+FSR} (ax_{in}^{2} + c)d(x_{in}) = \left[a\frac{x_{in}^{3}}{3} + cx_{in} \right]_{-FSR}^{+FSR} = 0 \\ e_{NL}(FSR) = aFSR^{2} + c = ENL \end{cases} \Rightarrow \begin{cases} a\frac{FSR^{2}}{3} + c = 0 \\ c = ENL - aFSR^{2} \end{cases} \Rightarrow$$

$$\begin{cases} a\frac{FSR^{2}}{3} + ENL - aFSR^{2} = 0 \\ c = ENL - aFSR^{2} \end{cases} \Rightarrow \begin{cases} a = \frac{3ENL}{2FSR^{2}} \\ c = -\frac{ENL}{2} \end{cases} \tag{4.4}$$

Di conseguenza possiamo scrivere la (4.3) come:

$$e_{NL}(x_{in}) = \frac{3ENL}{2FSR^2} x_{in}^2 - \frac{ENL}{2} \tag{4.5}$$

Se rimuoviamo la condizione che l'errore massimo si verifichi solo nel fondo scala, ammettendo che si possa verificare in un punto qualsiasi della scala, sostituiamo dunque al valore massimo di ENL una variabile aleatoria ξ_{NL} con distribuzione uniforme nell'intervallo ±ENL.

4.5.5 Errore di Noise (o rumore)

Questo tipo di errore è semplicemente preso in considerazione per tenere di conto ogni possibile segnale aggiuntivo che viene ad essere sovrapposto al segnale originario, per esempio per quelle schede di acquisizione interne al PC, che possono risentire dei molteplici disturbi che esso trasmette alla scheda di acquisizione. Oppure, esso può rientrare nel rumore generato dai circuiti propri della scheda di acquisizione (come già accennato, circuiti di amplificazione, condizionamento, etc.). Si veda [37] (nota applicativa della National Instruments), alla quale ci siamo ispirati per le considerazioni che seguono. L'effetto del rumore complessivamente generato dal convertitore A/D in se più il contributo dovuto ai circuiti esterni e dell'ambiente in cui esso è inserito può essere letteralmente devastante. La National fornisce una figura (riportata in fig 23) che ben riassume la situazione, relativa a casi reali.

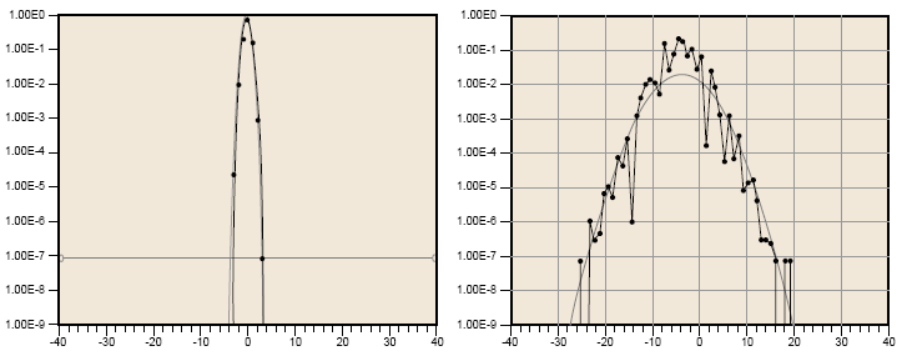


Figura 23: rumore nelle schede di acquisizione

Nella figura 23, a sinistra, il grafico rappresenta la distribuzione (istogramma) dei campioni la cui ampiezza ha un determinato valore. In particolare in ascissa l'ampiezza è data in termini di LSB, in ordinata si quantifica il numero di campioni aventi quel determinato valore (come frequenza sui punti complessivi). Il grafico di sinistra è relativo ad un convertitore A/D a 16 bit senza nessun segnale applicato, e i codici emessi rimangono "confinati" intorno allo zero e qualche punto cade al più nell'intorno di ±3 LSB; il grafico di destra è relativo anch'esso a un convertitore a 16 bit nelle medesime condizioni operative del primo, ma estremamente più rumoroso (o inserito in un ambiente rumoroso) con un rumore che arriva all'astronomica cifra di ±20 LSB. Ad onor del vero la National sostiene che il grafico di sinistra appartiene ad un loro convertitore, mentre il grafico di destra è inerente un convertitore della concorrenza; quello che a noi interessa mettere in evidenza è semplicemente la natura del fenomeno, e quanto può essere determinante ai fini delle prestazioni complessive. Come dunque a questo punto è facile evincere, il nostro rumore è ovviamente caratterizzabile con una variabile aleatoria ξ_N con distribuzione gaussiana e pari ad 1/3 del valore massimo di "noise" dichiarato dal costruttore.

4.4 Effetto complessivo delle cause di incertezza e analisi Monte Carlo

Rifacendoci ancora una volta a [36], possiamo ora descrivere esaustivamente l'effetto complessivo che le varie cause d'incertezza analizzate hanno sul segnale elaborato dalla scheda di acquisizione. La figura 24 sintetizza l'essenza del meccanismo:

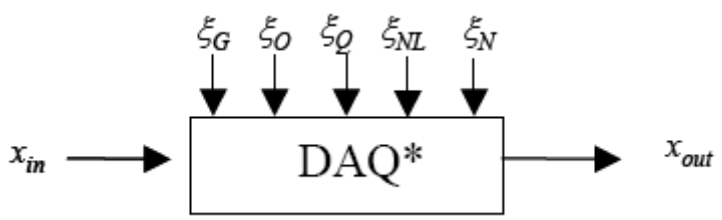


Figura 24: modello complessivo delle sorgenti di errore

Il segnale in ingresso X_{in} viene acquisto dalla scheda DAQ (Digital Aquisition Board) per generare il segnale d'uscita X_{out} il quale sarà alterato dalle cause d'errore descritte nei paragrafi precedenti e indicate come variabili aleatorie con la lettera ξ e dal pedice corrispondente. In particolare la scheda di acquisizione "reale" gli autori di [36] la indicano come DAQ, mentre con DAQ* si indica la simulazione della scheda, data la conoscenza delle variabili aleatorie ξ tramite le quali "modelliamo" le varie sorgenti di errore (note).

Pertanto, nel calcolo dell' incertezza applicando la propagazione delle incertezze come descritto nella GUM, faremo riferimento ai paragrafi da 4.5.1 a 4.5.5 dove definiamo la natura delle variabili aleatorie che scegliamo come meglio rappresentative dell'incertezza in questione. Nel caso invece si voglia effettuare una analisi di tipo Monte Carlo, dovremo "alimentare" i vari strumenti virtuali implementati con variabili costruite secondo il modello indicato in figura 23. In particolare useremo la seguente relazione:

$$x_{OUT} = x_{in} \cdot (1 + \xi_G) + \xi_O + \xi_N + \xi_Q + \xi_{NL} \cdot \left(\frac{3 x_{in}^{\,2}}{2 FSR^2} - \frac{1}{2} \right) \tag{4.6}$$

Nella quale i termini rappresentati dalla variabile ξ rappresentano le varie cause di incertezza modellate da variabili aleatorie con caratteristiche note; FSR è la tensione fondo scala del convertitore A/D usato. In particolare ξ_G rappresenta l'errore di guadagno, ξ_O l'errore di offset, ξ_N l'errore di noise, ξ_Q l'errore di quantizzazione e ξ_{NL} l'errore di non linearità.

L'errore di quantizzazione e di Noise si possono considerare variabili statisticamente indipendenti e dunque è possibile sommare il loro contributo alla variabile d'uscita; viceversa Offset e Guadagno sono totalmente correlate, e dovremo dunque tenere conto di questa correlazione; ci limiteremo comunque a sommare l'errore di offset e comprendere un termine, per l'errore di guadagno, che sia proporzionale al segnale d'ingresso. Queste ultime considerazioni sono delle semplificazioni che consideriamo accettabili in questo contesto. Gli errori di non linearità sono infine tenuti di conto con l'ultimo termine a secondo membro della 4.6 (tra parentesi).

Nel programma è stata implementata una routine che si “sostituisce” alla scheda di acquisizione, generando una variabile fittizia in accordo alla (4.6) e che dunque “alimenta” gli strumenti di misura per i quali è possibile effettuare una analisi statistica dei risultati (finestra incertezza di tipo A) e dunque ottenere un’analisi di tipo Monte Carlo. Nella figura 24-bis è possibile vedere la finestra che consente l’impostazione del device “fittizio”, e nelle figura 24-ter la finestra di impostazione dei parametri di incertezza descritti nel capitolo più altri relativi al tipo e caratteristiche del segnale usato.

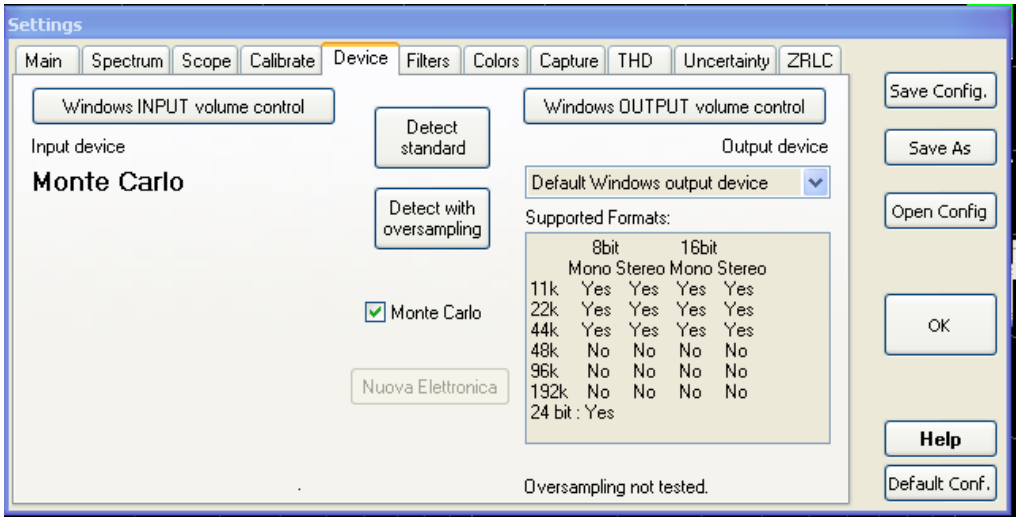


Figura 24-bis: impostazione del device di ingresso “monte carlo”

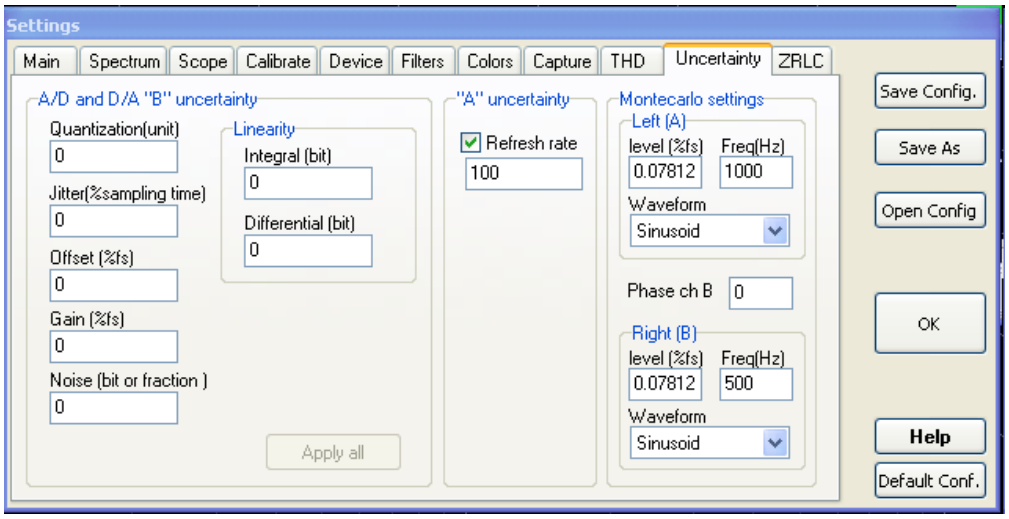


Figura 24-ter: parametri dell’incertezza e tipo di segnale usato

Capitolo 5

Architettura del programma e strumenti implementati

5.1 Introduzione

L'architettura software e hardware del programma è stata oggetto di un apposito capitolo in [1] relativamente alla versione del programma disponibile all'epoca, decisamente meno evoluta e più povera di funzioni e utilità. Si ritiene pertanto importante descrivere nuovamente a grandi linee l'architettura generale di VA ponendo enfasi sui punti per i quali c'è stata particolare evoluzione o quelli completamente nuovi.

Visual Analyser è un programma che appartiene ad una categoria di software che si vanno sempre più diffondendo nel mondo delle misure, ossia quelli dei "virtualizzatori" di strumenti di misura elettroniche/elettriche "PC-based"; essi partono dall'assunto che un personal computer è un hardware normalmente disponibile, e dunque a costo zero; e che quindi può essere usato per realizzare strumenti di misura (virtuali) che svolgono le stesse funzioni di uno strumento dedicato ma a costo notevolmente inferiore; e con tutta una serie di vantaggi ulteriori descritti nel capitolo I e nel capitolo corrente.

L'architettura software di questa categoria di programmi presenta degli elementi comuni, sebbene Visual Analyser presenti delle peculiarità esclusive, che appariranno chiare nel corso della trattazione.

Software come VA sono divenuti possibili verso l'inizio degli anni '90, grazie all'aumento considerevole della capacità elaborativa dei normali Personal Computer, che li ha resi in grado di competere con microprocessori dedicati come i DSP (cfr cap. III). Questi ultimi erano e sono generalmente utilizzati in tutte quelle situazioni in cui è necessario operare in tempo reale su dati relativi a segnali acquisiti a loro volta in tempo reale; e dunque nel tempo intercorrente tra l'acquisizione di un campione e l'altro o tra blocchi di essi. Sebbene è opinione diffusa che i DSP siano stati in gran parte sostituiti dai PC, in effetti esistono molti campi di applicazione in cui i DSP sono insostituibili; per esempio in tutta una fascia di applicazioni che fanno riferimento ai cosiddetti "sistemi embedded in tempo reale" sino ai più semplici elettrodomestici di uso comune e di dimensioni ridotte (es. telefoni cellulari, autoradio, elettromedicali portatili per diagnostica) e di tipo professionale come elettromedicali e strumenti di misura in genere, virtuali e non. Lo stato dell'arte vede i PC *affiancarsi* alle apparecchiature a DSP, od addirittura ad essere utilizzati contemporaneamente sulla stessa piattaforma hardware. Anche nei moderni personal computer per uso ufficio o privato, non è infrequente l'uso di schede grafiche, audio e/o video con DSP "on board" per effettuare elaborazioni in tempo reale su grosse quantità di dati, allo scopo di alleggerire la CPU da oneri computazionali eccessivi.

I personal computer possono così essere utilizzati per applicare la maggior parte degli algoritmi studiati dalla materia nota come "Elaborazione Numerica dei Segnali" il cui adattamento e ottimizzazione all'uso su elaboratore è a sua volta studiato dalla materia nota come "Calcolo Numerico" o "Analisi Numerica".

Programmi come Visual Analyser sono facilmente reperibili in rete, ma i programmi "stato dell'arte" si riducono a poche unità; in particolare, in quasi nessuno di essi è contemplato il benché minimo calcolo dell'incertezza, e laddove dichiarata essa non è supportata da nessuna indicazione che consenta di capire le tecniche utilizzate per la sua stima.

Si veda l'appendice III per una esaustiva presentazione e confronto (critico) dei prodotti presenti sul mercato al momento della stesura di questo lavoro.

Nota:

Nel corso del presente capitolo è possibile che il termine “task” venga usato in luogo di “thread” e viceversa, intendendo dunque la completa equivalenza dei termini; a rigore questo non è affatto corretto, ma in questa sede la cosa è ininfluente. Si tenga presente che la gestione dei thread non è supportata da tutti i sistemi operativi che al contrario possono comunque avere una gestione del multitasking; Windows e Linux offrono la gestione di entrambe le categorie. Ai fini del presente capitolo è inutile operare una distinzione tra i due concetti, comunque vicini tra loro, e pertanto con i termini “thread” e “task” si abbia ad intendere il generico concetto di processi in esecuzione concorrente e/o parallela.

5.2 Architettura generale e acquisizione campioni

Visual Analyser preleva i campioni dalla scheda di acquisizione con un meccanismo che fa un uso diretto delle API di Windows, per evitare la dipendenza da qualsiasi strato software aggiuntivo e allo scopo massimizzare la velocità di esecuzione; la natura strettamente real-time del programma ha reso necessario l’uso intensivo delle API di sistema in molte sezioni del programma, e talvolta il ricorso a porzioni di codice scritte direttamente in linguaggio assemblativo. In tal senso, le parti non strettamente real-time, quali le finestre di ispezione dei dati acquisiti e l’interfaccia utente nel complesso, sono invece state realizzate sfruttando oggetti di libreria (dei quali è comunque disponibile il codice sorgente) e “facilities” visuali del compilatore usato (C++Builder 2010) allo scopo di massimizzare la produttività e favorire la concentrazione delle energie profuse in questo lavoro esclusivamente su aspetti algoritmici e concettuali.

Ogni scheda di acquisizione dati, tra le quali includere anche una semplice scheda sonora, deve essere dotata di uno specifico driver e di meccanismi hardware/software interni che consentano di acquisire e ricevere campioni tramite chiamate dirette alle API di sistema, secondo quanto specificato nel capitolo III. Si osservi che, contrariamente a quanto accadeva sino a pochi anni fa, non è più possibile, o quasi,

un accesso diretto alle periferiche tramite chiamate in linguaggio assemblativo, e dunque l'uso delle API è quanto di più "vicino alla macchina" si possa utilizzare.

Il meccanismo generale prevede che i dati acquisiti vengano raggruppati in blocchi di dimensioni predefinite, definibili dall'utente, e ad una determinata frequenza di campionamento, anch'essa definibile dall'utente nei limiti della capacità della scheda di acquisizione stessa e capacità elaborative del PC.

I campioni così acquisiti vengono scaricati nella RAM del personal computer tramite un "canale" di comunicazione e preventivamente impacchettati a blocchi dalla scheda stessa. A parte la funzione di acquisizione e impacchettamento, la quasi totalità delle funzioni è svolta da Visual Analyser, nel senso che VA non prevede una fase di pre-elaborazione dei campioni da parte di hardware residente sulla scheda di acquisizione; queste sono funzioni necessarie per una tipologia di strumenti virtuali che operano a frequenze molto più elevate e devono essere coadiuvati da microprocessori esterni che effettuano una parte consistente delle elaborazioni (ad esempio filtraggi digitali, finestrature, FFT, etc.), relegando al programma su PC un semplice ruolo di gestore evoluto dell'interfaccia utente o poco più.

In tal caso, il software lato PC è dunque chiamato, rifacendoci ad una terminologia ben introdotta nel capitolo I, alla sostituzione di un numero minore di segmenti hardware.

In VA l'architettura generale prevede il funzionamento "simultaneo" (concorrente) di più task interni (implementati in multithreading) che adempiono a tutte le principali funzioni del programma, compresa quella di elaborazione, visualizzazione ed acquisizione del segnale, oltre alla gestione dell'interfaccia utente.

La struttura architetturale è quella indicata nella fig. 25.

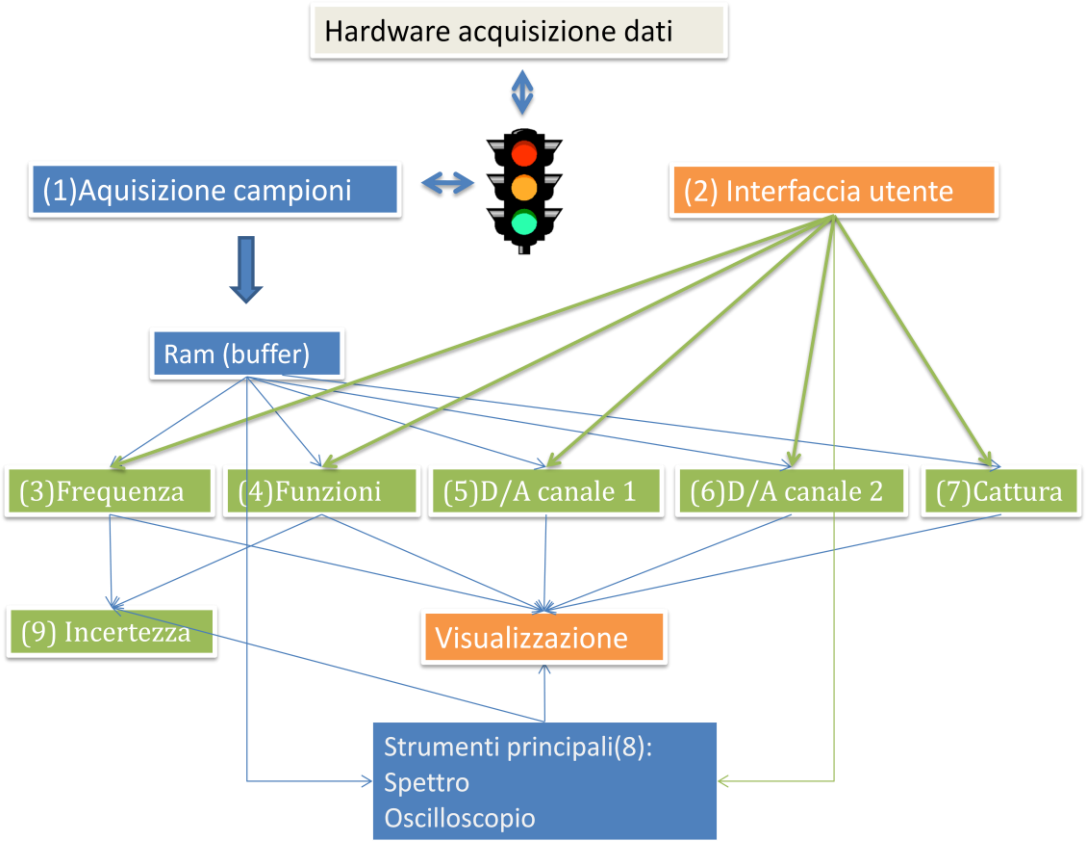


Figura 25: architettura del programma

Essa comprende dunque sette task principali, numerati da 1 a 7 più il 9, cui invero possono affiancarsene altri a seconda degli strumenti attivati (v. descrizioni task); il task (4) è da intendersi come generico, e verrà di volta in volta "personalizzato" in ragione della funzione specifica invocata. In prima battuta, il funzionamento generale è basato su di un thread principale di acquisizione dati che "alimenta" i numerosi thread che implementano le varie funzioni, depositando i campioni prelevati nella RAM del PC. Esso a sua volta riceve i dati dall'hardware di acquisizione. Il thread (2) dell'interfaccia utente gestisce l'interazione con l'utente (che può richiamare i vari strumenti e gestire le varie opzioni, mettere in "on" e "off", stabilire legami tra i vari strumenti, effettuare salvataggi, etc.) e la visualizzazione a video in tempo reale delle misure effettuate (esempio oscilloscopio, voltmetro).

Il task dedicato all'acquisizione dei campioni (indicato come "Acquisizione campioni" in figura, e individuato dal numero 1), dialoga direttamente con la scheda di acquisizione, e gira alla massima priorità consentita (tempo reale); esso si

sincronizza con la scheda a mezzo di un semaforo, messo a “verde” o “rosso” dalla scheda stessa a mezzo di interrupt hardware. Esso dunque si sospende sul semaforo eventualmente messo a “rosso”, sino a che l’hardware non ha finito di riempire il buffer di “n” campioni, e dunque per un tempo approssimativamente calcolabile in funzione della frequenza di campionamento scelta e dalle dimensioni del buffer. Per esempio, scegliendo una frequenza di campionamento di 40960 Hz ed un buffer di 4096 campioni, avremo che il tempo minimo necessario per riempire un buffer è pari a 4096/40960 = 100 millisecondi. Dunque, il thread “Acquisizione campioni” sarà sospeso per 100 millisecondi sino a che il semaforo non viene nuovamente settato a “verde”; appena sbloccato esso provvederà a copiare nel minor tempo possibile i dati nella RAM del computer, effettuando qualche semplice operazione di base (per esempio potrebbe essere necessario “shiftare” i dati di uno o più campioni per compensare la presenza di un solo convertitore analogico-digitale “multiplexato”, utilizzato per tutti i canali di acquisizione) e successivamente si sospenderà ancora in attesa di un nuovo buffer, e così via in un ciclo infinito. Un simile meccanismo, seppur essenzialmente sufficiente ad un corretto funzionamento nella maggior parte dei casi, è invero non ancora completo; è necessario implementare un meccanismo a coda per prevenire eventuali perdite di dati. Infatti, un sistema complesso come un personal computer, dove più programmi possono essere in esecuzione allo stesso tempo, non garantisce che al momento della messa a verde del semaforo sia possibile effettivamente eseguire il task sospeso; in altre parole, il computer potrebbe essere occupato a fare qualcosa d’altro a maggiore priorità (es. attività di sistema), e dunque andrebbe risvegliare effettivamente il Thread dopo un tempo aggiuntivo (latenza); con possibile perdita di dati. Infatti, se il buffer non viene scaricato in RAM prima che arrivi il successivo, esso viene irrimediabilmente perso.

Si è dunque implementata una struttura a coda che consente di memorizzare in RAM una serie di buffer di campioni, il cui numero viene determinato dinamicamente in funzione dei parametri scelti (frequenza di campionamento e dimensione del buffer). In tal modo si riesce a compensare temporanee fluttuazioni nella disponibilità di risorse elaborative, contando sul fatto che generalmente tutte le operazioni di misura richiesta dal programma vengono eseguite in un tempo notevolmente minore del

tempo intercorrente tra due buffer successivi. Quest'ultima condizione è indispensabile al corretto funzionamento della coda software; se infatti i calcoli venissero effettuati in un tempo quasi uguale a quello inter-buffer, non ci sarebbe possibilità, in caso di accumulo dei buffer, di "smaltire" i buffer temporaneamente accumulati e dunque la coda sarebbe completamente inutile. Taluni programmi (anche commerciali) non usano il meccanismo a coda, e fanno affidamento sul fatto che gli eventi che portano a dover accumulare i buffer sono di per se rari; e dunque considerano accettabile una temporanea perdita di dati. In VA sono stati implementati svariati meccanismi di cattura dei dati anche molto sofisticati, pilotati da eventi e soglie, e che pertanto rendono VA utilizzabile come uno strumento adatto a catturare segnali impulsivi e sporadici; in tal senso diverrebbe inaccettabile una perdita sia pur minima di dati utili.

Il meccanismo di gestione della coda, già ampiamente descritto in [1], è realizzato tramite chiamate ad API di sistema e allocazioni dirette di aree di memoria; lo schema indicato in fig 25 è da intendersi come "logico".

5.3 Strumenti principali

Nel blocco individuato come "strumenti principali" trovano realizzazione i due strumenti "principali" di VA, l'oscilloscopio ed analizzatore di spettro. Oltre ad essere i due strumenti di misura sempre abilitati, anche quando in funzionamento a "finestre flottanti", sono alla base di altri strumenti che utilizzano le misure da essi fornite (esempio ZRLC, vedi cap. VIII, IX). In particolare poi lo strumento "analizzatore di spettro" verrà descritto in dettaglio nel cap. VII, e nel capitolo VI relativo al rounding error. Diamo qui pertanto un accenno alle caratteristiche principali dell'analizzatore di spettro e qualche cenno all'architettura dell'oscilloscopio, una cui importantissima funzione è descritta nel paragrafo 5.8 di questo stesso capitolo (conversione D/A). Una caratteristica comune a questi due strumenti è di non essere strettamente realizzati a thread. Sebbene essi siano compresi nel thread principale di VA, e dunque ancora realizzati a thread, non esiste

per essi l'allocazione specifica di un task. Questo per svariati motivi, primo fra tutti l'esplicita volontà di limitare il proliferare di thread e dunque di frequenti "switch di contesto" di per se fonte di notevole "consumo" di risorse computazionali. Il thread principale di VA è quello relativo all'acquisizione dei campioni ("1" in figura 25); esso gira in un loop infinito ed a priorità elevata (quella che in Windows è definita come "real-time") e contiene le chiamate alle procedure relative al calcolo di FFT e oscilloscopio, disegno a video dei risultati su grafici, oltre a calcoli di supporto (per esempio trigger nel caso dell'oscilloscopio o valori caratteristici delle forme d'onda). In particolare il calcolo della FFT è eseguito sul buffer di campioni corrente, le cui dimensioni sono dunque vincolate ad essere una potenza di due; il calcolo è effettuato mediante una trasformata FFT a decimazione di tempo (si rimanda a [1] per una descrizione dettagliata). Gli stessi dati utilizzati per calcolare la trasformata di Fourier vengono utilizzati per implementare lo strumento oscilloscopio.

5.3.1 Analizzatore di spettro

L'ampiezza delle armoniche in funzione della frequenza è visualizzata in una delle finestre grafiche della finestra principale di VA (v. fig. 26)

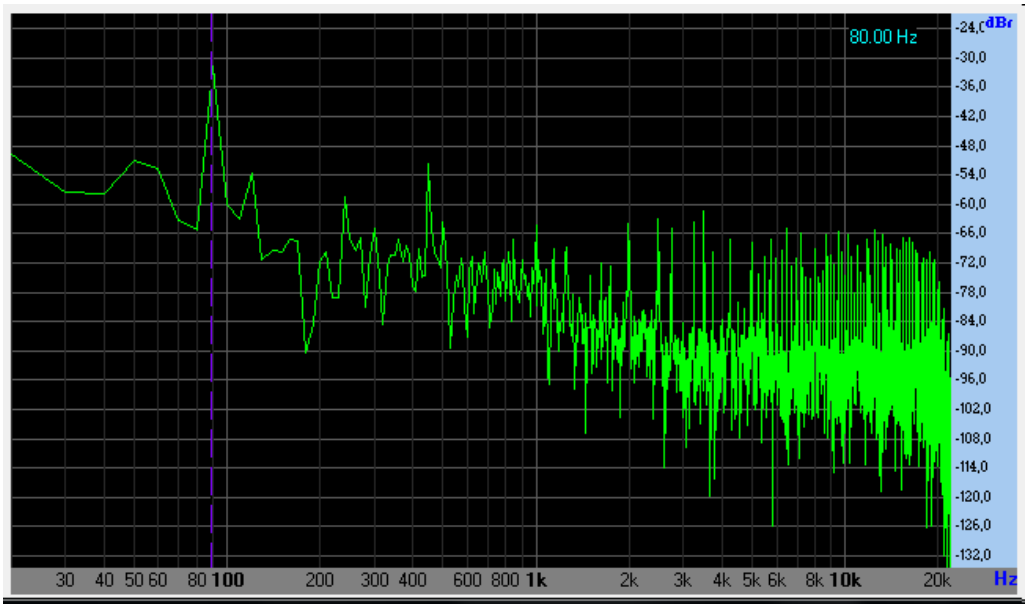


Figura 26: analizzatore di spettro

Viceversa la fase è rappresentata in una finestra separata, normalmente non abilitata. Circa quest'ultima è stato implementato un meccanismo a soglia, tramite il quale la fase, armonica per armonica, viene effettivamente visualizzata se l'ampiezza relativa a quella determinata armonica supera una soglia prefissata (in dB od in Volt).

Questa opzione è particolarmente utile se è necessario osservare la fase di un segnale: se esso è composto da poche armoniche principali, la fase che interessa visualizzare a schermo è appunto quella relativa alle sole armoniche effettivamente costituenti il segnale; nella finestra della fase verrebbero invece visualizzate tutte le armoniche calcolate, la cui maggioranza è in effetti solo costituita da rumore. Ma siccome nel diagramma delle fasi abbiamo la frequenza in ascissa e la fase in ordinata, avremmo un grafico composto da valori di fase completamente casuali sulla maggior parte del grafico (variabili da -180 a +180 gradi) che renderebbero estremamente difficile la lettura dei valori d'interesse, anche loro con valore della fase compresa nel medesimo intervallo.

In altre parole, in questo tipo di grafico l'ampiezza (in gradi) della fase delle armoniche "utili" del segnale e comparabile con quello delle armoniche di ampiezza trascurabile (es. rumore), e dunque di difficile distinzione.

Per chiarire le idee, nella figura 27 è riportato il grafico della fase di un segnale sinusoidale puro a 1000Hz, senza nessuna soglia applicata; come è facile vedere esso risulta praticamente inutilizzabile; impostando invece una soglia paragonabile con il livello dell'armonica principale del segnale analizzato otteniamo un grafico della fase sicuramente efficace (v. fig. 28).

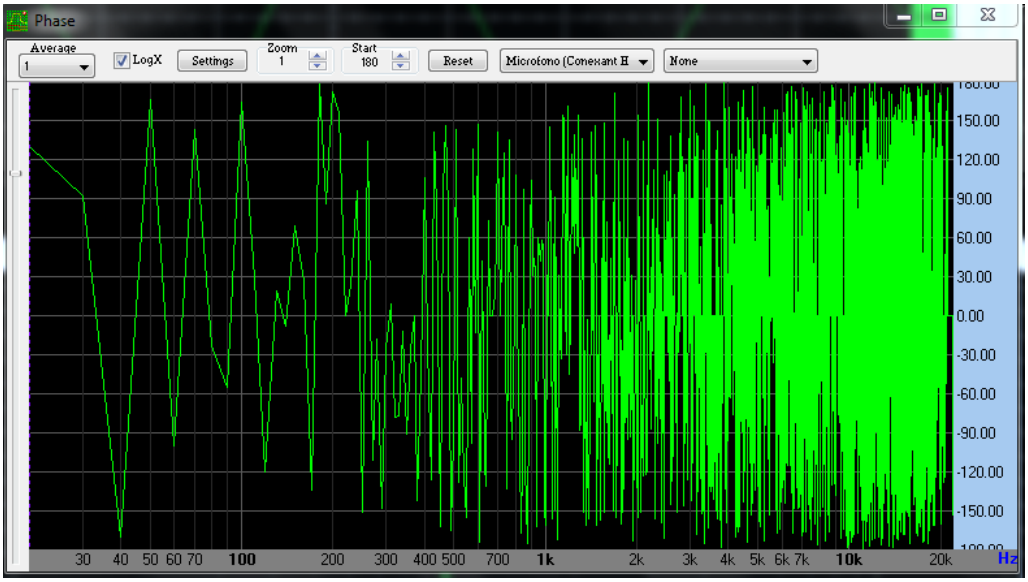


Figura 27: fase di un segnale sinusoidale a 1000 Hz senza soglia

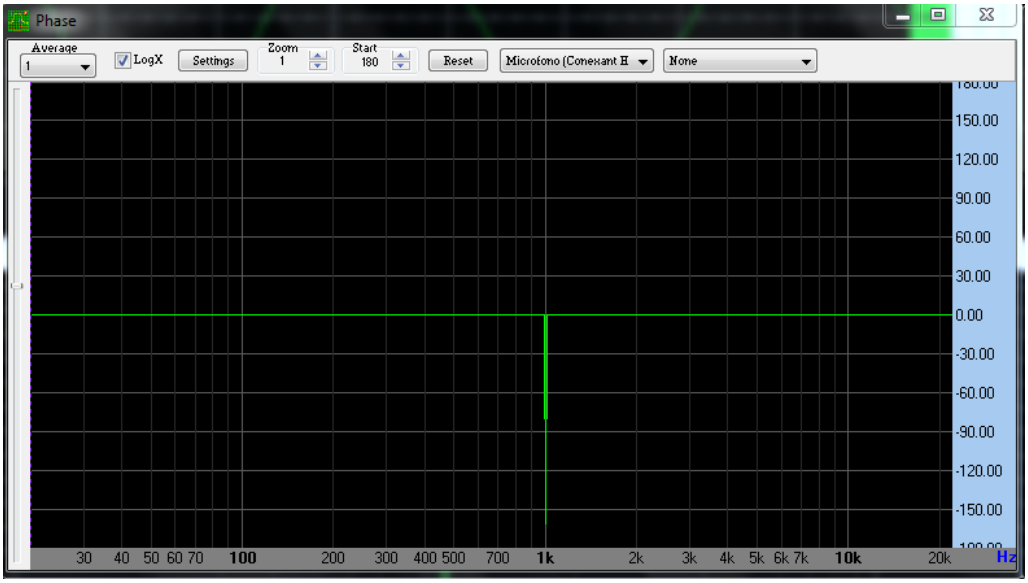


Figura 28: fase del segnale sinusoidale di fig. 7 con soglia impostata a -70 dB

5.3.2 Oscilloscopio

L'oscilloscopio è, come l'analizzatore di spettro, uno degli strumenti direttamente presente nella finestra principale di VA. Esso consente di visualizzare il segnale acquisito nel dominio del tempo, ossia, detto in altri termini, riportando in asse X il valore del tempo ed in asse Y l'ampiezza del segnale. Quest'ultima può essere in percentuale fondo scala, per massima generalità, oppure in Volt se è stata effettuata

la taratura. Quest'ultima si "propaga" sia nell'analizzatore di spettro, se selezionata la rappresentazione lineare in asse Y, sia nel voltmetro. La taratura (calibration) (par. 2.3) è una funzionalità aggiuntiva che consente di effettuare l'associazione tra un valore noto di tensione (noto come valore picco-picco, picco o RMS) ed il valore numerico letto. I parametri calcolati vengono (opzionalmente) memorizzati in un file con estensione ".cal" e richiamati manualmente o automaticamente alla partenza del programma. Naturalmente la taratura in Volt è dipendente dalla scheda di acquisizione usata e da molti altri fattori che a parità di dispositivo possono far variare nel tempo alcune caratteristiche, per cui va eventualmente ripetuta il più spesso possibile ed almeno ogni volta che ci si accinge ad effettuare una nuova sessione di misure. Si pensi infatti alle variazioni delle caratteristiche della scheda dovute al variare della temperatura, umidità e invecchiamento dei componenti; in pratica, è possibile associare ad ogni scheda un file di taratura (estensione .cal), da richiamare previa selezione, ma esso potrebbe comunque divenire obsoleto nel giro di poco tempo. Come regola pratica si può considerare valido il file memorizzato anche a distanza di giorni o mesi se ci si accontenta di una minore accuratezza, ed effettuare una nuova taratura in solo in caso si necessiti della massima accuratezza possibile; in tal senso è dunque consigliabile effettuare la taratura immediatamente prima della sessione di misure.

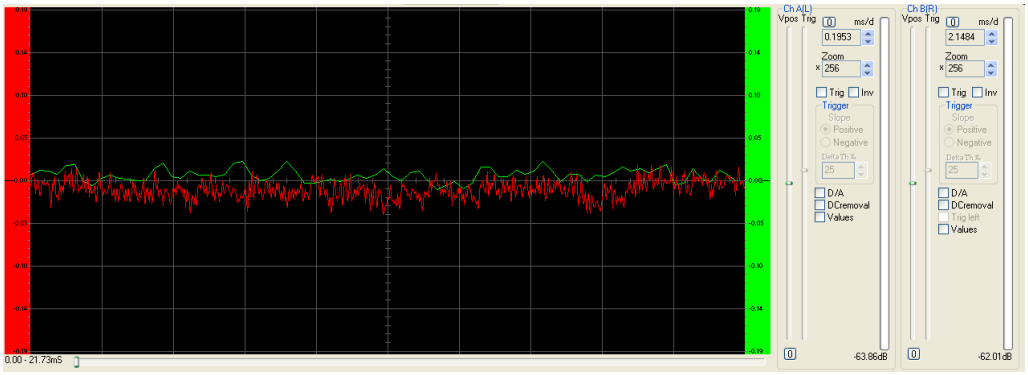


Figura 29: la finestra oscilloscopio con le opzioni

A titolo di esempio si consideri che uno dei più importanti strumenti implementati in VA è l'impedenzimetro (cap. VIII e IX), per il quale sono stati sviluppati complessi

algoritmi di calibrazione per l'eliminazione degli errori sistematici più significativi; calibrazione viene automaticamente ripetuta prima di ogni singola misura.

La rappresentazione in percentuale fondo scala è relativa al "fondo scala digitale" ossia dunque dipendente dalla profondità di bit selezionata. Essa dunque è espressa in funzione del massimo valore numerico rappresentabile dalla variabile utilizzata per rappresentare il singolo campione. Per esempio, utilizzando 8 bit, si ha che il segnale raggiunge il 100% di ampiezza quando la sequenza numerica che lo rappresenta raggiunge il valore di 256 ($=2^8$).

Da osservare che i sistemi Windows presentano notevoli complicazioni per quanto riguarda l'esatta impostazione dei livelli di sensibilità d'ingresso, perché essi dipendono da parametri hardware della scheda di acquisizione che vengono generalmente controllati via software (il famoso mixer di Windows) ed altri indipendenti controllati in hardware (o firmware) direttamente dall'hardware della scheda. In altre parole, una eventuale taratura potrebbe venire inficiata da variazioni imprevedibili dei controlli di guadagno "disseminati" nel percorso del segnale (ad esempio perché un altro software ne varia il valore). In tal senso il meccanismo di taratura implementato provvede a disabilitare ogni possibile controllo di sensibilità/livelli da esso stesso gestiti, ma lo stesso non è possibile fare per azioni "esterne" che il programma non può controllare. E' lasciato dunque al controllo dell'utilizzatore finale il compito di non effettuare variazioni di alcun tipo che non siano gestite automaticamente dal programma.

Le principali caratteristiche dello strumento oscilloscopio sono:

- Visualizzazione in tempo reale della forma d'onda, per ogni singolo buffer; in altre parole, ogni buffer di campioni prelevato dalla scheda di acquisizione, viene immediatamente passato alla funzione oscilloscopio che provvede a disegnarla a video per tramite di procedure realizzate direttamente ad API per motivi di velocità;
- Time-division; avendo a che fare con un numero discreto di punti, sia lato campioni che lato scheda video (il disegno è costituito da un'insieme di punti) esso viene in pratica implementato gestendo la corrispondenza

"numero di campioni/ pixel utilizzati per rappresentarli". Il metodo di default fa corrispondere direttamente un campione e un pixel. Variando il time-division possono essenzialmente verificarsi due casi rispetto alla situazione base: si visualizzano "n" punti per pixel, dove "n" è necessariamente un numero intero, e dunque viene visualizzato un segmento di retta verticale che unisce tutti i punti; si visualizza un campione ogni "n" pixel, dove anche in questo caso "n" è un numero intero, ma rappresenta di quanti pixel differiscono due campioni consecutivi; in questo caso i due punti spaziati di "n" pixel vengono raccordati da un segmento di retta.

- Gestione della "posizione orizzontale"; se un buffer di punti eccede come numero quello dei pixel effettivamente usati per tracciare il grafico (e fissate le impostazioni di time-division) il grafico dello strumento oscilloscopio è da intendersi come una "finestra" che consente di accedere ad una porzione dei dati effettivi (che eccedono le dimensioni della "vista" reale che offre la finestra), finestra che si può spostare a piacimento tramite una trackbar appositamente abilitata a questa funzione e che si trova a ridosso dell'asse X. A fianco della trackbar è riportata l'indicazione (in millisecondi) della porzione temporale dei dati effettivamente visualizzata.
- Trigger: questa funzione consente di effettuare il trigger del segnale relativamente alla soglia impostata sul fronte di salita o di discesa del segnale; l'algoritmo utilizzato è relativamente semplice; tramite apposite trackbar presenti nei pannelli dei comandi dei due canali è possibile impostare il livello di soglia desiderato, visualizzato da una barra tratteggiata orizzontale; l'algoritmo di trigger provvede ad "aspettare" che il segnale raggiunga quel determinato livello e solo allora provvede al suo effettivo disegno, a partire da quel punto. Per rendere al funzione più efficace è stata impostata una "delta-threshold" (espressa in millesimi) che consente di effettuare un controllo sulla soglia a meno di un delta. In altri termini, invece di aspettare che la soglia definita venga superata di un solo campione (delta threshold impostata a zero) si aspetta che la soglia venga superata di un "tot"

di campioni. In tal modo si riesce a “triggerare” anche su forme d’onda rumorose

- Conversione D/A: questa funzione verrà descritta in dettaglio nel paragrafo 5.8; a grandi linee essa consente di ricostruire la forma d’onda applicando il teorema di Nyquist in tempo reale e di conseguenza, per motivi di velocità, con opportune semplificazioni e meccanismi di pre-calcolo. Questa funzione si rende necessaria quando il numero di punti discreti disponibili e molto ridotto rispetto alla frequenza del segnale da analizzare. Infatti, sebbene matematicamente i campioni disponibili siano sufficienti a rappresentare esaustivamente la forma d’onda, la rappresentazione a video con semplice raccordo dei punti adiacenti con segmenti di retta ne costituisce un’approssimazione inaccettabile. Tramite questa funzione è possibile ricostruire il segnale originale e ottenere una rappresentazione virtualmente continua del segnale stesso.
- Parametri in tempo reale: oltre alle funzioni accessibili da pannello, le cui ultime semplici funzionalità verranno descritte nel successivo punto, è stato implementato un meccanismo grafico che consente di determinare i principali parametri della forma d’onda tramite l’uso del mouse durante il funzionamento in tempo reale. In particolare è possibile determinare l’ampiezza picco-picco del segnale e la sua frequenza. Quest’ultima è determinabile definendo tramite mouse i punti di inizio e fine del calcolo, ed allo stesso modo l’ampiezza impostando i due estremi tra i quali effettuare il calcolo. I risultati del calcolo vengono presentati in una finestra flottante adiacente al cursore del mouse.
- Restanti funzioni: comprende la possibilità di “zoom” del grafico, posizione verticale e funzione “inv”. Quest’ultima consente di invertire il disegno della forma d’onda, come è normalmente possibile fare in molti oscilloscopi commerciali, semplicemente invertendo il segno dei campioni. In tal modo è possibile compensare eventuali inversioni di fase di amplificatori inseriti nel percorso del segnale (per esempio presenti nella stessa scheda di acquisizione).

5.4 I thread

Nei paragrafi precedenti abbiamo descritto il funzionamento del thread principale (acquisizione dati) e funzioni da esso implementate, in questo paragrafo ed in tutti i paragrafi successivi verranno effettuate alcune ulteriori considerazioni su questo thread e sul funzionamento globale del programma; infine, verranno descritti in dettaglio i rimanenti thread in appositi sottoparagrafi.

Il thread di acquisizione dei campioni lavora incessantemente e rende disponibili i campioni in RAM tramite il classico meccanismo "produttore – consumatore", sotto forma di sequenze di valori. Dal punto di vista dei "consumatori" i campioni sono contenuti in vettori di dimensione prefissata, cui è possibile accedere in condizioni di massima sicurezza e in assenza di conflitto. Per ottenere questo risultato, la maggior parte degli strumenti risulta implementata in questo stesso thread, che implementa le operazioni relative agli strumenti simulati in maniera strettamente sequenziale, ancorché estremamente rapida e trasparente all'utente, in modo da non dover risolvere problematiche di accesso contemporaneo ai dati. In caso di realizzazione a thread separati, vi sarebbe un proliferare di differenti thread con conseguente calo di prestazioni; vuoi per la necessità di frequenti cambi di contesto, e vuoi perché si sarebbe costretti ad un uso intensivo di primitive di sincronizzazione come semafori, eventi o mutex.

Il thread interfaccia utente (numero 2 in figura 25) gestisce l'interfaccia utente, intendendo per essa la gestione dell'interazione tra utente e programma, e dunque la gestione degli strumenti classici usati da Windows, o "controlli": bottoni, menù, checkbox, listbox, edit eccetera, oltre naturalmente al mouse in se e molti altri dettagli cui sostanzialmente l'utente medio di Windows è generalmente abituato.

I rimanenti thread invece gestiscono funzioni particolari che non possono essere gestite nel ciclo principale, sebbene anche loro debbano essere gestite in tempo reale, anche se con un grado di "constraint" temporale più ridotto. In tal senso, vengono fatti girare con priorità meno elevate dei due thread principali e progettati per dare risposte in tempi dell'ordine delle decine di buffer, invece che di un solo buffer come

per gli strumenti considerati strettamente in tempo reale. Per chiarire le idee, si pensi ai tempi relativi al solito esempio di un tempo intercorrente tra due buffer di 100 millisecondi. Tutti gli strumenti la cui esecuzione completa avviene tra due buffer consecutivi, e dunque entro 100 millisecondi sono considerati "strettamente in tempo reale" (per esempio l'oscilloscopio, per il quale è fondamentale visualizzare tutti i campioni a video mentre vengono acquisiti, con latenza minima). Un'altra categoria di strumenti, abbisognano per loro natura di un numero maggiore di campioni per poter effettuare un calcolo sufficientemente accurato delle grandezze misurate, e quindi di tempi di calcolo proporzionalmente più elevato; ma essi misurano delle grandezze che per loro stessa natura non variano molto rapidamente e dunque la misura è considerata accettabile se fornita in tempi dell'ordine dei secondi (per esempio la misura della frequenza: uno dei meccanismi implementati in VA utilizza un numero di punti dell'ordine delle dieci volte i punti del buffer "base" acquisito, e questo per poter raggiungere una risoluzione accettabile; allora esso è implementato in un thread separato a bassa priorità che esegue i calcoli nei tempi morti dei calcoli a maggiore priorità, spuntando tempi di esecuzione più elevati ma adeguati alla grandezza da misurare).

I thread complessivamente implementati allo stato attuale di evoluzione del software sono:

- Acquisizione campioni (1);
- Interfaccia utente (2);
- Frequenzimetro (3);
- Generatore di funzioni (due Thread) (4);
- Conversione D/A (due Thread) (5,6);
- Cattura e data log (7).
- Incertezza statistica ("n" Thread)(8)

Il primo è stato ampiamente descritto; il secondo è stato descritto ed è comunque "letteratura" di Windows. Di seguito le descrizioni dei rimanenti.

5.5 Frequenzimetro

Il thread è relativo allo strumento frequenzimetro. Esso risulta implementato anche con altre tecnologie in differenti finestre del programma; in primis effettueremo la descrizione relativa al frequenzimetro implementato come finestra separata e thread dedicato, e con risoluzione definibile dall'utente (e dunque con incertezza conseguente). Il secondo metodo implementato, utilizzabile in tempo reale durante il normale funzionamento dell'oscilloscopio (per default sempre attivo), si basa su un semplice algoritmo di zero-crossing, il quale mostra i suoi limiti nel calcolo della frequenza di forme d'onda particolarmente rumorose. In ogni caso, anche in questa seconda modalità, è stato effettuato il calcolo dell'incertezza.

Il calcolo della frequenza dell'armonica a maggiore ampiezza viene normalmente effettuato tramite la FFT (Fast Fourier Transform) che consente di effettuare, una volta calcolata, una stima pressocchè immediata della frequenza "dominante", giacché risulta sufficiente individuare il coefficiente a maggior ampiezza e la frequenza ad esso associata. Per contro, la trasformata di Fourier calcolata tramite l'algoritmo FFT necessita di un numero di punti che sia una potenza di due; ed in ogni caso le armoniche calcolate dall'algoritmo sono dipendenti dall'ampiezza del buffer e dalla frequenza di campionamento. Data la relazione:

$$R = Fc / n \quad (Hz) \tag{5.1}$$

Con essa possiamo calcolare la risoluzione *R* della FFT; "*Fc*" è la frequenza di campionamento e "*n*" è il numero di punti del buffer. Tutte le armoniche calcolate dalla FFT saranno un multiplo di questa risoluzione, a partire da zero; per esempio, usando una frequenza di campionamento di 44100 Hz ed un buffer di punti di 4096 punti ($=2^{12}$) otteniamo una risoluzione R = 44100/4096=10,77 Hz. Ossia, le armoniche calcolate dalla FFT saranno alle frequenze "F":

$$F = R * i \quad dove\ i = 0..(4096/2)\ ed\ F\ in\ Hz \tag{5.2}$$

Ossia, se R=10,77 si ottiene:

0, 10.77, 21.54, 32.31, ……., 22050 Hz

dunque esse sono calcolate esattamente solo per determinati valori, ossia "discretizzate"; frequenze che ricadono in punti intermedi sono teoricamente non calcolate, ma nella realtà, in un sistema che usa un campionamento non sincrono, l'effetto del troncamento, sia pur mitigato da una eventuale finestratura, produrrà comunque armoniche che ricadono in quelle calcolate, e che dunque forniscono una sorta di indicazione (inaccurata) della misura voluta. In altre parole, una frequenza che cade tra due consentite verrà individuata come una delle due limitrofe, introducendo un'incertezza comunque calcolabile e direttamente proporzionale alla risoluzione della trasformata. Un discorso a parte va fatto per forme d'onda che presentano toni molto vicini tra loro, ossia che causano l'effetto noto come "Harmonic Interference" (cfr. cap. VII).

Come detto, le soluzioni adottabili sono molteplici; si può utilizzare lo strumento a zero-crossing, o quello FFT-based. Facciamo delle considerazioni che ci aiutino a capire come sia stata implementata questa seconda soluzione.

In prima battuta l'idea potrebbe essere quella di utilizzare la FFT normalmente calcolata dallo strumento analizzatore di spettro, e ricavare da essa velocemente la misura della frequenza; per migliorare la risoluzione si può pensare di aumentare il numero di punti del buffer. Questa ultima soluzione comporta svariati problemi; in primis, un aumento del buffer porta ad un intervallo tra due misure che si allunga proporzionalmente alla dimensione del buffer, indipendentemente dai tempi di calcolo; per esempio un buffer di 4096 punti ad una frequenza di campionamento di 40960 Hz è "riempito" ogni 100 mS dalla scheda sonora, e dunque, ammesso di riuscire a calcolare le misure desiderate in questo tempo, avremo 10 misure al secondo. Se il buffer è portato a 8192 avremo un tempo di 200mS ed una risoluzione di 5,38 Hz, e dunque 5 misure al secondo. Se volessimo ulteriormente migliorare la risoluzione, per esempio per arrivare a risoluzioni dell'ordine dell'unità o meno potremmo usare un buffer di 65536 punti ossia arrivare a 0.67 Hz di risoluzione. I tempi necessari a riempire un buffer salgono però a 1.48 secondi, ancora accettabili per una misura di frequenza.

Figura 30: finestra frequenzimetro (thread)

Il secondo problema che si presenta sfruttando questo approccio è più subdolo e strettamente dipendente dalla macchina; l'uso della FFT consente di spuntare tempi di esecuzione dell'algoritmo che sono dell'ordine di *nLog(n)* dove n numero di punti del buffer, al contrario di un ordine proporzionale a n^2 ottenibile con l'uso diretto della trasformata di Fourier; tuttavia, si osserva sperimentalmente che il calcolo di una FFT su un moderno dual-core è dell'ordine di 10 mS per un buffer di 4096 punti ma sale a valori di qualche secondo per buffer che raggiungono o superano il valore 65536, ossia dunque, per alcune macchine non si riesce a finire il calcolo prima dell'arrivo di un successivo buffer di punti, e questo è inaccettabile per molti strumenti ed in particolare per un frequenzimetro.

La soluzione adottata a questi problemi parte dall'assunto: non usare l'FFT calcolata dall'analizzatore di spettro; usare un thread con priorità medio bassa rispetto al thread principale di acquisizione campioni, ed effettuare uno zero-padding su un solo buffer di punti in maniera da ottenere il calcolo di una FFT su un buffer di dimensione arbitraria, che ha come ulteriore vantaggio di non dover aspettare il riempimento "reale" del buffer di punti. In altre parole, dato un buffer di punti qualsiasi, si aggiunge una quantità di zeri proporzionale alla risoluzione desiderata e si lancia un thread che calcola l'FFT, e dunque la frequenza a massima ampiezza, "quando c'è tempo", presentando a video i risultati e segnalando tramite un lampeggio di un led simulato che la misura è stata aggiornata (v. fig. 30). Si rileva sperimentalmente che alle massime risoluzioni (es. 0.16 Hz) i tempi di calcolo si aggirano sull'ordine dei secondi, e dunque in linea con gli strumenti reali, ed in più senza rallentare apprezzabilmente il funzionamento degli altri strumenti eventualmente in esecuzione.

L’incertezza associata alla misura della frequenza è dovuta naturalmente a molti fattori (cfr. VII) ma una componente sicuramente presente è quella dovuta alla risoluzione discreta, come già accennato; in tal caso essa viene modellata come un errore casuale a distribuzione uniforme, ossia è ragionevole supporre che la frequenza da calcolare possa ricadere con uguale probabilità in uno qualsiasi dei valori compresi nell’intervallo tra due frequenze adiacenti. Esso è pertanto modellato secondo una distribuzione uniforme la cui deviazione standard u^2, dato δx come intervallo, può semplicemente calcolarsi come:

$$u^2 = (\delta x)^2 / 12 \tag{5.3}$$

da cui è facile calcolare l’incertezza tipo come:

$$u = 0{,}29\ \delta x \tag{5.4}$$

e dunque una risoluzione (δx) di 0,16 Hz implica una incertezza tipo di 0.0464 Hz, mentre con un δx di 5 Hz otteniamo un’incertezza tipo di 1.49 Hz; l’incertezza estesa corrispondente, utilizzando un fattore di copertura prudenziale pari a 2 (cfr cap. II), risulta dunque di circa 3 Hz.

5.6 Generatore di funzioni

Il generatore di funzioni è in realtà costituito da una moltitudine di thread, utilizzati anche in altri strumenti che si avvalgono della generazione locale di forme d’onda (es. ZRLC, THD) allocati e mandati in esecuzione solo quando lo strumento viene effettivamente abilitato alla generazione; ossia, quando si apre la finestra del generatore di funzioni il task risulta ancora non istanziato; solo dopo la pressione del tasto “on” esso viene effettivamente allocato e mandato in esecuzione con i parametri selezionati (frequenza, forma d’onda, sfasamento, modalità di generazione eccetera). Esistono più thread relativi a questa funzione; essi funzionano tutti con il medesimo principio e sono stati aggiunti solo per comodità di programmazione e ragioni di efficienza e velocità di esecuzione; pertanto descriveremo il funzionamento comune a tutti senza addentrarci in ulteriori dettagli, tranne quando indispensabile.

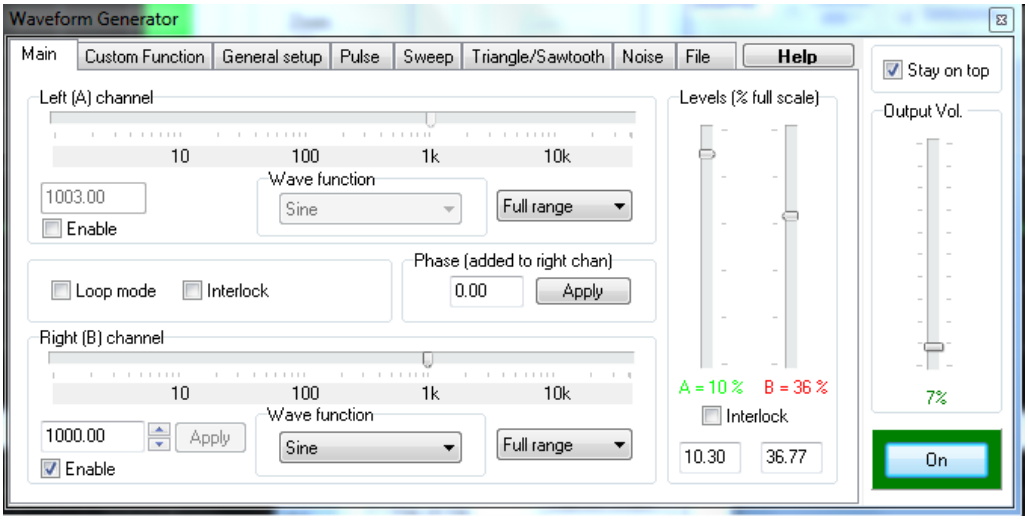


Figura 31: finestra generatore di funzioni, schermata principale (thread)

Il funzionamento generale del generatore di funzioni è simmetrico di quello di tutte le altre funzioni del programma, ossia la generazione delle forme d'onda (sintesi) implica un percorso inverso dei dati rispetto a quello di tutti gli altri strumenti. Ossia essi vanno dal PC alla scheda, invece che dalla scheda al PC. In altre parole ancora, nel generatore di funzioni il segnale è appunto generato, mentre per gli altri strumenti è misurato, ossia letto.

Le forme d'onda vengono generate da algoritmi interni che sono realizzati interamente in software; i campioni generati vengono calcolati tenendo presente la frequenza di campionamento selezionata e le dimensioni del buffer da riempire. Il meccanismo è illustrato dalla figura 32, dove è sinteticamente indicato il processo di generazione di una forma d'onda sinusoidale.

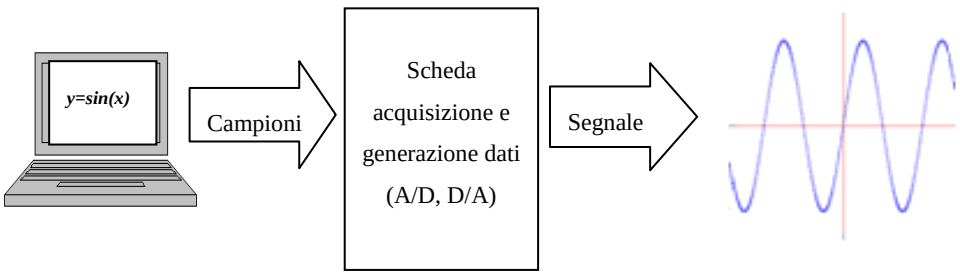


Figura 32: principio di funzionamento del generatore

I campioni vengono generati nel thread, e passati alla scheda sonora tramite RAM e bus di comunicazione (es. USB); la scheda sonora interpreta questi campioni come ottenuti da campionamento di forma d'onda analogica, e li converte tramite il convertitore D/A interno secondo i parametri definiti, ossia la frequenza di campionamento indicata.

Le forme d'onda che vengono generate sono prive di aliasing; infatti, il fenomeno dell'aliasing è in agguato anche nel caso della sintesi di forme d'onda; i campioni calcolati dall'algoritmo equivalgono a campioni ottenuti dal campionamento di una forma d'onda che, secondo quanto prescritto dal teorema di Shannon-Nyquist, dovrebbe essere limitata in banda, e dunque ottenere una forma d'onda priva di distorsione da sovrapposizione di spettri periodici. Per esempio, nel caso di generazione di una semplice sinusoide, come indicato sommariamente nella figura 32, è semplice ottenere una sinusoide relativamente indistorta, usando le funzioni di libreria del C ed eventualmente utilizzando algoritmi aggiuntivi di dithering per migliorare il processo di conversione digitale-analogico. Discorso diverso e più delicato per quanto riguarda la sintesi di forme d'onda più complesse, quali per esempio quadre, triangolari, dente di sega eccetera. Od addirittura completamente definibili dall'utente, ossia arbitrarie.

La maggior parte dei programmi reperibili in rete, anche di livello professionale, si limita a sintetizzare i campioni a partire da una sorta di "disegno" della forma d'onda desiderata. Per esempio, per un'onda quadra, sarebbe facile sintetizzare i campioni secondo la semplice formula "genera 1 se il campione è nel primo semiperiodo, genera -1 se il campione è nel secondo semiperiodo", eventualmente stabilendo un criterio per il punto "zero". In siffatta maniera, che si potrebbe facilmente estendere a forme d'onda di qualsiasi tipo, per esempio "disegnate" a video tramite un editor grafico (come spesso si trova in molti programmi), si sta a tutti gli effetti campionando una forma d'onda probabilmente *non limitata in banda* e dunque si sta introducendo una considerevole distorsione (aliasing), che si può facilmente osservare con lo strumento analizzatore di spettro; lo spettro d'ampiezza risulta

evidentemente “sporcato” da una discreta quantità di armoniche aggiuntive che non fanno parte del segnale desiderato.

In letteratura è possibile trovare algoritmi che garantiscono la generazione di forme d’onda prive di aliasing per qualche tipologia di segnale (tipicamente quadra e triangolare). Da prove pratiche effettuate i risultati non sono stati affatto soddisfacenti, e spesso l’aliasing risulta solo limitato ma non eliminato.

L’idea, tanto semplice quanto evidente, è quella di generare le principali forme d’onda predefinite utilizzando direttamente lo sviluppo in serie di Fourier, ossia utilizzare la ben nota relazione:

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{L} + b_n \sin \frac{n\pi x}{L} \right) \qquad (5.5)$$

In particolare essa è stata utilizzata per generare onde quadre e triangolari; per queste ultime è stato previsto il calcolo variabile (in tempo reale) dei punti di discontinuità (Dirichlet) in maniera da poter far variare con continuità l’onda da “dente di sega” a onda “triangolare” (v. figura 33 per segnale e spettro corrispondente).

E’ altresì evidente che la relazione 5.5 non può essere effettivamente implementata in un calcolatore (la sommatoria è infinita) e bisogna dunque accontentarsi di un’approssimazione, ossia limitare il numero di armoniche utilizzate; questo sia per motivi di puro e semplice tempo di calcolo, che per motivi relativi alla completa inutilità della cosa: se l’hardware utilizzato è limitato in banda, diverrebbe completamente inutile aggiungere il contributo di armoniche che comunque non potrebbero essere contenute nel segnale.

Nella versione realmente applicata la 5.5 è limitata ad un numero di armoniche calcolate in funzione della massima frequenza ammissibile dall’hardware utilizzato e/o, nel caso di forme d’onda arbitrarie (v. par. 5.7.2), dal numero di armoniche desiderato.

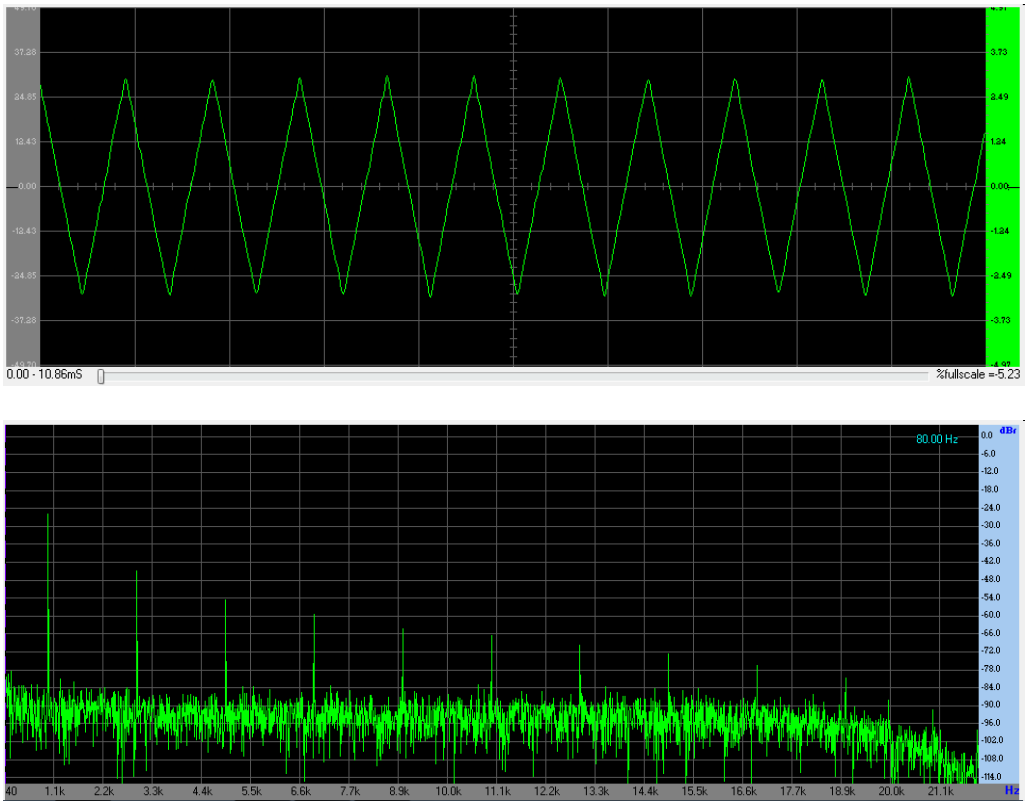


Figura 33: onda generata (1000 Hz) e suo spettro (senza aliasing)

Questo approccio implica un notevole impegno computazionale, in realtà meno consistente di quello previsto, giacché con opportuni accorgimenti e uso oculato del meccanismo del multithread e delle bufferizzazioni, si riesce ad utilizzare nella quasi totalità delle applicazioni. Inoltre, molti moderni microprocessori multi-core hanno le operazioni di seno e coseno direttamente definite in hardware e dunque eseguibili al prezzo di pochissimi cicli di clock.

Prima dunque di addentrarci nella descrizione più dettagliata di come si è realmente implementato il meccanismo di generazione, in particolare per le forme d'onda completamente arbitrarie, dobbiamo descrivere due diverse modalità di generazione che il programma mette a disposizione, compatibilmente con le capacità della scheda utilizzata:

- Generazione a "loop" infinito;
- Generazione "tempo reale".

5.6.1 Generazione loop infinito

Talune schede di acquisizione, e non ultime le scheda sonore, consentono una modalità di funzionamento in completo parallelismo hardware, che ben si presta per un ottimale funzionamento in molte applicazioni pratiche; infatti, è consentito scaricare una determinata quantità di campioni in un buffer interno alla scheda stessa, e dunque attivare il processo di conversione digitale-analogico in un loop infinito circolare su quel gruppo di campioni. E' evidente come questo sia devoluto ad hardware interno della scheda, e consenta di svincolare completamente la CPU del PC ed il software applicativo stesso (VA) da qualsiasi onere computazionale, avendo in tal senso a disposizione un generatore separato a livello hardware. Per contro vengono a mancare alcune caratteristiche che in determinate situazioni sono invece indispensabili: non è possibile cambiare i parametri principali in tempo reale (per esempio frequenza, fase, forma d'onda) e non è sempre possibile generare più forme d'onda contemporanee, su canali diversi, di diversa frequenza; ed inoltre, data una frequenza di campionamento ed una frequenza (per esempio decimale) da generare non è sempre possibile, matematicamente, calcolare un numero di campioni che comprenda un numero di cicli che si raccordino in maniera circolare. Infatti, essendo tale buffer eseguito in un loop infinito, ossia circolarmente, è necessario che la forma d'onda sia "richiudibile" circolarmente senza discontinuità, altrimenti verrà introdotto un "salto" e dunque una distorsione (ossia introdurremo armoniche indesiderate). E questo in maniera consistente anche se si "salta" un solo campione. In altre parole la forma d'onda deve "finire" esattamente dove "comincia" in maniera da apparire come una forma d'onda infinita.

Questa modalità di generazione viene dunque utilizzata laddove non serve poter effettuare generazioni a frequenze del tutto arbitrarie, e non vi sia bisogno di variare parametri durante il funzionamento. Ed inoltre dove non serva generare più forme d'onda differenti su canali diversi: la maggior parte delle schede non consente di definire buffer di dimensioni diverse per i diversi canali di generazione. Questo ultimo punto sempre relativamente allo stato attuale di nostra conoscenza dei prodotti

presenti sul mercato, ed in ogni caso al momento in cui viene stampato il presente lavoro.

5.6.2 Generazione tempo reale

Questa modalità di generazione è quella che si avvicina in maniera quasi esaustiva a quella di uno strumento realizzato in hardware e che presenta la maggiore flessibilità possibile; il tutto al prezzo di una maggiore complessità computazionale, che per talune macchine potrebbe essere eccessivamente "resource-consuming". E' altresì da osservare che durante l'evoluzione di VA, l'hardware di base dei PC si è parimenti evoluto: la stragrande maggioranza dei PC sono oramai a CPU multiple (multi-core) e con un numero considerevole di funzioni matematiche definite in hardware (per esempio le funzioni trigonometriche principali). In tal senso, test effettuati in molte condizioni operative, rendono indistinguibile l'utilizzo dell'una o l'altra modalità; e dunque si preferisce usare quest'ultima pressocché ovunque, tranne ove rilevate evidenti problematiche di scarsità di risorse. Si noti anche come le opzioni disponibili nel "setup" del generatore di funzioni, relativamente ai parametri della modalità in tempo reale, consentono di adattare il generatore di funzioni alle situazioni operative che si vengono a verificare, e dunque re-distribuire opportunamente il carico computazionale. Quest'ultimo punto apparirà chiaro nel corso del presente paragrafo.

L'idea alla base di questa modalità di generazione delle forme d'onda è di realizzare in forma inversa quanto accade in fase di acquisizione dati (cfr. par. 1); ossia, creare un "pool" di buffer, generati in un ciclo infinito, contenenti i campioni del segnale. Dunque, dedicare un thread alla generazione continua del segnale, che riempie dei vettori di dimensione prefissata con i campioni e li passa alla scheda sonora.

Il processo è sintetizzato dalla figura 34, dove per semplicità si fa riferimento alla generazione di una sinusoide.

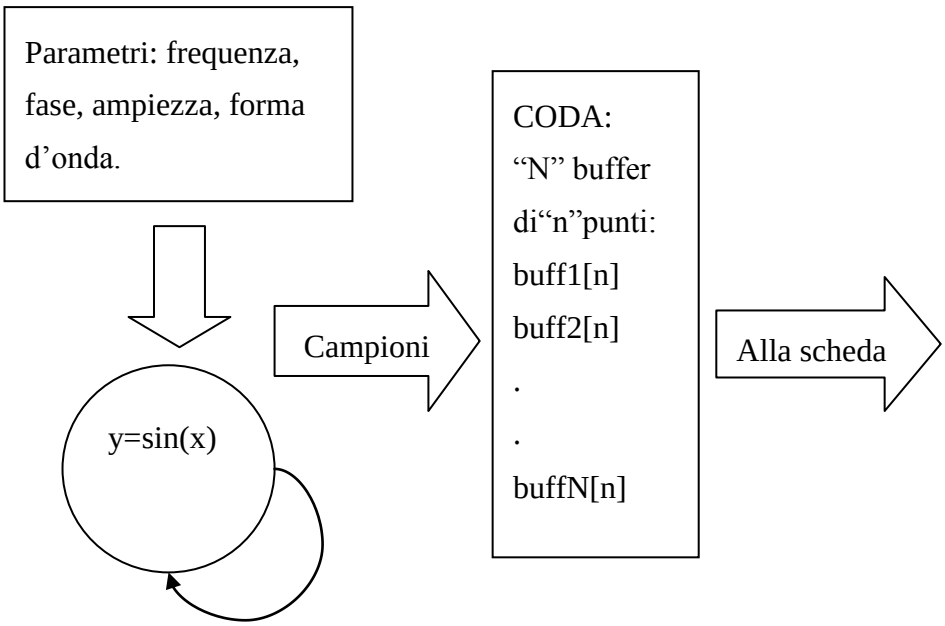


Figura 34: generatore in tempo reale

Il thread di generazione è costituito da un ciclo infinito, i cui parametri sono modificabili dall'utente in tempo "quasi" reale (questa espressione verrà presto motivata); esso genera continuamente i campioni del segnale, che vengono "pacchettizzati" e inviati alla scheda sonora su base continua. I tempi di reazione (latenza) di questo meccanismo sono ovviamente strettamente legati alla frequenza di campionamento scelta, alle dimensioni del buffer e ad numero degli stessi. Fissiamo le idee con un esempio. Supponiamo di aver scelto i seguenti parametri:

Frequenza di campionamento: 44100 Hz

Buffer: 4410 punti

Numero di buffer: 5

I campioni sono dunque generati con intervallo temporale l'uno dall'altro pari a $1/44100 = 2.27 \cdot 10^{-5}$ secondi; ogni singolo buffer rappresenta dunque $4410 \cdot 2.27 \cdot 10^{-5} = 0.1$ secondi di segnale. E dunque moltiplicando per il numero di buffer otteniamo $0.1 \cdot 5 = 0.5$ secondi. Ossia, una eventuale variazione di uno dei parametri sarà effettivamente applicata al più dopo 0.5 secondi, giustificando la frase "tempo quasi reale". Questo tempo di ritardo, che definiremo "tempo di latenza" può essere

minimizzato riducendo il numero di buffer e/o il numero di campioni per buffer, ma non può essere comunque ridotto a zero. In ogni modo reso molto piccolo, e scelto in dipendenza delle esigenze applicative e dalle risorse del sistema.

Si osservi che per forme d'onda complesse, che richiedono il calcolo simultaneo di svariate funzioni sinusoidali e co-sinusoidali, è possibile, almeno in via teorica, sfruttare particolari funzioni rese disponibili su alcune schede sonore. In particolare, la capacità di alcune schede di eseguire in perfetto parallelismo hardware più forme d'onda contemporaneamente (polifonia delle schede sonore). Questa funzione è disponibile da tempo anche su schede sonore "entry level", tuttavia esso è assente su schede di acquisizione/generazione di tipo professionale e dedicate alle misure perché è una caratteristica tipica delle schede destinate ad uso musicale. Nella presente versione del programma, per motivi di generalità e massima compatibilità, si è preferito non utilizzare questa ulteriore modalità di generazione.

5.7 Generazione forme d'onda

Visual Analyser consente di generare forme d'onda tramite i meccanismi descritti nei paragrafi precedenti, e dunque implementa uno strumento virtuale chiamato "generatore di funzioni". In particolare, è possibile utilizzare un insieme di funzioni predefinite oppure definire delle forme d'onda arbitrarie a partire dal loro sviluppo in serie. Quest'ultimo è impostabile manualmente, tramite l'impostazione dei vari coefficienti dello sviluppo in serie di Fourier, oppure definibile graficamente tramite una funzione aggiuntiva che consente di impostare i parametri dello sviluppo in serie (tramite appositi "controlli") ed in tempo reale osservare la forma d'onda man mano costruita.

5.7.1 Le forme d'onda predefinite

Le forme d'onda predefinite sono selezionabili dalla finestra principale del generatore di funzioni (v. fig. 11).

In particolare esse sono:

1) Sinusoidale
2) Quadra
3) Triangolare
4) Rumore bianco
5) Rumore rosa
6) Sweep sinusoidale
7) Livello in continua positivo e negativo
8) Quadra aliasing
9) Impulsi

Alle quali si aggiunge una nutrita serie di opzioni accessibili dalla stessa finestra del generatore di funzioni (in "tab" adiacenti a quello principale) che per semplicità di trattazione omettiamo di descrivere. Quello che interessa evidenziare in questo paragrafo sono gli algoritmi utilizzati per la sintesi delle forme d'onda di cui ai punti 1..6, essendo la (7) un semplice livello in continua, la (8) e (9) realizzate con un "non-algoritmo", ossia tramite "disegno" a video di forma d'onda (v. par. 5.6).

Sinusoide

La generazione della forma d'onda sinusoidale non comporta particolari problemi, ed è effettuata tramite l'utilizzo diretto delle funzioni trigonometriche di libreria disponibili nel compilatore utilizzato; tramite essa viene implementata anche la generazione della forma d'onda quadra e triangolare, ed in generale di tutte, tranne il rumore bianco e rosa, come vedremo a breve.

Onda quadra

L'onda quadra è ottenuta applicando direttamente lo sviluppo in serie di Fourier, ossia la relazione (5.5) qui riportata per comodità:

$$f(x) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos \frac{n\pi x}{L} + b_n \sin \frac{n\pi x}{L} \right)$$

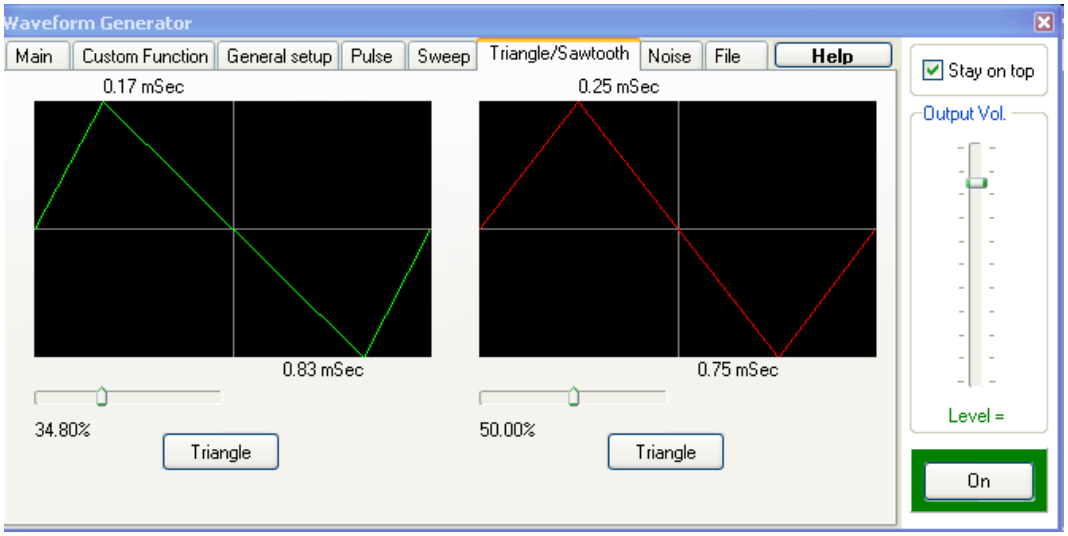
Se consideriamo una forma d'onda quadra dispari e con duty-cycle del 50% la relazione finale conterrà esclusivamente termini sinusoidali, e può essere scritta come (omettendo per semplicità il fattore dell'ampiezza):

$$\psi_{\mathbf{quadra}}(t) = \frac{4}{\pi} \sum_{k=1}^{\infty} \frac{\sin\left((2k-1)2\pi f t\right)}{(2k-1)} \tag{5.6}$$

Dunque, fissata la frequenza "f" e l'ampiezza calcoleremo ogni singolo punto della forma d'onda tramite la somma (5.6) estendendo la sommatoria sino all'ultimo termine valido per la banda passante disponibile. Per esempio, con una scheda di acquisizione/generazione avente una banda passante di 20 kHz, ed una frequenza di 1000 Hz, potremo arrivare sino a k = 19.

Triangolare e dente di sega

La forma d'onda triangolare è stata realizzata tramite uno sviluppo in serie di Fourier, calcolato definendo come parametri i punti angolosi che determinano il passaggio a differenti pendenze del periodo; in tal modo è possibile, tramite impostazione dei punti angolosi passare con continuità da una forma d'onda puramente triangolare ad una a dente di sega, il tutto assolutamente in tempo reale, ossia mentre la stessa è generata e utilizzata. La figura successiva mostra come sia possibile variare tramite dei controlli visuali (trackbar) la forma d'onda generata.



Nella figura, la forma d'onda in rosso è puramente triangolare, mentre quella in verde (i cui punti angolosi sono stati impostati a 0.17 e 0.83 mS, relativamente ad un campionamento pari a 40960 Hz e una frequenza del segnale di 1000Hz) si avvia a divenire a dente di sega. L'impostazione di questi valori, nel caso della finestra mostrata in figura, è stata effettuata tramite trackbar e mentre la generazione del segnale era in corso. La forma d'onda generata è assolutamente priva di aliasing in ogni configurazione scelta (determinazione automatica dei coefficienti dello sviluppo in serie di Fourier in tempo reale).

Rumore bianco

Il generatore di rumore bianco è stato banalmente ottenuto sfruttando le funzioni di libreria, e consentendo di scegliere tra due differenti distribuzioni (gaussiana e uniforme), sebbene future estensioni consentiranno di scegliere tra un numero ben maggiore di distribuzioni. E' anche possibile impostare la media e la varianza.

L'uso di algoritmi tipo Box-Muller per la generazione di distribuzioni gaussiane non ha portato significativi miglioramenti in termini di qualità del segnale ottenuto (probabilmente perché le funzioni di libreria già fanno uso del medesimo algoritmo!), e dunque si è optato per l'uso delle funzioni di libreria, che tra le varie cose, sembrano essere ottimizzate in termini di velocità di esecuzione (scritte in linguaggio assemblativo).

Rumore rosa

Il generatore di rumore rosa è stato ottenuto a partire dal generatore di rumore bianco, con distribuzione uniforme, applicando ad esso un filtro di così definito:

```
b0 = 0.99886 * b0 + white * 0.0555179;
b1 = 0.99332 * b1 + white * 0.0750759;
b2 = 0.96900 * b2 + white * 0.1538520;
b3 = 0.86650 * b3 + white * 0.3104856;
b4 = 0.55000 * b4 + white * 0.5329522;
b5 = -0.7616 * b5 - white * 0.0168980;
pink = b0 + b1 + b2 + b3 + b4 + b5 + b6 + white * 0.5362;
b6 = white * 0.115926;
```

dove con “white” è da intendere un campione di rumore bianco a distribuzione uniforme e “pink” il valore del singolo campione del segnale “rumore rosa”.

L’uso di questo filtro dovrebbe garantire prestazioni accettabili; una variante che fa uso dell’algoritmo di Voss non sembra fornire prestazioni migliori. Tramite questo semplice filtro, i cui valori sono calcolati per una frequenza di campionamento di 44100 kHz, dovrebbe fornire un’accuratezza nella generazione del rumore rose entro ±0.05 dB oltre la frequenza di 9.2 Hz,con una pendenza del filtro di circa -10 dB/decade.

Sweep sinusoidale lineare

Questa funzione consente di ottenere uno “sweep” in frequenza (noto anche come segnale “chirp”) di tipo sinusoidale, sebbene sia possibile ottenere uno sweep in frequenza di una forma d’onda base qualsiasi. Il termine inglese “chirp” (cinguettare) deriva dal fatto che in campo audio un segnale chirp sinusoidale ha un suono simile a quello di un cinguettio.

La funzione che definisce un segnale di questo tipo è:

$$F(t) = F_0 \ \sin \Phi(t), 0 \leq t \leq T \tag{5.7}$$

Dove T è il periodo del segnale e Φ la frequenza. Appare evidente che l’ampiezza di questo segnale è costante, e che la frequenza varia con il tempo. La variazione della frequenza può avvenire in diverse modalità, quella più semplice è ovviamente di tipo lineare; cerchiamo dunque di ottenere una variazione di tipo lineare. La variazione istantanea della frequenza con il tempo è data da:

$$\omega(t) = \frac{d\phi(t)}{dt} \tag{5.8}$$

Per ottenere una variazione istantanea lineare useremo la seguente relazione:

$$F(t) = F_0 \ \sin(\mathrm{at}^2 + \mathrm{bt}), 0 \leq t \leq T \tag{5.9}$$

La cui variazione istantanea di frequenza è, applicando la 5.8:

$$\omega(t) = \frac{d(\mathrm{at}^2 + \mathrm{bt})}{dt} = 2at + b \tag{5.10}$$

Ossia di tipo lineare come desiderato. Per essa imposteremo:

$$a = \frac{\omega_2 - \omega_1}{2T}, b = \omega_1 \quad (5.11)$$

Dove con ω_1 e ω_2 si indicano le pulsazioni della frequenza iniziale e finale dello sweep. Come detto, è dunque in generale possibile ottenere uno sweep con caratteristiche diverse, per esempio a partire da forme d'onda base differenti (es. quadra, triangolare) e con caratteristiche di variazione della frequenza diverse da lineari (es. quadratiche o logaritmiche). Future estensioni, dettate da esigenze di laboratorio, potranno pertanto comprendere nuove funzioni di sweep.

5.7.2 Forme d'onda arbitrarie

Per garantire la massima generalità e flessibilità, è stata prevista la possibilità di definire una forma d'onda completamente arbitraria e priva di aliasing. Per ottenere quest'ultimo punto, non ci si è limitati a consentire un semplice "disegno a video" della forma d'onda con conseguente generazione distorta da evidente aliasing, ma si è impostato un meccanismo che consente di costruire la forma d'onda a partire dal suo sviluppo in serie di Fourier.

Questo è possibile essenzialmente in due modalità; la prima consente di impostare in maniera semplice i coefficienti dello sviluppo in serie (in termini di frequenza e ampiezza) che pertanto devono essere noti (ossia bisogna calcolarlo anticipatamente, come è ovvio).

La seconda modalità consente di effettuare le stesse operazioni della prima, ma con la possibilità di immediata visualizzazione della forma d'onda costruita (anteprima) e immissione semplificata dei parametri tramite "controlli" di Windows (Trackbar, bottoni, etc.). In entrambi i casi è possibile memorizzare la forma d'onda in un file che può essere caricato manualmente in qualsiasi momento oppure automaticamente alla partenza del programma; in tal senso è possibile crearsi una "libreria" di forme d'onda arbitrarie.

In entrambe le modalità è possibile definire una "modulazione" della forma d'onda ottenuta, sebbene essa sia ottenibile direttamente in sede di calcolo dei coefficienti.

Impostazione forme d'onda arbitrarie

Per semplicità si suppone di avere forme d'onda definite da somme di seni; è immediato riportare lo sviluppo in serie di Fourier a semplici somme di seni, con evidenti assunzioni di simmetria della forma d'onda. I parametri che si possono impostare sono l'ampiezza del singolo coefficiente, la fase e la frequenza. La figura seguente mostra la finestra che consente di impostare la forma d'onda "custom":

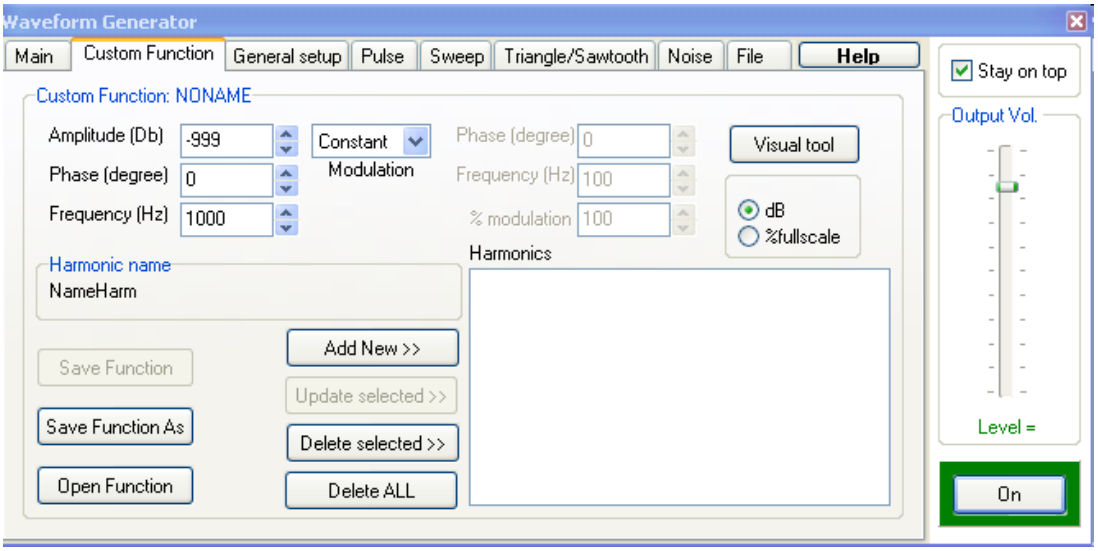


Figura 35: impostazione forma d'onda arbitraria

Impostazione con "tool visuale"

Questa funzione consente di impostare gli stessi parametri della funzione precedente, ma tramite ausili grafici che consentono una anteprima "in tempo reale" intendendo in questo caso che ad ogni aggiunta di un coefficiente si ottiene immediata anteprima della forma d'onda creata. La figura che segue, mostra il "Visual Tool" in fase operativa, dove è stata impostata una forma d'onda semplice, costituita da due sinusoidi, di frequenza rispettivamente di 50 e 100 Hz, con ampiezza relativa di -3.66 e -9.54 dB. E' stata prevista la possibilità di impostare sino a 20 armoniche (ma future estensioni possono prevedere numeri ben maggiori).

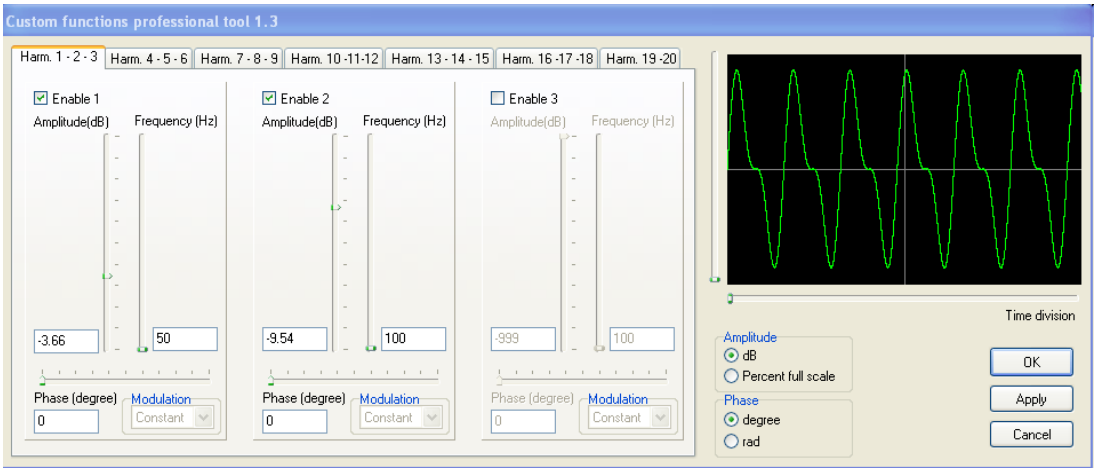


Figura 36: visual tool

5.8 Conversione D/A

Il titolo di questo paragrafo potrebbe indurre facilmente in errore, in quanto esso sembra argomento relativo ad altre sezioni (per esempio a discussioni sulla scheda di acquisizione e hardware relativo). In verità, come si vedrà nel prosieguo del paragrafo, esso tratta proprio di una conversione Digitale – Analogico realizzata a livello software da due thread separati (uno per ogni canale) e costituisce un esempio particolarmente significativo del concetto "sostituire hardware con del software" e dunque dell'essenza di uno strumento virtuale.

Rimandiamo al cap. III per spiegazioni dettagliate sul teorema del campionamento e argomenti correlati, e ad [1] per descrizioni approfondite sul codice implementato in VA; limitiamoci in questa sede a descrivere l'architettura del thread ed alla descrizione della funzionalità implementata, rifacendoci esplicitamente anche a [1]. Rispetto a quanto descritto in questa ultima fonte, il thread ha subito delle notevoli variazioni, sia per quanto riguarda il fattore prestazionale che sulle politiche di gestione del thread stesso; tuttavia, la descrizione che segue resta ancora compatibile con [1] ed anzi da essa liberamente ispirata.

Ogni programma simile a Visual Analyser utilizza i campioni numerici prelevati dai buffer interni della scheda sonora; essi sono limitati in banda (da un filtro anti-

aliasing presente nella scheda stessa) e campionati ad una determinata frequenza stabilita dall'utente. I campioni così ottenuti rappresentano completamente il segnale acquisito (Shannon-Nyquist), e vengono pertanto utilizzati per rappresentare a video il segnale (per esempio) nella finestra oscilloscopio.

La tecnica utilizzata normalmente consiste nel disegnare a video i punti acquisiti raccordandoli con un segmento di retta, ottenendo una rappresentazione veloce da disegnare e relativamente "simile" al segnale analogico da essi rappresentato. Invero, questo va bene per frequenze fondamentali del segnale molto minori della massima frequenza ammessa; tipicamente minore di 1/5 .. 1/6 della frequenza di Nyquist. Per fissare le idee, un segnale campionato a 10 kHz avrà una banda passante che si estende sino a 5 kHz; una sinusoide a 1000 Hz sarà pertanto rappresentata da 10 punti a ciclo (=10 kHz /1000Hz).

Questo significa che per ogni ciclo del segnale avremo 10 punti raccordati da segmenti di retta. Ossia un disegno la cui approssimazione del segnale reale analogico (composto da infiniti punti) è ancora chiaramente percettibile, ma parimenti tollerabile (figura 37).

Se visualizziamo un segnale a 100Hz la situazione sarà ancora migliore (figura 38) avendo a disposizione ben 100 punti per ciclo (in questo caso, vista la risoluzione "discreta" dell'accoppiata scheda grafica/monitor, esso risulta praticamente indistinguibile dal segnale originale).

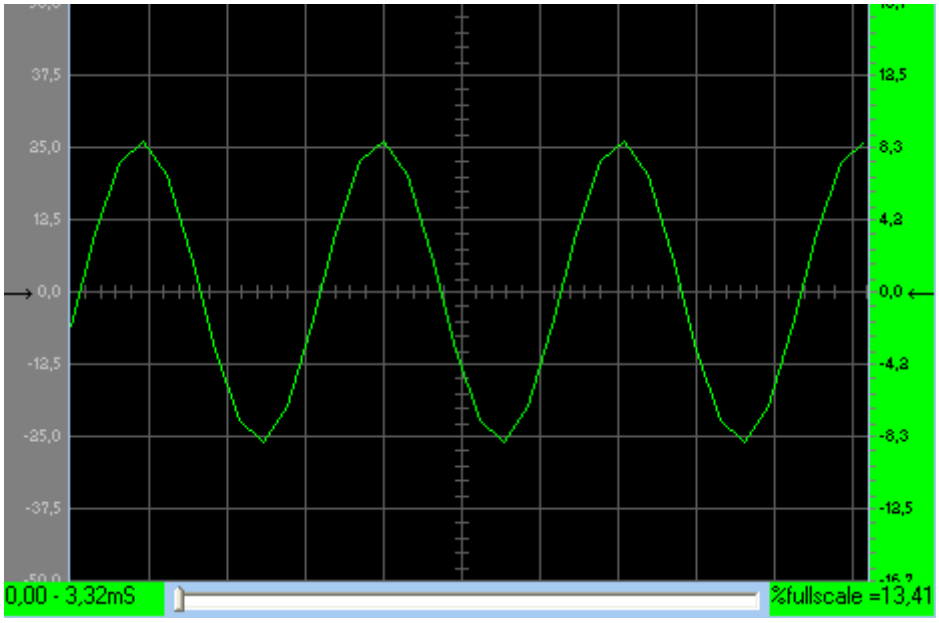


Figura 37: sinusoide 1000 Hz campionata a 10Khz

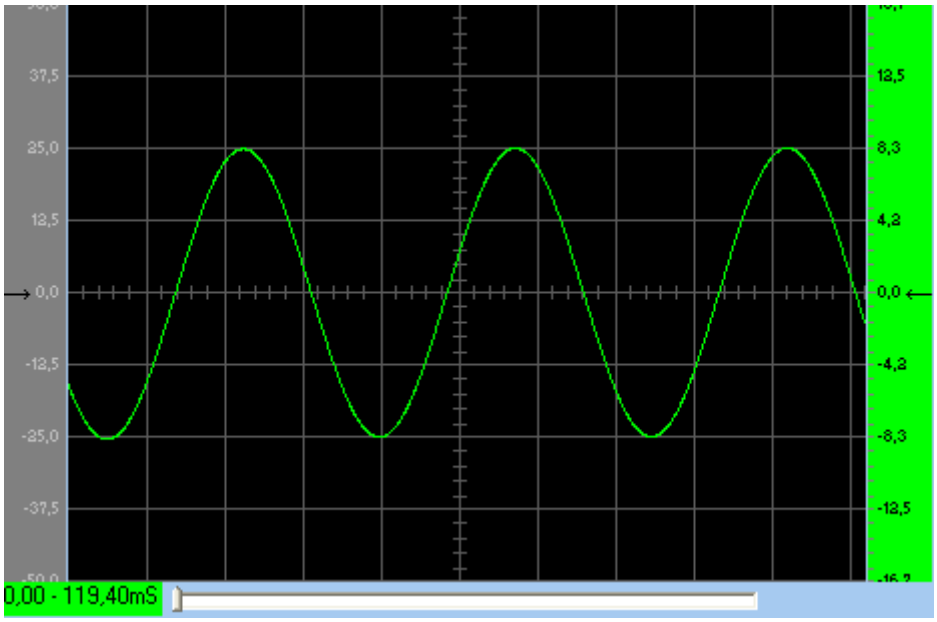


Figura 38: sinusoide 100Hz campionata a 10 kHz

La situazione cambia drasticamente all'approssimarsi degli estremi superiori della banda passante. Consideriamo il caso limite di un segnale a 5000 Hz: esso sarà disegnato avendo a disposizione solo due punti per ciclo, del tutto insufficienti per poter disegnare direttamente il segnale a video. Esso apparirà infatti come un'onda triangolare (figura 39). La situazione migliora drasticamente spuntando la checkbox

"D/A" relativa al canale corrispondente (in questo caso il sinistro) come si vede in fig. 40. A fronte di una maggiore complessità computazionale, il segnale originale viene ricostruito esaustivamente tramite l'algoritmo che verrà descritto nei capoversi successivi.

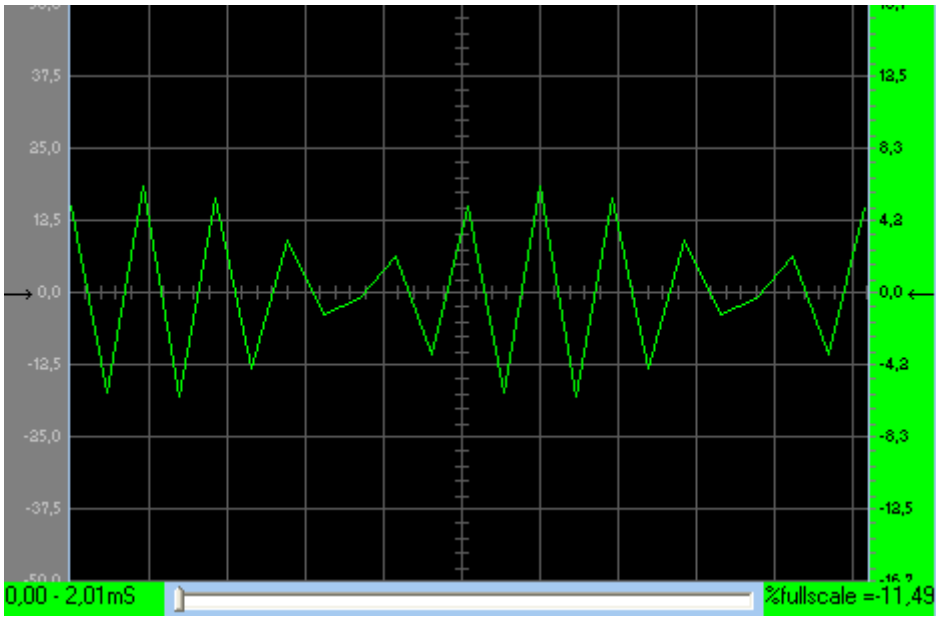


Figura 39: sinusoide a 5000 Hz campionata a 10 kHz

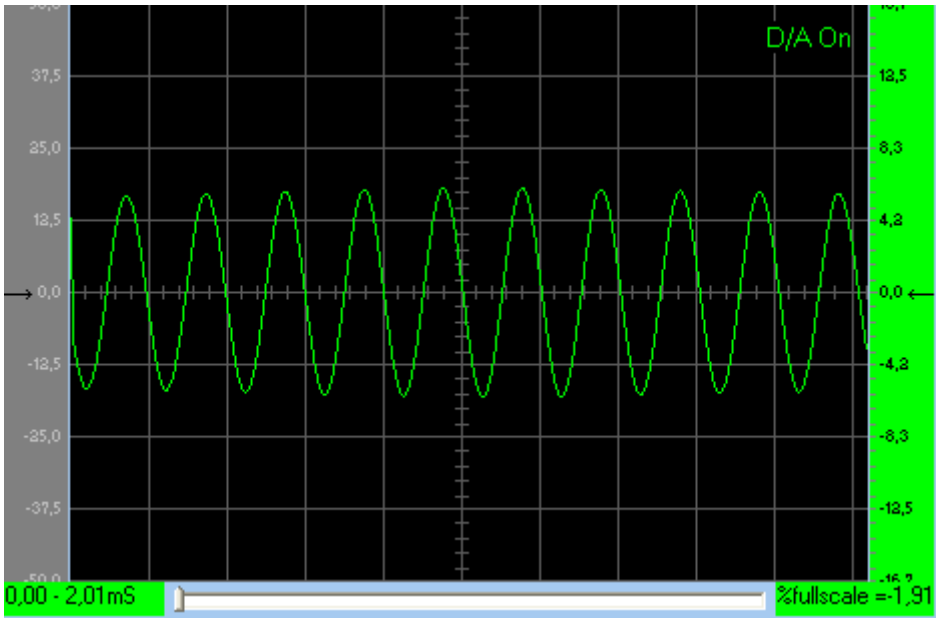


Figura 40: sinusoide a 5000 Hz campionata a 10Khz con conversione D/A attivata

Se da un lato il teorema del campionamento ci insegna che le informazioni per ricostruire completamente il segnale analogico ci sono tutte, è ovvio che per far ciò bisogna applicare correttamente e completamente l'algoritmo. Cosa teoricamente fattibile, ma in pratica troppo onerosa a livello computazionale. Visual Analyser riesce tuttavia nell'intento.

Vediamo ora [1] in maggiore dettaglio come riuscire a risolvere il problema, come opera il thread, e perché si rende necessario utilizzare un thread.

I punti disponibili per la conversione digitale-analogica sono quelli relativi ad un buffer, e sono rigorosamente una potenza di due, per esigenze di altri algoritmi (FFT). La dimensione tipica di un buffer è di 4096 punti (ma definibile a piacere tra un set di valori predefiniti o manuali), e la dimensione standard della finestra oscilloscopio è dell'ordine dei 400 pixel (sebbene possa arrivare sino alla massima risoluzione orizzontale consentita dalla scheda video, ed in generale sino ad un valore completamente arbitrario). La formula da applicare è la 5.6, qui ripetuta per comodità, che implica una sommatoria (al limite infinita) estesa comunque *almeno* a tutti i punti del segnale disponibile e ripetuta per ogni punto del segnale da ricostruire.

$$x(t) = \sum_{n} x(nT_c) \sin c(\pi \frac{t - nT_c}{T_c}) \tag{5.6}$$

La prima considerazione da fare è che bisogna ripetere la sommatoria per almeno 400 volte su 4096 punti. Ossia avremmo 4096*400 = 1.638.400 iterazioni che comprendono:

- 1.638.400 divisioni;
- 4.915.200 moltiplicazioni;
- 1.638.400 operazioni di seno (la funzione "sinc(x)" è una abbreviazione di sin(x)/x);
- 1.638.400 sottrazioni.

Le operazioni di estrazione di seno e operazione di divisione sono comunque le più onerose a livello di cicli di CPU; sommando tutte le precedenti otteniamo un totale di 9.830.400 operazioni complessive da compiere in (molto) meno di 100 millisecondi. Infatti, tolti i 10..20 millisecondi necessari per le restanti operazioni (calcolo FFT, disegno a video eccetera), e considerando ragionevole riservarsi un margine di almeno 10 mS per fluttuazioni nella disponibilità delle risorse (es. CPU assegnata ad altri processi), il tempo rimanente si aggira intorno ai 60 millisecondi. Le prove sperimentali hanno fornito tempi che si aggiravano proprio intorno a tale valore, cosa che dipende ovviamente fortemente dalla macchina usata e dallo stato di carico. Il risultato è che il sistema diventava troppo critico per poter essere utilizzato praticamente, anche con moderni microprocessori a doppio "core" ed elevate frequenze di clock. A questo si aggiunga il fatto che dovendo usare il generatore di funzioni o peggio la funzione doppia traccia, si sarebbe avuto un ulteriore consumo di risorse "nel ciclo" cosa che rendeva ancor più improponibile l'uso di un simile algoritmo in maniera diretta e per ogni buffer.

La possibile soluzione arriva dalle seguenti considerazioni. Intanto, fissata una determinata frequenza di campionamento il termine *Tc* diviene una costante, e lo stesso dicasi per i termini *n*Tc*, dato che ad ogni buffer il contatore del ciclo della ricostruzione (n) riparte da zero. Lo stesso dicasi per la variabile t, che rappresenta (in Visual Analyser) il tempo "relativo" da inizio a fine schermata, ossia ad ogni schermata riparte da zero.

L'idea è quindi questa: "pre-calcolare" le quantità che non variano ad ogni nuovo buffer e per quel dato valore di *Tc*, ossia di frequenza di campionamento. In particolare la quantità:

$$\sin c(\pi \frac{t - nT_c}{T_c}) \tag{5.7}$$

viene pre-calcolata tramite un thread a bassa priorità (una tantum) e posta nella matrice *sinc[t][k]* il cui significato delle due variabili indipendenti è il seguente:

- t = è il tempo t di cui si vuole sapere il valore del punto del segnale analogico ricostruito;
- k = è il parametro su cui iterare per applicare la 5.6 (che può arrivare ad essere compresa tra zero e tutti i punti del buffer).

Pre-calcolando queste quantità, le operazioni da compiere in tempo reale si limitano a 4096*400 operazioni per il disegno di una schermata, ossia *9.830.400 – 1.638.400 = 8.192.000* in meno ad ogni giro!

Prove sperimentali sulla (solita) macchina (composta da un Core Duo E9600, 40960 Hz di frequenza di campionamento e 4096 punti di buffer) hanno rilevato un tempo aggiuntivo di calcolo nel ciclo principale compreso tra 5..10 millisecondi, rendendo l'algoritmo praticamente utilizzabile.

Valgono le seguenti considerazioni aggiuntive. Intanto le variazioni delle dimensioni del buffer possono influire significativamente sulle considerazioni sinora effettuate, in special modo laddove le dimensioni del buffer siano comparabili con le dimensioni della finestra oscilloscopio. Inoltre, una finestra grafica di dimensioni maggiori (per esempio di 1024 punti) aumenta significativamente il tempo di calcolo. Ancora, l'uso della funzione trigger può portare alla necessità di effettuare l'estensione del ciclo di calcolo della conversione digitale analogico a parte dei punti del buffer precedente aumentando ulteriormente il carico computazionale. Per tali ragioni la conversione interna può essere disabilitata "manualmente" e persino quando abilitata, è automaticamente disinserita da Visual Analyser in talune particolari condizioni; per esempio, se si rilevano configurazioni tali per cui la sua applicazione non comporta miglioramenti apprezzabili nel disegno del segnale a video.

Un simile risultato, rarissimo in questa categoria di programmi, è stato ottenuto tramite l'aggiunta di un thread dedicato e l'ottimizzazione (ed approssimazione) delle formule necessarie per il calcolo del segnale originale (ossia il teorema del campionamento), mantenendo un accettabile grado di qualità del segnale e di esecuzione in tempo reale. In appositi capitoli e paragrafi verranno descritte le

caratteristiche matematiche dell'algoritmo. Per consentire la massima flessibilità d'uso è possibile abilitare manualmente la funzione "conversione D/A" direttamente dal pannello principale posto alla destra dell'oscilloscopio, consentendo, laddove non necessario (per esempio nel caso dei 100 Hz campionati a 10 kHz di figura 6) di risparmiare una notevole dose di risorse di calcolo da lasciare così disponibili a beneficio di altre funzioni eventualmente attivate. Invero, come avremo modo di vedere nei paragrafi successivi, anche quando la funzione D/A è attivata Visual Analyser cerca di utilizzarla nella maniera più ridotta possibile, sulla base di un apposito algoritmo di controllo; esso determina che il risultato a video non cambierebbe anche se usata e dunque provvede alla sua temporanea disabilitazione. In taluni casi tuttavia, se non si vuole correre il rischio di avere perdita di dati, è opportuno disattivare del tutto (manualmente) questa opzione.

5.9 Cattura e data log

Il processo di "cattura" dati e data log è particolarmente curato in VA, e risulta una caratteristica importante e particolarmente potente e tipica di uno strumento virtuale, che è infatti di più difficile implementazione in uno strumento reale. Prima di addentrarci in spiegazioni relativamente approfondite, facciamo notare come abbiamo voluto mantenere una differenziazione dei due termini "cattura" e "data log", semplicemente intendendo rispettivamente per il primo un processo di acquisizione dati completo di grafico, colori, nomi degli assi e delle grandezze in gioco (in un file con estensione apposita, e visualizzabile con "viewer" apposito con possibilità di analisi, stampa e ispezione), mentre per il secondo un semplice salvataggio "grezzo" dei valori corrispondenti all'asse X e Y eventualmente in un file testo. NOTA: è possibile copiare i "dati grezzi" nella clipboard e successivamente (per esempio) esportarli in programmi esterni come Word o Excel (o equivalenti, es. Openoffice). Essendo la seconda modalità (dati grezzi) una semplificazione della prima, faremo riferimento solo alla prima (cattura). VA utilizza dei thread specifici per l'acquisizione dati, esplicitamente sincronizzati al thread

principale tramite primitive di sincronizzazione canoniche (mutex, semafori, lock, event, etc.); le varie modalità di funzionamento della cattura dipendono dalla grandezza che si intende acquisire, anche se tutte basate sul medesimo principio di funzionamento. Ad esempio la cattura dei dati "nel dominio del tempo", ossia i dati per esempio visualizzati dall'oscilloscopio, sono affiancati contestualmente ad una finestra che ne consente il calcolo contemporaneo della relativa trasformata di Fourier, e dunque dello spettro, cosa per esempio assente in altri tipi di cattura, come ad esempio la cattura delle misure di impedenza. Con VA è possibile catturare le seguenti grandezze, identificate dagli strumenti che le misurano:

- oscilloscopio;
- analizzatore di spettro, ampiezza e fase;
- THD, THD+n con sweep in frequenza;
- valori misurati da strumento ZRLC;
- incertezza statistica (vedi par. 5.9).

Prima di descrivere separatamente i punti di cui sopra, diamo alcuni ulteriori importanti dettagli. In generale ogni finestra di cattura dei dati consente di catturare i dati ed effettuarne un salvataggio, che può avvenire sostanzialmente nei modi già accennati, ossia salvataggio completo, dati testo e/o clipboard. In generale, la finestra di cattura dei dati presenta una interfaccia comune per ogni tipologia di cattura, costituita dal grafico in se e opzioni dipendenti dal tipo di dati catturati. In generale il salvataggio dei dati avviene, nella sua forma più completa, nel formato ".tee" ossia il formato stabilito dagli oggetti usati per la visualizzazione dei dati (TeeChart) ma che ovviamente dipende dal tipo di cattura. In altre parole, i dati relativi alla cattura di una impedenza avranno un formato differente da quello relativo ad un oscilloscopio od analizzatore di spettro. Per esempio nella tipo di grandezza rappresentata sull'asse Y oppure dal numero di grafici e modalità di rappresentazione (a linee, barre, etc.). Questo pone un problema nel processo inverso, ossia nella lettura dei dati "postuma" ossia se si vuole aprire un file di cattura precedentemente salvato. E' infatti possibile utilizzare la stessa finestra appena aperta automaticamente dopo la cattura oppure utilizzare un "viewer" formato dalle stesse finestre di cattura ma richiamabili

"offline". Il menù è presente direttamente nella finestra principale, nel tab "more" ed è per comodità riportato nella figura 41.

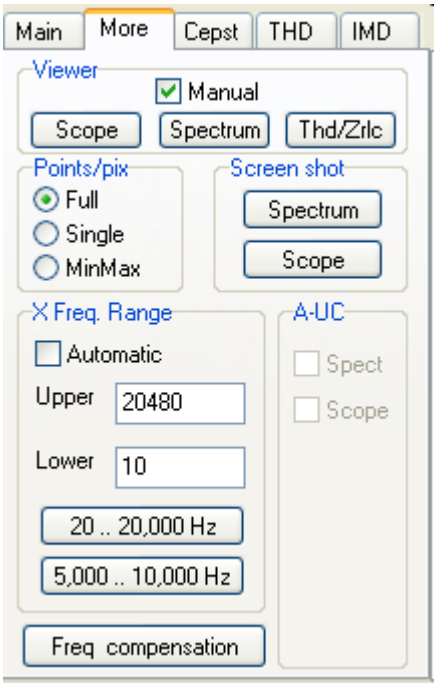


Figura 41: il menù a bottoni del visualizzatore

In esso si può dunque esplicitamente notare come sia demandato all'utilizzatore con quale visualizzatore effettuare la lettura del file dati.

Questo comporta che in caso di errata manovra (per esempio aprire un file di dati oscilloscopio con il visualizzatore dell'impedenza) si avrà una errata visualizzazione dei dati (assi, tipi di grafici, etc.).

Per evitare problemi di questo tipo si è aggiunta una ulteriore comoda funzione che consente, previa de-selezione della checkbox "manual" di richiamare una procedura che stabilisce il "tipo" di file dati e richiama il visualizzatore preposto. Questo è ottenuto aggiungendo nel file dati informazioni apposite. In figura 42 è possibile vedere il menù "automatico".

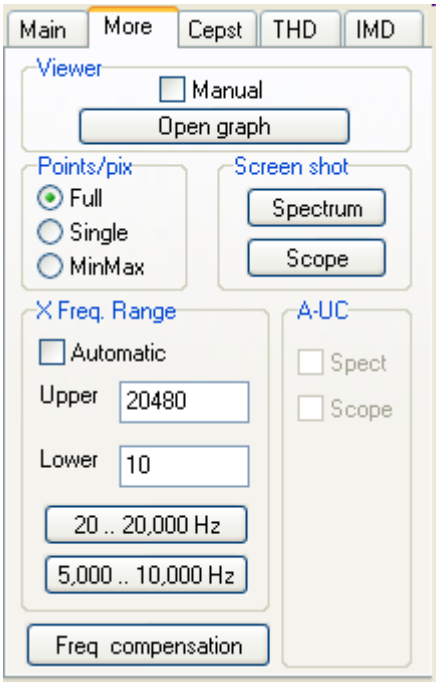


Figura 42: menù a scelta automatica

5.9.1 Cattura dati oscilloscopio

Nel tab "main" della finestra principale del programma è presente il bottone "capture scope" che consente di iniziare il processo di cattura dei dati visualizzati nell'oscilloscopio. Se il programma non è in "on", ossia non sta girando il task di cattura dati, esso viene messo automaticamente in esecuzione e viene dato inizio al processo di cattura. Questo processo è apparentemente semplice, in quanto si tratta di memorizzare i campioni provenienti dalla scheda di acquisizione. Invero il processo è ben più complesso per una serie di ragioni. Esse appariranno chiare nel corso del presente paragrafo. Appena dato il via al processo di cattura esso viene monitorato da una barra di scorrimento che si sostituisce temporaneamente al bottone stesso, e la cui durata è impostata nella finestra di setting, tab "capture" (descritta a breve). In caso di processi di cattura lunghi (per esempio impostati erroneamente oppure di cui non si abbia più necessita) è possibile interrompere l'acquisizione in qualsiasi momento semplicemente cliccando sulla stessa barra di scorrimento; la cattura viene interrotta e i dati memorizzati sino all'interruzione visualizzati nella finestra, la

stessa che apparirebbe alla fine del processo completo di cattura. Essa si presenta come in fig. 43 .

Figura 43: la finestra di cattura oscilloscopio

Essa presenta un'area in cui è presente il grafico dei dati acquisiti, con la possibilità di essere "zoomato" e "spostato" con continuità tramite il mouse ed uso coordinato dei tasti destro e sinistro. In particolare per effettuare lo zoom di una porzione di grafico si deve posizionare il cursore del mouse sull'angolo in alto a sinistra della zona rettangolare che si vuole "amplificare", cliccare con il tasto sinistro e, mantenendolo premuto, posizionare il cursore sull'angolo destro in basso della zona desiderata; ed infine rilasciare il pulsante. Per effettuare l'operazione inversa è sufficiente effettuare un'azione inversa alla precedente ma tracciando un rettangolo di dimensioni qualsiasi. Per far scorrere il grafico in ogni direzione è sufficiente cliccare con il tasto desto e trascinare il mouse nella direzione voluta.

Tutte queste operazioni sono effettuabili anche tramite delle scrollbar poste ai bordi del grafico e/o tramite i pulsanti posti in basso a sinistra nella finestra. Nella medesima zona è possibile visualizzare le "coordinate" del cursore nelle unità di misura della grandezza catturata.

La cattura dell'oscilloscopio presenta una serie di caratteristiche particolarmente interessanti; tra le quali la possibilità di effettuare una conversione digitale-analogica simile a quella descritta nel par. 5.8 ma dedicata ai dati selezionati nella finestra di cattura, e svincolata dal ciclo complessivo real-time dello strumento oscilloscopio, e alla bisogna realizzata con un thread a bassa priorità che agisce secondo il solito principio di "fai le cose solo quando hai tempo disponibile". In tal senso è dunque possibile selezionare una porzione dei dati acquisiti (si considerano selezionati quelli effettivamente visibili nella finestra e dunque evidenziati tramite zoom e scorrimento) ed iniziare il processo di conversione cliccando sul menù a tendina principale voce "D/A". Il processo inizia in background ed è monitorato da una barra di scorrimento; cliccando sulla quale è possibile interrompere e provocare la visualizzazione dei soli dati convertiti sino a quel momento. Dopo la conversione D/A i dati visualizzati sono quelli convertiti ed è possibile tornare indietro cliccando di nuovo sul medesimo menù a tendina.

Un'altra caratteristica interessante è la possibilità di acquisire i dati in funzione di soglie pre-impostate ed effettuare una pre-acquisizione relativamente all'istante effettivo di inizio acquisizione dati.

Quest'ultima caratteristica è particolarmente interessante se usata in abbinamento all'uso delle soglie, sebbene utile in generale pur con un processo di acquisizione iniziato manualmente (e con le imprecisioni del caso, dipendenti dallo stato di "occupazione" delle risorse elaborative).

La descrizione di queste due ultime funzioni è facilitata riportando la schermata relativa alla finestra di setup della funzione "capture" (fig. 44).

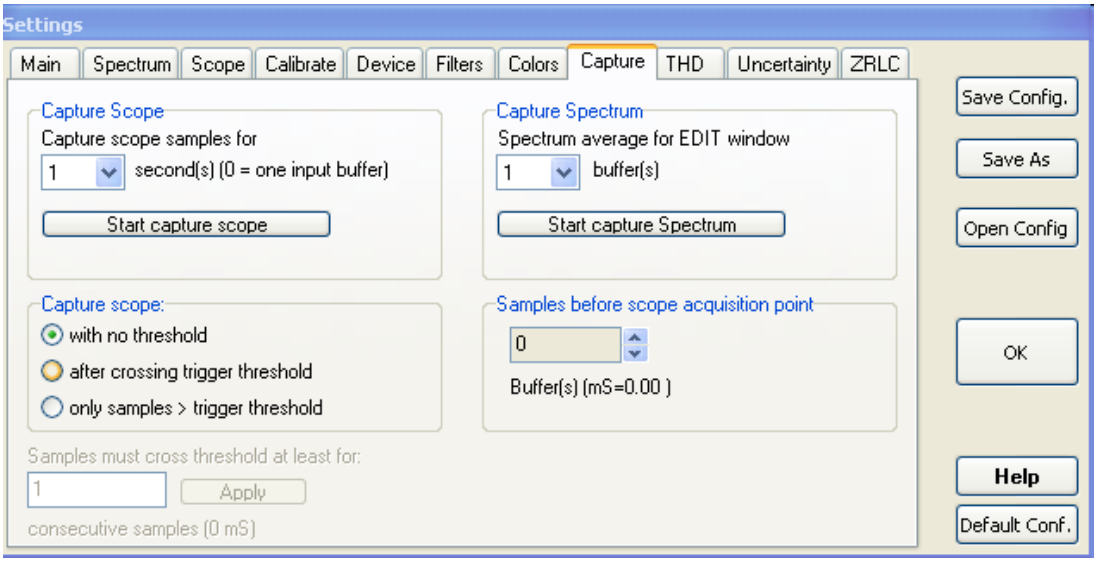


Figura 44: setup della funzione capture (scope & spectrum)

In essa, in relazione alla funzione "capture scope", è possibile impostare:

- Il tempo di acquisizione, espresso in secondi, con la convenzione che ZERO secondi hanno il significato della minima acquisizione possibile, ossia UN solo buffer di acquisizione dati, ed il cui tempo corrispondente è dipendente dalla dimensione del buffer stesso e dalla frequenza di campionamento (calcolato ed indicato alle destra della finestra).
- La soglia che determina l'inizio effettivo del processo di acquisizione dal momento della pressione del tasto "capture scope". Essa è in particolare prevista in due modalità, oltre a quella "no threshold" che rappresenta il default (e dunque acquisizione immediata subito dopo la pressione del tasto): la prima è quella "*after crossing trigger threshold*", ossia il processo di acquisizione inizia effettivamente solo dopo che il livello del segnale SUPERA la soglia che si imposta con il controllo di trigger dell'oscilloscopio. In tal senso il controllo di livello di trigger ha un DOPPIO significato: livello di trigger, se usato con trigger "on" oppure livello di intervento del processo di acquisizione quando in cattura dati oscilloscopio. Da notare che il trigger in quanto tale può continuare ad essere usato in concomitanza dell'acquisizione dei campioni. Selezionando invece l'opzione "*Only samples > trigger threshold*" si effettua la cattura se il livello del segnale supera la soglia, e si catturano SOLO i campioni che effettivamente la

superano. In quest'ultimo caso e nel precedente è possibile avvalersi di una ulteriore opzione ("*samples must cross threshold at least for*") che consente di estendere il concetto di soglia ad un valore che sia uguale o superi il valore di soglia almeno per un numero "n" di campioni (che di default è pari a 1).

- Il tempo di pre-acquisizione; il programma da la possibilità di acquisire campioni per un tempo PRECEDENTE l'evento che causa l'effettiva acquisizione dei campioni, definibile a piacere entro un massimo stabilito di 500 mS (che in future versioni potrebbe essere aumentato anche considerevolmente). Nel caso di selezione per esso di un valore maggiore di zero, all'interno della finestra di visualizzazione dati acquisiti sarà presente una linea verticale tratteggiata ad indicare il punto esatto di effettiva acquisizione dei campioni, intendendo in tal modo che i campioni precedenti tale confine siano quelli presenti per "tot" mS prima dell'acquisizione. In altre parole, se si imposta un tempo di pre-acquisizione di 100 mS ed una soglia di intervento di 2 volt "semplice", il segnale sarà acquisito solo dopo che la soglia supera almeno per un campione il valore di due volt e per 100mS prima di questo evento.

Una ulteriore ed importantissima funzione, che in sinergia con quelle appena descritte rende questa funzione di acquisizione una delle più complete e potenti di tutto il programma, è la possibilità di calcolare lo spettro del segnale relativamente alle porzioni del segnale visualizzato, come per la conversione D/A. In pratica, dal menù a tendina, opzione View, è possibile selezionare l'opzione "Spectrum" che fa comparire un ulteriore grafico il cui contenuto è esattamente lo spettro del segnale temporale della finestra di cattura dei campioni. In particolare per esso è possibile applicare una finestra di smoothing diversa da quella applicata in fase di acquisizione dati in tempo reale ed eventualmente ridefinire la frequenza di campionamento. Una ulteriore e potente opzione è quella di attivare il calcolo automatico dello spettro: è sufficiente selezionare l'opzione "Auto" presente in basso al centro della finestra. In tal modo qualsiasi variazione alla finestra del segnale temporale (per esempio in conseguenza di uno zoom o scorrimento) sarà immediatamente riportata nella rappresentazione in frequenza del segnale. In tal modo sarà rapidamente possibile effettuare qualsiasi operazione di analisi del segnale selezionato. In ultimo si osservi

che nei menù a tendina è possibile impostare svariate operazioni “cosmetiche” ma abbastanza interessanti, sulle quali non si ritiene utile soffermarsi in questa sede; a titolo di esempio la possibilità di uso dei “marker”, ossia delle etichette che “marcano” il punto con i valori esatti che esso rappresenta, visualizzazione lineare/logaritmica, salvataggio, etc. opzioni peraltro presenti in tutte le finestre di visualizzazione dei dati acquisiti.

5.9.2 Cattura Analizzatore di spettro

Questa funzione consente di acquisire lo spettro in forma completa, ossia formato dal diagramma delle ampiezze e della fase. Per questa finestra valgono le stesse opzioni generali descritte nel paragrafo precedente sulle modalità di zoom e scorrimento, e per il salvataggio nei vari formati (testo, completo e clipboard).

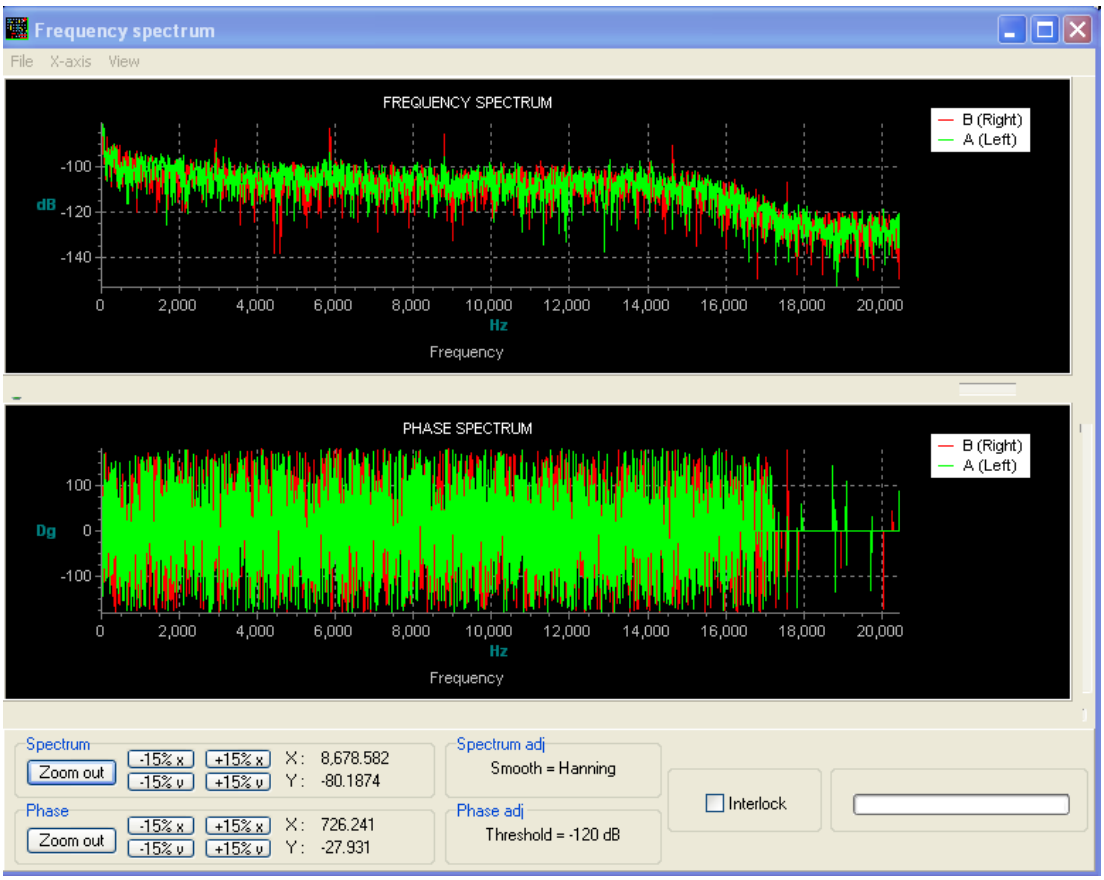


Figura 45: La finestra di cattura spettro

Lo spettro di fase e di ampiezza possono essere sincronizzati o meno, a seconda delle esigenze; lo spettro acquisito è esattamente quello visualizzato dalla finestra real-time, e come tale mantiene i settaggi per essa effettuati; pertanto non è possibile applicare una finestratura arbitraria in quanto già applicata dal programma principale. Per facilità di consultazione essa è comunque riportata in apposito riquadro in basso al centro.

5.9.3 Cattura THD

Nell'uso della funzione calcolo della distorsione armonica, sia nella versione standard che nella versione che tiene conto del rumore (THD+noise), è possibile aggiungere i valori delle misure ad un grafico che in ascisse ha il valore della frequenza. E dunque effettuare una serie di misure ripetute a frequenza variabile.

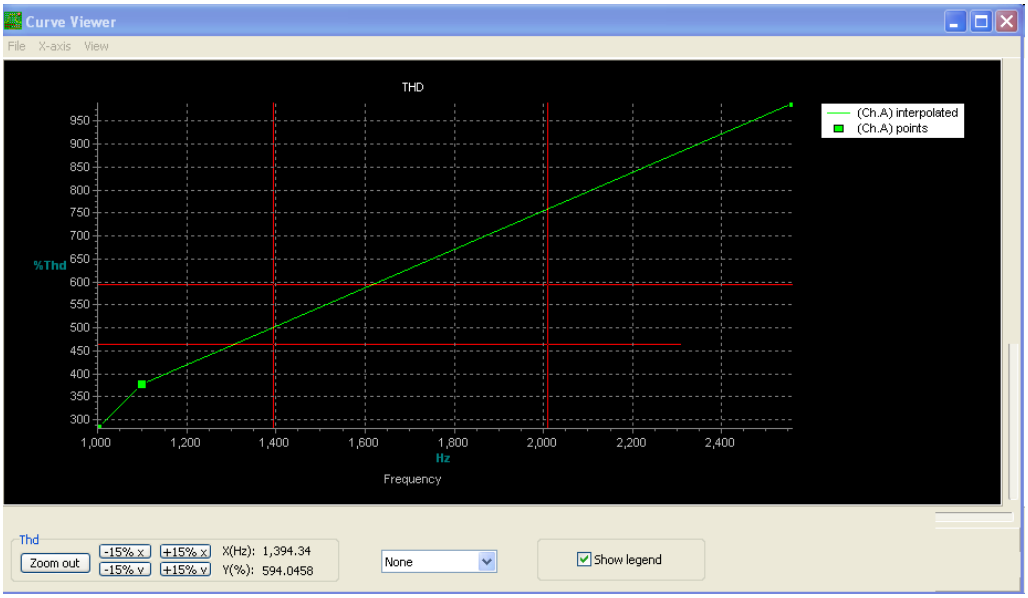


Figura 46: la finestra di cattura THD

Data la natura della misura è altresì evidente che aumentando la frequenza di misura diminuisce proporzionalmente l'accuratezza di misura; essa infatti si basa sul calcolo dei contributi delle armoniche multiple della fondamentale (che costituiscono l'essenza stessa del fenomeno distorsione) la cui presenza è limitata dalla banda passante della scheda di acquisizione. A titolo d'esempio, il calcolo della distorsione

armonica di una forma d'onda a 10 kHz usando una scheda di acquisizione con banda passante limitata a 20 kHz avrebbe come conseguenza che il calcolo della distorsione verrebbe effettuato su una sola armonica, mentre per una frequenza (p. es.) di 1000 Hz su ben 19 armoniche. Una delle caratteristiche interessanti offerte da questa finestra di cattura è la possibilità di interpolare i dati con svariate funzioni canoniche, come per esempio la spline, B-spline e lineare.

5.9.4 Cattura ZRLC

Valgono per essa le considerazioni generali di 5.8.1 relativamente alle caratteristiche di visualizzazione in zoom e scorrimento, oltre a salvataggio e opzioni di visualizzazione. La funzione di cattura ZRLC consente di catturare le misure effettuate con lo strumento ZRLC (cfr. cap. VIII), e quindi capacità, induttanza, resistenza od impedenza in genere (con calcolo di parte reale e immaginaria). L'acquisizione può avvenire in due modalità: con spazzolamento nel "dominio del tempo" ossia senza cambiare frequenza del segnale di test, e dunque semplicemente effettuando misure ripetute in numero definibile a piacere dall'utente; oppure variando progressivamente la frequenza del segnale di test con un passo predefinito e con l'ulteriore possibilità di effettuare misure ripetute nel dominio del tempo per ogni frequenza prevista nello spazzolamento.

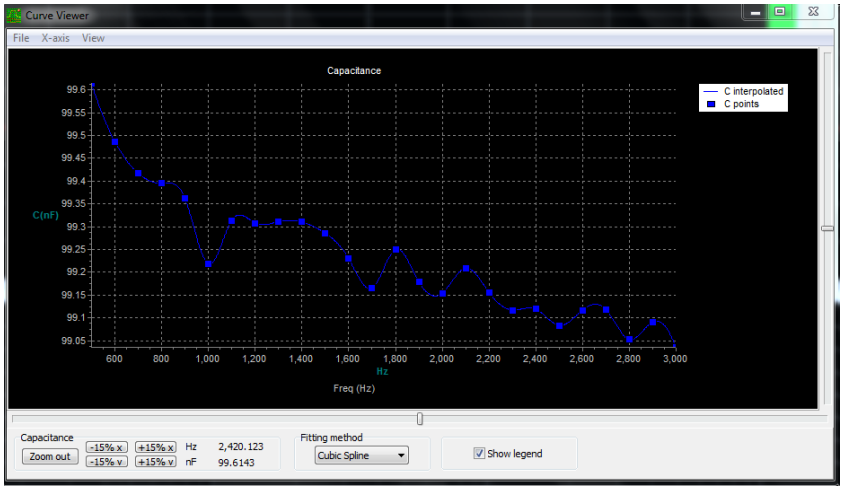


Figura 47: la finestra di cattura impedenzimetro nel dominio della frequenza

Quest'ultima possibilità è particolarmente interessante, perché consente di avere un grafico che mostra come varia l'impedenza di un bipolo al variare della frequenza ed impostando al contempo, per ogni singola frequenza di misurazione, un ben determinato grado di accuratezza della misura.

Anche in questa finestra è stata aggiunta la possibilità di interpolare il grafico con una curva continua secondo tre differenti algoritmi di interpolazione (spline, B-spline e lineare).

5.9.5 Cattura Incertezza statistica

La cattura dell'incertezza statistica riguarda solo il salvataggio dei dati relativi all'istogramma nel formato standard dalla finestra di calcolo incertezza di tipo "A" per la cui descrizione si rimanda al paragrafo successivo.

5.10 Incertezza

L'incertezza di misura è un parametro generalmente calcolato "ad hoc" per ogni strumento; esiste tuttavia una procedura generale, disponibile per tutti gli strumenti, che viene realizzata tramite un thread; essa è relativa al calcolo dell'incertezza statistica, ossia di tipo "A" (cfr. cap. II). Per ogni strumento implementato è possibile attivare un thread, associato ad una finestra, che calcola i parametri principali dell'incertezza statistica e l'istogramma delle frequenze, utile per valutare la distribuzione delle misure ottenute (che spesso, come è facile evincere dal teorema del limite centrale, è di tipo gaussiano).

La peculiarità di questo thread è la possibilità di calcolare in tempo reale l'incertezza statistica associata alla misura, effettuando il calcolo dei parametri su un gran numero di misure ripetute ed ottenendo pertanto una valida stima dello scarto tipo della media; è inoltre possibile attivare più finestre contemporanee (una per ogni strumento implementato) effettuare salvataggi e manipolazione dei grafici e dati

ottenuti; è anche possibile copiare i dati relativi alle misurazioni in clipboard ed eventualmente appoggiarsi a tool esterni come Excel o Word (o equivalenti).

In particolare è possibile calcolare le seguenti grandezze:

- media;
- deviazione Standard (e varianza);
- scarto tipo della media;
- istogramma delle frequenze.

La finestra generale è visibile nella seguente figura (48).

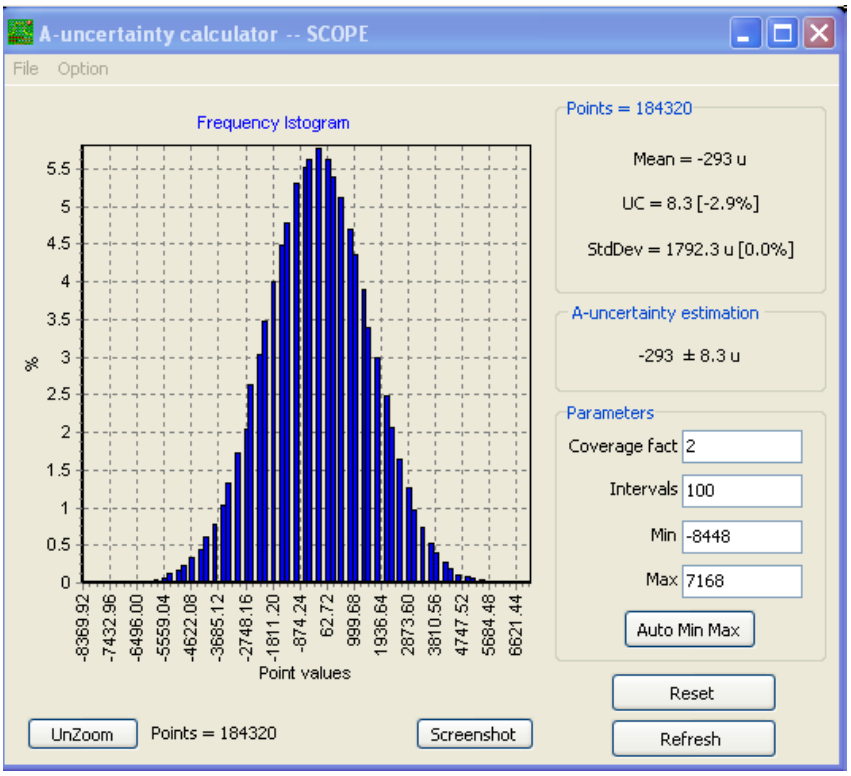


Figura 48: Finestra per il calcolo dell'incertezza di tipo A

Il meccanismo generale di esecuzione è il seguente. E' stata definita una classe che definisce una serie di metodi che consentono di memorizzare i risultati della misura in vettori interni; essi possono essere riempiti sino ad un massimo di valori che dipende essenzialmente dalla memoria disponibile della macchina usata, con delle soglie "prudenziali" comunque definibili dall'utente ed in ogni caso che non possono

eccedere il limite di memoria che consente la stabilità del sistema (valutata tramite apposite procedure).

Tutto il resto è implementato da una nutrita serie di metodi, che possono essere invocati in qualsiasi momento; il punto di forza di questo oggetto è che consente di "accumulare" velocemente i risultati delle misure (ossia con consumo di risorse praticamente irrisorio) e poi di effettuare tutti i calcoli necessari solo quando possibile, invocando i metodi predefiniti. Inoltre, come vedremo, alcuni calcoli sono stati implementati in maniera da minimizzare i problemi tipici degli algoritmi implementati su calcolatori con registri a dimensione finita (es. cancellazione, algoritmi mal condizionati etc.).

Le routine implementate (metodi) consentono di effettuare le seguenti operazioni (verranno descritte solo quattro tra le più significative):

AddSample: consente di aggiungere il valore di una misura ai vettori interni;

Compute: calcola media, varianza, deviazione standard e scarto tipo; per fare questa operazione, che a breve descriveremo in dettaglio, usa delle variabili interne di "accumulo" che vengono mantenute per tutta la durata del ciclo di misure; essa può essere usata anche SENZA memorizzare i campioni con "AddSample", ma in tal caso non si potrà disegnare l'istogramma;

AddPointToInterval: aggiunge il risultato di una misura, internamente memorizzata da AddSample, all'istogramma senza visualizzarlo;

UpdateIstogram: disegna effettivamente l'istogramma.

Il concetto è chiaro; con AddSample si aggiungono i punti ai vettori interni, e tutte le altre routine (sono complessivamente venti) possono essere usate, opportunamente gestite da un thread, per fare i calcoli ed il disegno dell'istogramma "quando c'è tempo"; il programma deve infatti effettuare molte operazioni in tempo reale e talune non interrompibili (per esempio l'acquisizione dei campioni) pertanto l'unica operazione fondamentale è quella di memorizzare le misure, le altre possono essere fatte in momenti di minore criticità. Dunque se necessario, per ogni strumento, è possibile allocare un thread che a sua volta istanzia un oggetto per il calcolo della

incertezza di tipo A e gestire in maniera "intelligente" le operazioni di memorizzazione, calcolo e visualizzazione dei risultati. Questa, che di base è la filosofia del programma tutto, consente una esecuzione praticamente parallela di tutte le finestre (e dunque gli strumenti simulati) del programma.

Descriviamo ora i dettagli dell'implementazione della routine "Compute" che effettua il calcolo delle grandezze statistiche, ottimizzata al fine di garantire velocità e algoritmi "ben condizionati". In particolare, data l'espressione della media:

$$\mu = \frac{1}{N}\sum_{i=0}^{N-1} x_i \tag{5.8}$$

e della varianza:

$$\sigma^2 = \frac{1}{N-1}\sum_{i=0}^{N-1}(x_i - \mu)^2 \tag{5.9}$$

dove:

N = numero dei campioni del segnale;

x_i = campione i-esimo

μ = media

σ^2 = varianza

E' facile convincersi della semplicità di calcolo di queste grandezze, e da esse derivate; pur tuttavia l'uso diretto delle 5.8 e 5.9 porta ad un uso particolarmente inefficiente delle risorse elaborative. Questo per due motivi, estremamente semplici: primo, nella 5.9 si può verificare che la differenza tra parentesi coinvolga numeri tra loro troppo simili, e dunque dare origine ad una forma di rounding-error (cfr. cap. VI) detta "cancellazione". Inoltre, e particolarmente grave, ogni qual volta un campione viene aggiunto è necessario ricalcolare le 5.8 e le 5.9 ripetendo ex-novo le sommatorie.

Una possibile soluzione al problema è quella di manipolare opportunamente le 5.8 e 5.9 per riscrivere la 5.9 in questa maniera:

$$\sigma^2 = \frac{1}{N-1}\left[\sum_{i=0}^{N-1} x_i^2 - \frac{1}{N}\left(\sum_{i=0}^{N-1} x_i\right)^2\right] \tag{5.10}$$

In questa relazione, perfettamente equivalente alla 5.9, si possono evidenziare due termini; il primo:

$$\sum_{i=0}^{N-1} x_i^2 \tag{5.11}$$

ed il secondo:

$$\frac{1}{N}\left(\sum_{i=0}^{N-1} x_i\right)^2 \tag{5.12}$$

Il termine 5.11 è la *somma dei quadrati* che può essere memorizzato ad ogni giro, ossia ogni qual volta arriva un campione. Esso è dunque rappresentato da una sola variabile che va aggiornata ogni volta che viene invocata la "AddSample"; la 5.12 è il *quadrato della somma* ossia anch'essa può essere rappresentata da una sola variabile (somma dei campioni), infatti basta tenere conto solo della somma, ed all'atto del calcolo della varianza elevare al quadrato o nel caso della media dividere per N. In ultimo, si deve tenere conto anche del numero N di campioni acquisiti, ossia un'altra semplice variabile. Il gioco è dunque quello di tenere conto essenzialmente di sole tre quantità, invece di effettuare il calcoli di 5.8 e 5.9 ad ogni giro. Con queste tre quantità "accumulate" è dunque possibile calcolare in maniera efficiente e veloce la 5.8, ossia la media, e la 5.9 (varianza) ed eventualmente, tramite l'estrazione di radice quadrata, la deviazione standard ed altre grandezze correlate.

Il calcolo e disegno dell'istogramma richiede invece la memorizzazione di tutti i campioni separatamente, in un vettore, cosa comunque anch'essa semplice e poco

onerosa computazionalmente; il disegno dell'istogramma, e quindi la suddivisione dei campioni negli intervalli stabiliti, il calcolo delle frequenze e infine il disegno effettivo del grafico (per esempio a barre) è un'attività relativamente "resource-consuming" che giustifica l'implementazione di un thread per l'esecuzione di essa in condizioni di minima interferenza con le routine principali (e critiche) degli strumenti di misura.

Il supplemento della GUM [40] indica che la relazione 5.10 potrebbe anch'essa essere affetta da una forma piuttosto severa di errore di cancellazione, e dunque consiglia caldamente l'uso della relazione standard 5.9. Nella opzioni del programma è stata pertanto prevista la possibilità di selezionare il metodo voluto per il calcolo della varianza; in particolare è dunque possibile selezionare l'uso di 5.9 o della 5.10. L'uso pratico con la maggior parte degli strumenti ha comunque evidenziato che in media i risultati sono stati molto simili, e comunque di determinare quelle sessioni di misura in cui è possibile usare il metodo "veloce" senza particolari difficoltà.

Capitolo 6

Rounding error

6.1 Introduzione

Come ampiamente descritto in altre sezioni, esistono numerose cause concomitanti che contribuiscono al calcolo complessivo di una quantità che abbiamo a più riprese definito “incertezza di misura”. Vogliamo in questo capitolo porre l’enfasi su di una sola causa, e darne parimenti una valutazione numerica per stimare quanto essa contribuisca alla determinazione dell’incertezza composta, ossia all’incertezza “complessiva” calcolata tenendo presente “tutte” le sorgenti di incertezza, compatibilmente con la conoscenza del fenomeno misurato, corretta definizione del misurando e non ultimo della bontà degli strumenti utilizzati.

Gli strumenti virtuali, per loro stessa definizione sostituiscono componenti hardware con del software, eseguendo un gran numero di algoritmi mutuati dal mondo dell’Elaborazione Numerica dei segnali e dal Calcolo Numerico, più altri eventualmente definiti ad hoc per lo specifico problema.

L’esecuzione di algoritmi su elaboratori implica una approssimazione di base in relazione alla rappresentazione che in essi abbiamo dei numeri: questi sono infatti rappresentati con un numero finito di bit, ossia un numero reale, di per se infinito, è “approssimato” con un numero finito, ossia con un numero di cifre limitato dalla

capacità dei registri della macchina. Inoltre, ogni operazione effettuata tra due o più operandi comporta risultati che per loro natura effettuano ulteriori arrotondamenti e troncamenti. Si pensi, per chiarire le idee, ad una moltiplicazione tra due semplici numeri reali (già troncati) a 32 bit, rappresentati in qualsivoglia maniera (in virgola fissa o mobile, per esempio); essi darebbero luogo, generalmente, ad un numero reale di dimensioni nuovamente infinite e comunque almeno a 64 bit e dunque (dovendo esso essere nuovamente contenuto in 32 bit) introducendo ancora un "errore"(approssimazione) nella rappresentazione che di esso abbiamo in memoria.

Questo fenomeno, noto come "errore di arrotondamento" o "rounding error" è di per se una ulteriore causa di incertezza, in particolar modo per gli strumenti virtuali che fanno un uso intensivo di unità elaborative (DSP, Microcontrollori, Microprocessori).

Vogliamo qui dimostrare, tramite analisi rigorose "white box" confermate da test sperimentali (montecarlo), che l'incertezza introdotta dall'errore di arrotondamento può essere resa trascurabile rispetto all'entità delle incertezze introdotte dall'hardware di acquisizione; in particolare, faremo un confronto tra due tipiche cause di incertezza introdotte dall'hardware (jitter e quantizzazione, cfr. III) e il rounding error. I risultati di questo capitolo sono basati sul lavoro citato in [26] e [28], e presentati anche in [3]; applicheremo i concetti di propagazione dell'errore (cfr. II) all'algoritmo più complesso utilizzato in VA, ossia la trasformata di Fourier, calcolata tramite l'FFT (Fast Fourier Transform). In prima battuta daremo le relazioni che descrivono il modello matematico delle incertezze considerate (eq. 6.1 e 6.2); successivamente descriveremo il modello matematico utilizzato per implementare lo strumento (trasformata di Fourier, eq. 6.3) e in ultimo la propagazione delle incertezze nello stesso (eq 6.4). Per finire effettueremo una valutazione numerica tramite una procedura automatica di analisi montecarlo integrata nel programma, nella quale terremo conto delle scelte effettuate per minimizzare l'errore di arrotondamento ed effettueremo un confronto dei risultati ottenuti (par. 6.5).

6.2 Modello matematico del “rounding error”

Il fenomeno sinora descritto qualitativamente, può esprimersi nella seguente maniera (intendendo con il simbolo “fl” la rappresentazione nei registri macchina del numero floating point considerato; laddove esso simbolo risulta assente è da intendersi il numero “vero”):

$$fl(x \cdot y) = (x \cdot y) \cdot (1 + \gamma) \tag{6.1}$$

$$fl(x + y) = (x + y) \cdot (1 + \eta) \tag{6.2}$$

dove γ e η sono variabili casuali a media nulla, indipendenti da *x, y* e $(x \cdot y)$ o $(x + y)$ rispettivamente, e uniformemente distribuite nel range $[-2^b \,..\, 2^b]$ dove “b” è il numero di bit utilizzati dalla rappresentazione del numero reale in virgola mobile (floating point). Si dimostra che l’incertezza introdotta dalle operazioni di moltiplicazione e addizione, può essere approssimativamente posta pari a:

$$U_{fl\_m} = \sqrt{0.18 \cdot 2^{-2 \cdot b}} \tag{6.3}$$

$$U_{fl\_a} = \sqrt{p \cdot 0.18 \cdot 2^{-2 \cdot b}} \tag{6.4}$$

Dove la (6.3) rappresenta l’incertezza data da un operazione di moltiplicazione e la (6.4) di addizione. “b” ha il significato consueto di profondità di bit, mentre il fattore “p” è la probabilità che un’operazione di addizione dia origine ad un errore di arrotondamento; infatti una somma di due valori sufficientemente piccoli (e/o magari tra due interi) può non dare origine a errore di arrotondamento. Nei calcoli terremo presente un valore di p molto elevato, per considerare il caso peggiore.

6.3 Modello matematico del “Jitter” ed “errore di quantizzazione”

Prenderemo in considerazione due cause tipiche di incertezza, presenti in tutte quelle schede di acquisizione che fanno uso di convertitori analogico digitali (e viceversa), e dunque le sole utilizzabili con il programma sviluppato e descritto in questo lavoro.

Il processo di campionamento (cfr. III) da origine a molteplici cause di incertezza come ampiamente descritto in IV, tra le quali il cosiddetto "errore di Jitter" e l'errore di quantizzazione.

Il primo, ricordiamo brevemente, è dovuto a fluttuazioni casuali dell'istante di campionamento, per esempio dovuto a instabilità del segnale di clock utilizzato dal convertitore, mentre il secondo è consequenziale alla limitatezza (cardinalità) dell'insieme dei numeri usati per rappresentare delle quantità di per se continue, e quindi potenzialmente costituite da un numero infinito di possibili valori. Entrambi i fenomeni danno origine ad una causa di indeterminazione nel valore della grandezza associata ad ogni campione, che può essere modellato con una apposita distribuzione di probabilità; ossia considerando che il contributo apportato al valore di ogni campione è una variabile casuale caratterizzata da una distribuzione di probabilità e media nulla.

L'incertezza introdotta a causa dell'errore di quantizzazione può essere facilmente tenuta di conto tramite la relazione seguente:

$$U_q = \frac{V_R \cdot 2^{-b}}{\sqrt{12}} \tag{6.5}$$

Dove "b" è il numero di bit utilizzati nel convertitore analogico-digitale (16 nel nostro caso, da non confondere con il numero di bit utilizzati nei numeri in virgola mobile interni al calcolatore) e V_R è la tensione massima d'ingresso del convertitore stesso. Questa incertezza introdotta su ogni campione può dunque essere modellata come un rumore casuale a distribuzione uniforme con media nulla. Il valore è immediatamente desumibile dalle caratteristiche dell'integrato utilizzato come convertitore A/D.

L'incertezza introdotta dal jitter, è ricavabile con considerazioni analoghe, ma in questo caso un minimo più complesse. Questa incertezza può infatti essere vista come una incertezza introdotta in primis sull'istante di campionamento t_n modellata con una distribuzione di probabilità uniforme nell'intervallo [$-J\tau$ .. + $-J\tau$] ed a media nulla, con varianza $(J\tau)^2/3$. Questa incertezza sull'istante di campionamento si "propaga" al valore dell'ennesimo campione acquisito *x[n]*, nel senso che anch'esso

può dunque essere visto come una variabile aleatoria. In particolare esso può essere visto come la somma di un valore ideale ed una variabile aleatoria α_n a valor medio non nullo ed uniformemente distribuita; e dunque, il valore *x[n]* sarà affetto da un'incertezza pari alla deviazione standard della variabile α_n. Se approssimiamo *x(t)* nell'intervallo $[n\tau - J\tau,\ n\tau + J\tau]$ con una funzione lineare, il range di variazione può essere approssimato con l'intervallo $[-a_n J\tau,\ +a_n J\tau]$ la cui deviazione standard è dunque:

$$U_{jn} = \frac{a_n \cdot J_\tau}{\sqrt{3}} \tag{6.6}$$

dove "*a*" è la derivata prima della *x(t)* nel punto considerato. Non si faccia confusione con il valore di *x[n]*; "*a*" è la pendenza della retta del segnale originario, dato che ovviamente è legato a come questo varia, e andrebbe dunque considerato un valore che meglio approssima le caratteristiche del segnale stesso. Per semplicità consideriamo per esso un valore unitario. Vedremo poi cosa comporta la variazione di questo parametro nel valore dell'incertezza.

Le relazioni (6.5) e (6.6), oltre a (6.3) e (6.4) verranno utilizzate, insieme alla relazione che descrive il nostro processo di misurazione (vedi paragrafo successivo), per calcolare l'incertezza complessiva data dalle tre cause descritte, di cui evidenzieremo i singoli contributi e ne effettueremo debita comparazione.

6.4 Modello matematico della misura

L'algoritmo utilizzato nello strumento "analizzatore di spettro" è la DFT (Discrete Fourier Transform), che considereremo essere utilizzato sotto l'ipotesi di "campionamento sincrono", ossia quella particolare modalità poi utilizzata da altri strumenti implementati e che si avvalgono del calcolo dello spettro del segnale (es. ZRLC, cfr. cap. VIII).

La DFT di una sequenza { x(n) } formata da "N" punti è:

$$x(k) = \frac{1}{N} \sum_{n=0}^{N-1} x(n) e^{-Jk\beta_n} \quad k=0..N-1 \tag{6.7}$$

Dove $\beta_n = 2\pi n / N$ e assumiamo di essere nella condizione di "campionamento sincrono" (cfr. cap. VII). Ponendo:

$$R(k) = \sum_{n=0}^{N-1} x(n) cos\ (k\,\beta_n)$$

$$I(k) = \sum_{n=0}^{N-1} x(n) sin\ (k\,\beta_n) \tag{6.8}$$

Possiamo scrivere:

$$M(k) = \frac{1}{N}\sqrt{R^2(k) + I^2(k)}$$

$$\varphi(k) = arctg(-\tfrac{I(k)}{R(k)}) \tag{6.9}$$

Queste relazioni, che esprimono rispettivamente parte reale/immaginaria e modulo/fase del modello matematico della misura, verranno utilizzate per applicare la legge di propagazione delle incertezze relativamente alle cause d'incertezza prese in considerazione (floating point, jitter e quantizzazione).

6.5 Propagazione delle incertezze

Applichiamo ora la legge di propagazione delle incertezze alla relazione 6.8 che esprime il modello matematico della misura come parte reale e immaginaria; nell'applicare la legge così come espressa della GUM, considereremo il contributo delle incertezze dato dai singoli campioni costituenti la sequenza d'ingresso. Successivamente ricaveremo la medesima espressione relativamente alle 6.9, che esprime il modello come modulo e fase, e finalmente ricaveremo tutti i contributi all'incertezza finale, tra i quali potremo effettuare una comparazione ed eventualmente calcolare l'incertezza finale (anche se lo scopo del presente capitolo è

solo quello di valutare l'ordine di grandezza delle differenti cause di incertezza e non l'incertezza complessiva).

6.5.1 Quantizzazione

Applicando la legge di propagazione delle incertezze alla relazione 6.8, ossia esprimendo il modello della misura come parte reale ed immaginaria, otteniamo la seguente espressione:

$$\begin{aligned} U^2_{R(k)}\big|_q &= \sum_{m=0}^{N-1}\left(\frac{\partial R(k)}{\partial x(m)}\right)^2 U^2_{x(m)} = \sum_{m=0}^{N-1}\cos^2(k\beta_m)U^2_q \\ U^2_{I(k)}\big|_q &= \sum_{m=0}^{N-1}\left(\frac{\partial I(k)}{\partial x(m)}\right)^2 U^2_{x(m)} = \sum_{m=0}^{N-1}\sin^2(k\beta_m)U^2_q \end{aligned} \tag{6.10}$$

Ed essendo:

$$\begin{cases} \sum_{m=0}^{N-1}\cos^2(k\beta_m) = \begin{cases} N & k=0 \\ \frac{N}{2} & k\neq 0 \end{cases} \\ \sum_{m=0}^{N-1}\sin^2(k\beta_m) = \begin{cases} 0 & k=0 \\ \frac{N}{2} & k\neq 0 \end{cases} \end{cases} \tag{6.11}$$

La 6.10 diventa:

$$\begin{cases} U^2_{R(k)}\big|_q = \begin{cases} NU^2_q & k=0 \\ \frac{N}{2}U^2_q & k\neq 0 \end{cases} \\ U^2_{I(k)}\big|_q = \begin{cases} 0 & k=0 \\ \frac{N}{2}U^2_q & k\neq 0 \end{cases} \end{cases} \tag{6.12}$$

6.5.2 Jitter

Analogamente a quanto fatto in 6.5.1 otteniamo analoga relazione per il "time jitter", ricordando che il contributo all'incertezza dato ad ogni singolo campione dal time jitter dipende dalla pendenza del segnale in quel punto:

$$\begin{aligned} U^2_{R(k)}\big|_J &= \sum_{m=0}^{N-1}\left(\frac{\partial R(k)}{\partial x(m)}\right)^2 U^2_{J_m} = \sum_{m=0}^{N} \cos^2(k\beta_m) U^2_{J_m} \\ U^2_{I(k)}\big|_J &= \sum_{m=0}^{N-1}\left(\frac{\partial I(k)}{\partial x(m)}\right)^2 U^2_{J_m} = \sum_{m=0}^{N} \sin^2(k\beta_m) U^2_{J_m} \end{aligned} \tag{6.13}$$

6.5.3 Numeri in virgola mobile

Forniamo ora un'espressione per il calcolo dell'incertezza introdotta dall'errore di arrotondamento dei numeri in virgola mobile.

Introduciamo delle accettabili semplificazioni, ammettendo che le moltiplicazioni sono effettuate solo tra termini del tipo $x(n)\cdot cos(k\beta_n)$ e $x(n)\cdot sin(k\beta_n)$, e dunque ottenendo:

$$\begin{cases} fl[R(k)]_m &= \sum_{n=0}^{N-1} x(n)\cdot\cos(k\beta_n)\cdot(1+\gamma_n) \\ fl[I(k)]_m &= \sum_{n=0}^{N-1} x(n)\cdot\sin(k\beta_n)\cdot(1+\gamma_n) \end{cases} \tag{6.14}$$

Analogamente introduciamo una simile espressione per esprimere l'incertezza da errore di arrotondamento data dalle operazioni di somma e sottrazione; in questo caso limitiamoci ai soli termini del primo ordine, ottenendo:

$$\begin{cases} fl[R(k)]_a &= \sum_{n=0}^{N-1} x(n)\cdot\cos(k\beta_n) + \sum_{n=1}^{N-1}\eta_n\sum_{i=0}^{n} x(i)\cdot\cos(k\beta_i) \\ fl[I(k)]_a &= \sum_{n=0}^{N-1} x(n)\cdot\sin(k\beta_n) + \sum_{n=1}^{N-1}\eta_n\sum_{i=0}^{n} x(i)\cdot\sin(k\beta_i) \end{cases} \quad (6.15)$$

Dove η_n è l'incertezza introdotta dell'ennesima addizione.

Infine, applicando la procedura indicata dalla GUM alle 6.14 e 6.15 otteniamo le seguenti espressioni, ricordando che i termini con il pedice "*fl_m*" indicano le moltiplicazioni in virgola mobile e quelli con "*fl_a*" le addizioni:

$$\begin{cases} U^2_{R(k)}\big|_{fl\_m} &= \sum_{m=0}^{N-1}\left(\frac{\partial R(k)}{\partial\gamma_m}\right)^2 U^2_{fl\_m} = \sum_{m=0}^{N-1}(x(m)\cos(k\beta_m))^2 U^2_{fl\_m} \\ U^2_{I(k)}\big|_{fl\_m} &= \sum_{m=0}^{N-1}\left(\frac{\partial I(k)}{\partial\gamma_m}\right)^2 U^2_{fl\_m} = \sum_{m=0}^{N-1}(x(m)\sin(k\beta_m))^2 U^2_{fl\_m} \end{cases} \quad (6.16)$$

$$\begin{cases} U^2_{R(k)}\big|_{fl\_a} &= \sum_{m=1}^{N-1}\left(\frac{\partial R(k)}{\partial\eta_m}\right)^2 U^2_{fl\_a} = \sum_{m=1}^{N-1}\left(\sum_{i=1}^{m} x(i)\cos(k\beta_i)\right)^2 U^2_{fl\_a} \\ U^2_{I(k)}\big|_{fl\_a} &= \sum_{m=1}^{N-1}\left(\frac{\partial I(k)}{\partial\eta_m}\right)^2 U^2_{fl\_a} = \sum_{m=1}^{N-1}\left(\sum_{i=1}^{m} x(i)\sin(k\beta_i)\right)^2 U^2_{fl\_a} \end{cases} \quad (6.17)$$

6.6 Incertezza espressa su modello con modulo e fase

Applicando la legge di propagazione delle incertezze alle 6.9, modello della misura espresso come modulo e fase, otteniamo le seguenti importantissime espressioni:

$$U^2_{M(k)} = \frac{1}{N^4 M^2(k)}[R^2(k)U^2_{R(k)} + I^2(k)U^2_{I(k)} + 2R(k)I(k)U(R(k),I(k))] \quad (6.20)$$

$$U^2_{\varphi(k)} = \frac{1}{N^4 M^4(k)}[I^2(k)U^2_{R(k)} + R^2(k)U^2_{I(k)} - 2R(k)I(k)U(R(k),I(k))]. \tag{6.21}$$

Dove

$$U(R(k),I(k)) = \sum_{m=0}^{N-1} \frac{\partial R(k)}{\partial x_m} \cdot \frac{\partial I(k)}{\partial x_m} \cdot U^2_{x(m)} = \sum_{m=0}^{N-1} \cos(k\beta_m) \cdot \sin(k\beta_m) \cdot U^2_{x(m)} \tag{6.22}$$

Che esprime l'eventuale correlazione tra le grandezze *R(k)* e *I(k).*

Tramite queste espressioni, ed introducendo in esse i risultati ottenuti nei paragrafi precedenti, ossia il contributo delle cause espresse da 6.12, 6.13, 6.14, 6.15, riusciremo finalmente ad ottenere delle equazioni finali che ci consentiranno il calcolo dell'incertezza su modulo e fase delle differenti cause d'incertezza esaminate in questa sede.

.

6.5.4 Quantizzazione, modulo e fase

Notiamo che nella relazione 6.22 $\sum_{n=0}^{N-1} cos\,(k\,\beta_n)\,sin(k\beta_n) = 0$ per ogni k, possiamo finalmente desumere le relazioni seguenti:

$$U^2_{M(k)}\big|_q = \begin{cases} \frac{1}{N}U^2_q & k = 0 \\ \frac{1}{2N}U^2_q & k \neq 0 \end{cases}$$

$$U^2_{\varphi(k)}\big|_q = \begin{cases} \text{Se X Reale } \varphi(0) = 0 & k = 0 \\ \frac{1}{2N \cdot M^2(k)}U^2_q & k \neq 0 \end{cases} \tag{6.23}$$

Facciamo delle osservazioni sulle relazioni testè ricavate, come anche chiaramente riportato in [28]. Intanto è evidente che l'incertezza in valore assoluto sul modulo dipende dal valore dell'incertezza di ogni campione *x(n)*, la quale a sua volta dipende dal numero di bit del convertitore A/D e dal numero di punti usati per la DFT. Quindi, essa decresce al crescere del numero di bit e dal numero di punti usati. Circa la fase, anche in questo caso una relazione di proporzionalità inversa è evidenziata in funzione del numero di punti N, e dal modulo quadrato di M. Il che significa che toni a differente frequenza ma stessa ampiezza sono affetti dalla medesima incertezza (in valore assoluto) sulla fase. Supponendo che il numero effettivo di bit del convertitore A/D è costante nel range di frequenza usato, si può notare che l'effetto della quantizzazione non dipende dalla dinamica del segnale.

6.5.5 Jitter, modulo e fase

Sostituendo la 6.13 e la 6.22 nelle 6.20 e 6.21 otteniamo:

$$U^2_{M(k)}\big|_J = \frac{1}{N^4 M^2(k)}\left[R^2(k)\sum_{m=0}^{N-1}\cos^2(k\beta_m)U^2_{J_m} + I^2(k)\sum_{m=0}^{N-1}\sin^2(k\beta_m)U^2_{J_m} + 2R(k)I(k)\sum_{m=0}^{N-1}\sin(k\beta_m)\cos(k\beta_m)U^2_{J_m}\right]$$

$$U^2_{\varphi(k)}\big|_J = \frac{1}{N^4 M^4(k)}\left[I^2(k)\sum_{m=0}^{N-1}\cos^2(k\beta_m)U^2_{J_m} + R^2(k)\sum_{m=0}^{N-1}\sin^2(k\beta_m)U^2_{J_m} + 2R(k)I(k)\sum_{m=0}^{N-1}\sin(k\beta_m)\cos(k\beta_m)U^2_{J_m}\right]. \quad (6.24)$$

6.5.6 Errore di arrotondamento, modulo e fase

Introducendo nelle espressioni generali 6.21 e 6.22 le 6.14 e 6.15 otteniamo:

$$
\begin{aligned}
&U^2_{M(k)}\big|_{fl\_m} \\
&= \frac{1}{N^4 M^2(k)}\left(R^2(k)\sum_{m=0}^{N-1}(x(m)\cos(k\beta_m))^2 U^2_{fl\_m} \right. \\
&\quad \left. + I^2(k)\sum_{m=0}^{N-1}(x(m)\sin(k\beta_m))^2 U^2_{fl\_m} \right) \\
&U^2_{\varphi(k)}\big|_{fl\_m} \\
&= \frac{1}{N^4 M^4(k)}\left(I^2(k)\sum_{m=0}^{N-1}(x(m)\cos(k\beta_m))^2 U^2_{fl\_m} \right. \\
&\quad \left. + R^2(k)\sum_{m=0}^{N-1}(x(m)\sin(k\beta_m))^2 U^2_{fl\_m} \right)
\end{aligned}
\tag{6.25}
$$

$$
\begin{aligned}
&U^2_{M(k)}\big|_{fl\_a} \\
&= \frac{1}{N^4 M^2(k)}\left(R^2(k)\sum_{m=1}^{N-1}\left(\sum_{i=0}^{m}x(i)\cos(k\beta_i)\right)^2 U^2_{fl\_a} \right. \\
&\quad \left. + I^2(k)\sum_{m=1}^{N-1}\left(\sum_{i=0}^{m}x(i)\sin(k\beta_i)\right)^2 U^2_{fl\_a} \right) \\
&U^2_{\varphi(k)}\big|_{fl\_a} \\
&= \frac{1}{N^4 M^4(k)}\left(I^2(k)\sum_{m=1}^{N-1}\left(\sum_{i=0}^{m}x(i)\cos(k\beta_i)\right)^2 U^2_{fl\_a} \right. \\
&\quad \left. + R^2(k)\sum_{m=1}^{N-1}\left(\sum_{i=0}^{m}x(i)\sin(k\beta_i)\right)^2 U^2_{fl\_a} \right)
\end{aligned}
\tag{6.26}
$$

$$U^2_{M(k)}\big|_{fx} = \frac{1}{N} \cdot U^2_{fx} \tag{6.27}$$

$$U^2_{\varphi(k)}\big|_{fx} = \frac{1}{NM^2(k)} \cdot U^2_{fx}. \tag{6.28}$$

6.7 Calcolo dell'incertezza su un caso pratico e confronto

Sfruttando le relazione analizzate in dettaglio nei paragrafi precedenti, ed effettuando i calcoli tramite apposite routine scritte "ad-hoc" nel programma, passiamo ad effettuare il calcolo dei vari contributi d'incertezza. Da notare due cose; in primis l'incertezza relativa al jitter e la quantizzazione sono ovviamente "residenti" nella scheda di acquisizione; mentre l'incertezza dovuta al rounding error è chiaramente dovuta all'algoritmo utilizzato. Nella realtà, l'algoritmo utilizzato è la FFT (Fast Fourier Transform) la cui complessità computazionale è decisamente inferiore (in termini di operazioni effettuate) rispetto alla DFT. In particolare è noto che la prima implica un numero di operazioni proporzionali al logaritmo del numero di punti mentre la seconda è proporzionale al quadrato degli stessi. In tal senso, il calcolo dell'incertezza sulla DFT conduce a risultati sicuramente peggiori di quelli su FFT, a parità di condizioni. Per i risultati ottenuti in questo paragrafo si veda anche [3]. Il test è stato condotto usando i seguenti parametri:

Risoluzione in bit scheda di acquisizione	16
Frequenza di campionamento	40960 Hz
Numero di punti	4096
Frequenza di test	1000 Hz
Numero reale in virgola mobile	IEEE 80 bit. Mantissa 64 bit.

La scelta della frequenza di campionamento è delle dimensioni del buffer, nonché la frequenza scelta evidenziano una condizione di campionamento sincrono (cfr. cap. VII). L'uso delle variabili floating point in standard IEEE a 80 bit è stato esteso a tutte le variabili del programma coinvolte nel calcolo degli algoritmi utilizzati negli strumenti di misura.

E' stata aggiunta una sezione di codice, non normalmente fruibile dagli utilizzatori "standard" del programma per effettuare il calcolo "white box" ed in tempo reale dell'incertezza associata alle tre cause evidenziate e con i parametri riportati in tabella.

In particolare, per il calcolo dell'incertezza sulla quantizzazione è stata utilizzata le formule 6.23, per il jitter le 6.24 (ipotizzando un valore medio di pendenza, vedi 6.6) e per il calcolo del rounding error dovuto alle moltiplicazioni in virgola mobile e somme/sottrazioni rispettivamente 6.25 e 6.26.

I risultati ottenuti sono riportati nella tabella seguente:

Rounding error	*Quantizzazione*	*Jitter*
$3.151169 \cdot 10^{-17}$ %	$4.866698 \cdot 10^{-6}$ %	$3.189439 \cdot 10^{-3}$ %

Tabella 1: Incertezza espressa come relativa rispetto al valore atteso

Dalla tabella si evince chiaramente che utilizzando variabili in virgola mobile IEEE a 80 bit (long double), in sinergia con la risoluzione a 16 bit del convertitore, l'incertezza associata all'errore di arrotondamento è nettamente trascurabile rispetto alle altre due cause; essendo queste ultime dovute all'hardware si può trarre l'evidente conclusione che le cause d'incertezza si "spostano" completamente sull'hardware.

Pertanto, nel presente lavoro eviteremo di considerare i contributi, pur sempre presenti, che l'errore di arrotondamento porta nella determinazione dell'incertezza composta. Si noti altresì che i calcoli sono stati effettuati per la versione "pesante", in termini computazionali, dell'algoritmo, ossia relativamente alla DFT e non alla FFT

che darebbe risultati ancora migliori; infatti per quest'ultima, come detto in inizio di paragrafo, si ha un numero di operazioni (moltiplicazioni e addizioni) significativamente inferiore all'algoritmo "canonico".

Pagina lasciata intenzionalmente bianca

Capitolo 7

FFT e cause d'incertezza da essa introdotte

7.1 Introduzione

In questo capitolo prenderemo in considerazione l'algoritmo più complesso utilizzato nel programma, come già indirettamente fatto nel capitolo precedente, al fine di evidenziare le cause d'incertezza da esso introdotte (tipiche dell'algoritmo FFT o DFT e non generali come il rounding error), le soluzioni adottate per la loro eliminazione/riduzione presenti in letteratura, e infine porremo l'accento su quelle effettivamente considerate in VA. Questo approfondimento si rende necessario perché l'FFT è alla base di un'altra importante funzione implementata nel programma, descritta nel capitolo successivo, che si basa sull'uso dei dati ricavati dall'analizzatore di spettro; ed anche in altri, oltre naturalmente all'analizzatore di spettro in se. L'incertezza introdotta dall'FFT si "propaga" negli algoritmi eventualmente utilizzati per gli strumenti basati su di essa, sino ad influire sulla misura in maniera consistente. Metteremo innanzi tutto in evidenza lo schema generale teorico di uno strumento "analizzatore di spettro" basato su FFT (o su DFT, Discrete Fourier Transform, FFT fornisce esattamente gli stessi risultati ma più velocemente) così come poi è stato praticamente implementato nel programma. Si farà accenno al problema delle "finestrature" che costituiscono una tecnica di buon livello per la riduzione degli errori introdotti dallo "spezzettamento" dei dati in ingresso (trasformata di Gabor). Invero, per lo strumento di maggiore rilevanza

presentato in questa tesi (ZRLC, vedi cap. VIII) che pur si basa sullo strumento "Analizzatore di Spettro", le particolari tecniche adottate hanno fatto si che la finestratura non sia del tutto necessaria, ed anzi viene addirittura evitata. Per la stesura di questo capitolo ci siamo avvalsi di [30] con esplicita autorizzazione dell'autore che ha fornito validi consigli e materiale bibliografico.

7.2 Spettro di un segnale, DFT, FFT e trasformata di Gabor

Lo spettro di un segnale viene generalmente ricavato per tramite di una DFT, dall'Inglese "Discrete Fourier Transform". Esso risulta essere un algoritmo assai poco efficace dal punto di vista di un elaboratore, ed anzi, nella pratica esso non è proponibile per il calcolo in tempo reale dello spettro di un segnale campionato; questo sino alla scoperta di un nuovo ed efficace algoritmo, noto come FFT (Fast Fourier Transform). La scoperta è stata effettuata verso la metà degli anni 60 ad opera di Cooley e Tukey [47]; per esso l'ordine di complessità computazionale è $O(Nlog(N))$ (data una sequenza di N punti), mentre per la DFT è $O(N^2)$. E' evidente la maggiore convenienza in termini di risorse elaborative richieste.

Per gli scopi del presente lavoro, siamo esclusivamente interessati a spettri di segnali tempo discreti e quantizzati, ossia approssimati con numeri memorizzabili nella RAM di un comune elaboratore, e dunque con cardinalità finita. In altre parole, un numero reale di per se infinito viene *approssimato* con una quantità finita.

Dato un segnale analogico $x(t)$ sottoposto ad un processo di campionamento, è possibile, a partire da una sua sequenza di N campioni sui quali è applicata la trasformata discreta di Fourier (DFT) (o equivalentemente la FFT) calcolare sequenza $X(k)$ rappresentante i campioni della trasformata di Fourier $X(f)$ del segnale analogico di partenza.

Lo spettro di un segnale risulta dunque essere la sequenza $X(f)$, in generale di tipo complesso, dalla quale potremo ricavare lo spettro di ampiezza e lo spettro di fase.

Essi rappresentano lo stesso segnale di partenza analogico, ma rappresentato, come si usa dire, nel "dominio della frequenza". Di converso, il segnale originario, campionato e non, si dice rappresentato "nel dominio del tempo". La rappresentazione nel domino del tempo è quella che, in termini pratici, siamo abituati a vedere nell'oscilloscopio. Quella nel domino della frequenza è invece una modalità di rappresentazione in cui la variabile tempo scompare, ed il segnale è visualizzato in termini delle sue componenti costitutive alle varie frequenze, che nell'ottica della trasformata di Fourier (continua e discreta) sono segnali armonici puri (sinusoidi e cosinusoidi) considerati come "atomici" ossia non composti da altri segnali. In altre parole ancora, qualsiasi segnale risulta composto da una somma (finita o infinita) di segnali elementari di diversa frequenza. L'unico segnale composto "da se stesso" è di tipo sinusoidale. Lo spettro di ampiezza di un segnale sarà allora composto da un grafico che in ascissa avrà la frequenza delle armoniche componenti ed in ordinata l'ampiezza delle stesse; e da un diagramma di fase, che è un analogo grafico, che dunque avrà in ascissa la frequenza delle varie componenti armoniche ed in ordinata la fase espressa in gradi o radianti. Nel corso del presente lavoro saranno presentati numerosi esempi di spettri, intesi come l'insieme dei due diagrammi.

Come vedremo, c'è da fare attenta distinzione tra segnali periodici e non, e conseguente rappresentazione spettrale; il concetto di base, ossia la scomposizione in somma di segnali elementari resta comunque valido (sebbene, ovviamente, taluni segnali saranno rappresentati si da somme di segnali elementari, ma con frequenze variabili con continuità su un dominio reale e con sommatorie integrali).

Per poter fare un uso pratico della FFT, la cui definizione implica una sommatoria estesa a tutti i punti del segnale, dunque al limite infinita, è necessario introdurre l'elemento tempo. In tal senso negli strumenti reali che implementano analizzatori di spettro per tramite del calcolo di una FFT su sequenze (finite) di campioni digitalizzate, viene usata la cosiddetta trasformata di Gabor. Essa consiste essenzialmente in una FFT effettuata su porzioni del segnale di ugual dimensione, in cui possiamo immaginare scomposta la sequenza (infinita) del segnale d'ingresso. In altri termini, il segnale d'ingresso verrà "impacchettato" a blocchi di dimensione

predefinita cui corrisponderà un certo intervallo temporale. Su ognuno di questi pacchetti verrà effettuata l'operazione di calcolo dello spettro per tramite di FFT, ottenendo dunque una "successione temporale di spettri", spettri che verranno disegnati a video sequenzialmente. Per esempio, scegliendo pacchetti di dimensione pari a 100 punti, supponendo che i punti siano prelevati tramite campionamento di un segnale analogico ogni 0.01 secondi avremo che una porzione di 100 punti corrisponderà a 1 secondo di segnale. Lo spettro verrà dunque calcolato ogni secondo e ogni secondo lo schermo dello strumento verrà aggiornato con il nuovo spettro calcolato. Il valore 100 è stato scelto a titolo puramente esemplificativo giacché, come è noto e vedremo a breve, l'FFT non può essere effettuata solo su un numero di punti arbitrari.

L'adattamento della FFT operato da Gabor (Dennis Gabor, 1946) è chiamato STFT (Short-Time Fourier Transform) ed è un modo per aggiungere la dimensione temporale alla trasformata di Fourier, sebbene in tal modo vengano introdotte delle evidenti approssimazioni (e dunque cause d'incertezza).

Va da se che il calcolo dello spettro può essere effettuato fisicamente nel tempo che intercorre tra due "pacchetti" di dati, e dunque diventa essenziale che l'algoritmo usato sia sufficientemente veloce da consentire il calcolo, date le risorse elaborative disponibili. Come detto, la DFT consente una efficienza di calcolo proporzionale al quadrato dei punti, mentre FFT proporzionale al numero dei punti moltiplicato il logaritmo degli stessi. Dunque drasticamente più efficiente ed in linea con le risorse elaborative disponibili sui computer meno potenti.

7.3 Schema generale della trasformata di Gabor

Lo schema a blocchi di fig. 49 rappresenta la struttura di base per il calcolo della trasformata di Gabor (tratta da [33]).

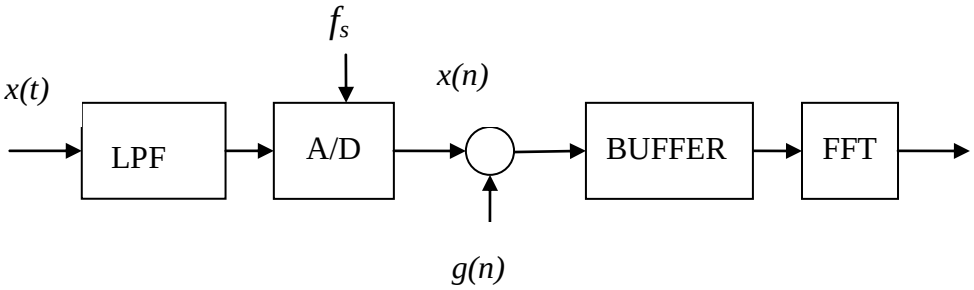


Figura 49: trasformata di Gabor

In essa si possono osservare i seguenti blocchi funzionali:

1. LPF = Low Pass Filter, è un filtro passa basso necessario per il condizionamento del segnale in vista del campionamento; serve ad evitare il problema dell'Aliasing;
2. A/D, è il convertitore Analogico digitale; serve a convertire il segnale tempo continuo in un segnale tempo discreto e quantizzato; in altre parole un segnale memorizzabile nella RAM di un computer; il segnale in uscita è rappresentato dalla sequenza *x(n)* la cui durata è virtualmente infinita, e infinita sarebbe la sommatoria che definisce la trasformata di Fourier approssimata tramite la FFT; la sequenza *x(n)* viene spezzata in "pezzi" di lunghezza finita; questa sequenza di durata finita viene ottenuta tramite moltiplicazione per la sequenza *g(n)*, di durata finita anch'essa e utilizzata per implementare quella operazione di compensazione detta finestratura, atta a diminuire quanto più possibile tutti i problemi che trasformata di Gabor comporta (v. avanti);
3. BUFFER è il buffer che memorizza i campioni del segnale a blocchi di dimensione predefinita e pronti per essere trasformati dalla FFT;
4. FFT implementa l'operazione di trasformata sui blocchi memorizzati in ram.

In sintesi, il segnale analogico viene convertito in una sequenza discreta; la sequenza discreta suddivisa in blocchi di lunghezza fissa (v. fig. 50)

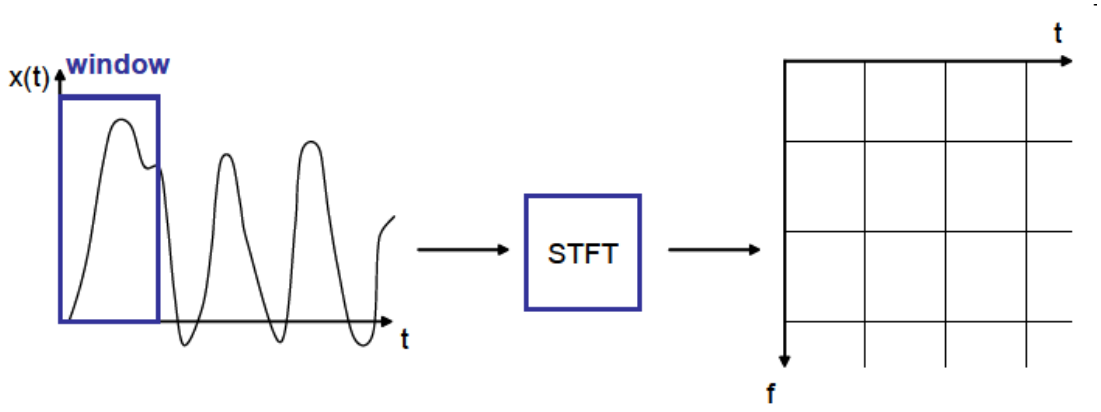


Figura 50: suddivisione in blocchi del segnale

Su questi blocchi viene effettuata l'operazione di trasformazione per tramite della FFT, equivalente ad essa in tutto e per tutto. L'unica limitazione che l'uso della FFT comporta è che l'insieme di punti su cui può essere effettuata l'operazione di FFT deve necessariamente essere una potenza di 2, ossia 2^n con "*n*" intero. Sicché, per esempio, valori validi per le dimensioni del buffer possono essere 2048, 4096, 8192 eccetera.

Nel prossimi paragrafi vedremo che il campionamento e lo spezzettamento del segnale comporta tre problemi principali noti come *Aliasing*, *Dispersione Spettrale* e *Interferenza Armonica*. Essi sono causa dell'introduzione di una serie di "errori" o meglio di "incertezze" quando le sequenze calcolate tramite FFT vengono usate in strumenti di misura. Una delle soluzioni che vedremo, accennata nelle sue linee principali nei prossimi paragrafi, è quella della finestratura del segnale, che consiste nel moltiplicare ogni "pacchetto" di campioni per una sequenza predefinita, indicata nella figura 49 come *g(n)*. Il segnale che dunque andremo a trasformare non sarà il segnale *x(n)* ma il prodotto *x(n)g(n)*. Su tale base, diamo le formule principali della DFT. In linea generale, una finestra *g(n)* è data dalla seguente relazione:

$$G(k) = \sum_{n=0}^{N-1} g(n) \exp\left(-j\frac{2\pi}{N}kn\right) = \sum_{n=0}^{N-1} g(n) W_N^{-kn} \qquad (7.1)$$

$$\text{per } k = 0,1,\ldots,N\text{-}1$$

Dove: $W_N = \exp\left(j\frac{2\pi}{N} \right)$

La DFT (o FFT) verrà dunque effettuata sulla sequanza complessiva *g(n)x(n)* e dunque potremo scrivere:

$$X_g(k) = \frac{1}{S}\sum_{n=0}^{N-1} x_g(n)\exp\left(-j\frac{2\pi}{N}kn \right) = \frac{1}{S}\sum_{n=0}^{N-1} x(n)g(n)W_N^{-kn} \qquad (7.2)$$
$$\text{per } k = 0,1,\ldots,N\text{-}1$$

Dove: $S = \sum_{n=0}^{N-1} g(n)$ è il fattore di normalizzazione, necessario per non alterare le ampiezze dei segnali in gioco per effetto dell'operazione di finestratura.

Cerchiamo ora di chiarire esaustivamente i problemi che l'operazione di finestratura comporta. Ricapitoliamo:

- Aliasing (7.3.1);
- Dispersione Spettrale (Spectral Leakage) (7.3.2);
- Interferenza Armonica (Harmonic Interference) (7.3.3).

7.3.1 Aliasing

Questo problema è invero molto più generale, e non specifico di una FFT, parimenti essendo l'FFT qui considerata effettuata su sequenze ottenute tramite operazioni di campionamento e quantizzazione, quale quelle operate da un convertitore A/D, è da considerarsi un problema (anche) ad essa associato. I dettagli di questo problema sono descritti nel capitolo sui sistemi digitali, ove verrà data una definizione esaustiva del teorema alla base del funzionamento dei convertitori A/D, ossia il teorema del campionamento o di Shannon-Nyquist. Esso afferma, a grandi linee, che un segnale analogico può essere completamente ed esaustivamente rappresentato da una sequenza discreta da esso ottenuta sotto certe condizioni; il segnale che se ne ricava è invero un segnale costituito da uno spettro che è la replica infinita dello spettro originale del segnale. Quando vengono rispettate le "regole" dettate dal

teorema del campionamento, gli spettri periodici sono ben distanziati tra loro, ed un'opportuna operazione di filtraggio (passa basso) è sufficiente a ricostruire il segnale originale. Una non corretta applicazione delle ipotesi alla base del teorema del campionamento, porta a spettri periodici che si sovrappongono e dunque interferiscono tra loro, rendendo impossibile l'operazione di ricostruzione tramite filtraggio passa basso (Aliasing).

Una delle condizioni alla base del teorema del campionamento è che la frequenza usata per il campionamento del segnale sia almeno doppia della massima frequenza contenuta nel segnale stesso; allora, per evitare che possano esserci componenti armoniche non consentite, viene generalmente "trattato" il segnale originale con un filtro passa basso, corrispondente al blocco LPF della figura 49, che assicura la necessaria limitatezza in banda, data la frequenza di campionamento f_s (rappresentata sempre nella medesima figura).

7.3.2 Dispersione spettrale

La dispersione spettrale è un fenomeno direttamente legato all'operazione di suddivisione del segnale reale in blocchi di dimensione predefinita; l'FFT in genere considera che il segnale di cui sta effettuando l'operazione di trasformata è periodico; con un periodo pari all'intervallo di osservazione $T_w=N\Delta t$, dove N è il numero di punti su cui si valuta la DFT, mentre Δt è il periodo di campionamento del segnale di analisi. Se il numero di punti su cui si valuta l'FFT rispetta le condizioni dette di "campionamento sincrono" allora il fenomeno di dispersione spettrale non si manifesta.

Il campionamento è detto *sincrono* quando il periodo di osservazione T_w è un multiplo intero del periodo T del segnale osservato; e nel caso di segnale periodico con più componenti armoniche (multifrequenza) la condizione di campionamento sincrono implica che il periodo di osservazione sia un multiplo intero del *periodo comune* alle varie componenti armoniche del segnale osservato. Se dunque si verifica la condizione di campionamento sincrono, e la funzione di finestratura $g(n)$ è una

finestra rettangolare, ossia una "non-finestra", ed inoltre il numero di punti su cui si effettua la DFT sono tutti i punti contenuti nel periodo T_w, allora non si verificherà alcuna dispersione spettrale.

In caso contrario, appare il fenomeno della dispersione spettrale; essa consiste in un allargamento anomalo dello spettro del segnale (comparsa di lobi secondari ed un lobo principale di elevata ampiezza). Esso è intuitivamente comprensibile pensando che l'operazione di spezzettamento del segnale implica l'introduzione di nuove armoniche per "brusco troncamento" del segnale ai suoi estremi. Si pensi infatti ad un semplice segnale sinusoidale; lo spettro d'ampiezza di questo sarebbe costituito da una sola riga spettrale, ossia un impulso nel dominio della frequenza (una singola retta verticale). L'FFT di un segnale sinusoidale troncato ai suoi estremi destro e sinistro in maniera casuale, implica che il segnale diventi composto dall'armonica del segnale originario (fondamentale), più altre armoniche generate dal prodotto del segnale sinusoidale per due gradini, ossia bruscamente troncato. L'allargamento dello spettro si presenta (a grandi linee) come in figura 51.

Figura 51: effetto del troncamento del segnale; a sinistra spettro originale di un segnale sinusoidale a frequenza f0; a destra spettro "allargato" per effetto del troncamento; sono presenti altre armoniche e dunque lobi secondari e lobo principale.

Il problema è che ottenere condizioni di campionamento sincrono è estremamente difficile. Intanto perché il periodo del segnale può essere non noto; si potrebbe pensare di adattare la frequenza di campionamento al segnale, ma anche questo può non essere possibile per motivi tecnici; per finire, perché nel segnale esistono componenti non armoniche, ossia non multiple di una stessa frequenza fondamentale.

Ammesso dunque che in generale non è possibile ottenere condizioni di campionamento sincrono, si potrebbe pensare di utilizzare finestre *g(n)* diverse da una semplice finestra rettangolare, in maniera da minimizzare l'effetto della

dispersione spettrale. Questa è in effetti la soluzione generalmente adottata, ed anche in VA stesso vi è la possibilità di scegliere tra un numero consistente di finestre "standard" che consentono di ridurre in maniera consistente il fenomeno, in ragione della natura del segnale analizzato e dell'uso che si deve fare dello spettro (misure, analisi di segnali con particolari caratteristiche, etc.).

Come agisce la FFT

E' importante a questo punto comprendere come funziona qualitativamente l'FFT. Essa agisce su di un segnale per ipotesi periodico, indipendentemente dal fatto che esso lo sia effettivamente. Infatti, l'algoritmo "suppone" che la porzione di segnale su cui stia operando sia proprio un segnale periodico avente periodo pari alla lunghezza del segnale passato, indipendentemente dal fatto che esso sia periodico effettivamente. Per meglio comprendere questo importantissimo aspetto, si osservi la figura seguente:

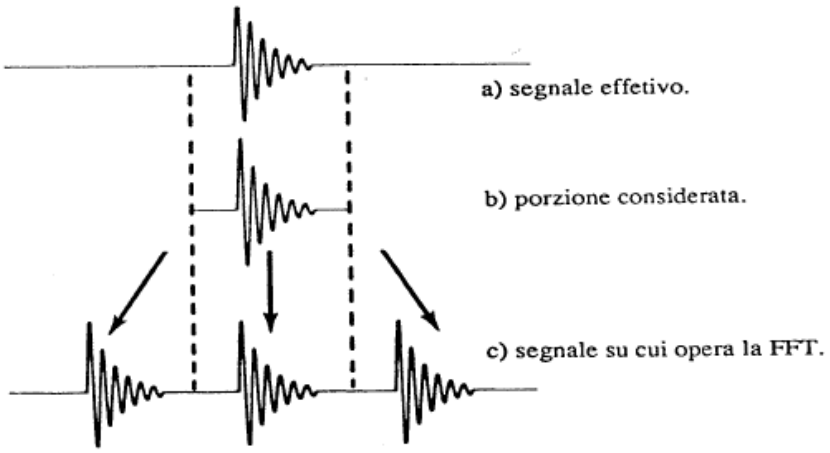


Figura 52: come opera FFT

Il segnale in (a) non è affatto periodico, e comunque la porzione di segnale considerata dopo aver effettuato la suddivisione in blocchi comprende proprio (esaustivamente) la porzione del segnale (tipicamente) transitorio, ossia quello rappresentato in (b). Se immaginiamo di trasformare il segnale in (b) che è quello "passato" alla FFT, e successivamente di anti-trasformarlo per ottenere il segnale originario, otterremo il segnale in (c) completamente diverso dall'originale. Ossia l'FFT opera effettivamente sul segnale (c) dunque diverso dal segnale reale.

Supponiamo di effettuare la stessa cosa su di un segnale periodico sinusoidale; si veda la figura 53.

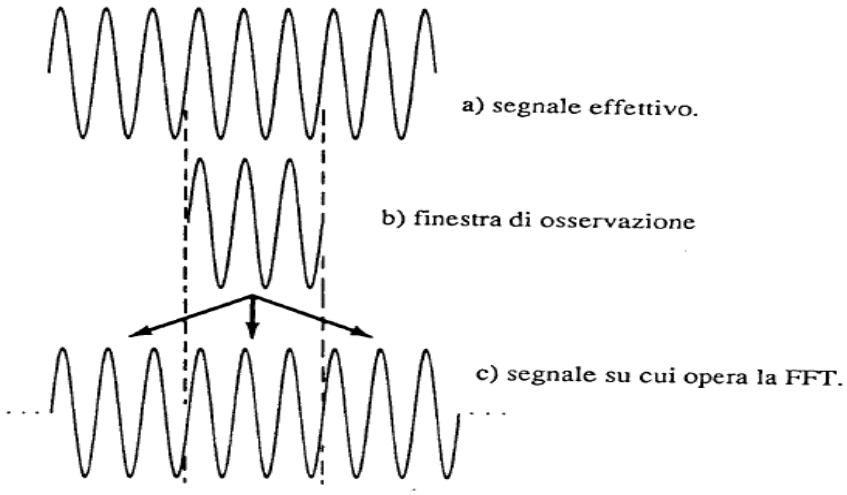


Figura 53: segnale segmentato in maniera corretta (sincrona)

In questo caso, sono rispettate le condizioni di campionamento sincrono, ed il segnale in (c) risulta dall'antitrasformazione della trasformata del segnale in (b) che risulta dunque la ripetizione infinita del segnale (b) la cui repliche si "agganciano" perfettamente. Vediamo ora cosa accade se il segnale non è stato correttamente suddiviso (ossia senza campionamento sincrono); nella figura successiva si può osservare chiaramente come il segnale che otteniamo in "c" sia un segnale completamente differente!

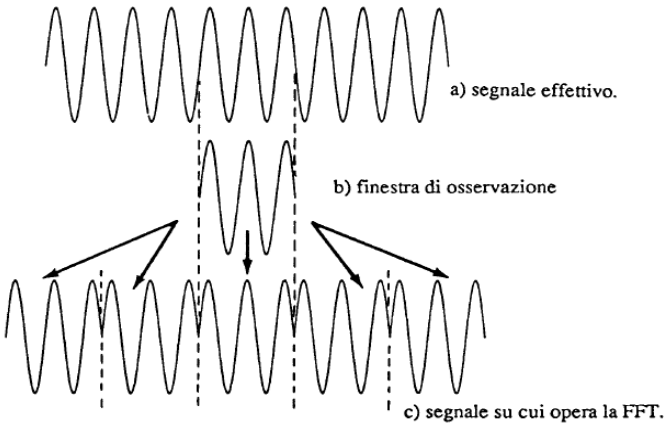


Figura 54: segnale segmentato in maniera non corretta (asincrona)

Come limitare i problemi

Come suggerito, l'idea è dunque, nell'impossibilità di effettuare un campionamento sincrono, quella di scegliere per la finestra *g(n)* una forma tale che consenta di minimizzare gli effetti descritti dalla sezione precedente; in particolare, l'idea generale è quella di dare "minor peso" ai campioni agli estremi del segnale, ossia proprio dove vengono introdotte le armoniche spurie per effetto del brusco troncamento. Il concetto può meglio essere chiarito facendo riferimento alla figura successiva.

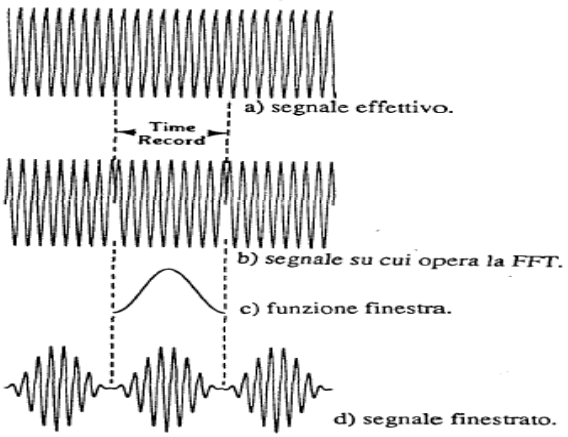


Figura 55: uso di finestre diverse dalla rettangolare

In essa si vede chiaramente che l'uso di una finestra diversa da quella rettangolare, comporta una ri-periodizzazione del segnale che almeno a livello di spettro implica un contenuto armonico meno alterato. La trasformata del segnale così trattato comporterebbe un restringimento del lobo principale di figura 50, e sicuramente una paritetica riduzione dei lobi secondari. In letteratura esistono un numero enorme di finestrature possibili, ed in particolare una categoria di esse, detta "flat-top" fornisce ottimi risultati in relazione al problema dell'incertezza di misura. Un trattamento adeguato di questo argomento esula dagli scopi di questo capitolo puramente descrittivo e qualitativo e si rimanda a [33] ed alla nutrita bibliografia in esso riportata sull'argomento.

7.3.3 L'interferenza armonica

Questo fenomeno è assai legato a quello descritto nel precedente paragrafo; si presenta quando nel segnale da analizzare esistono componenti armoniche molto vicine tra loro. L'effetto del campionamento asincrono, e dunque la comparsa di allargamenti spettrali fa si che essi si sovrappongano in maniera distruttiva rendendo impossibile la distinzione delle armoniche originali.

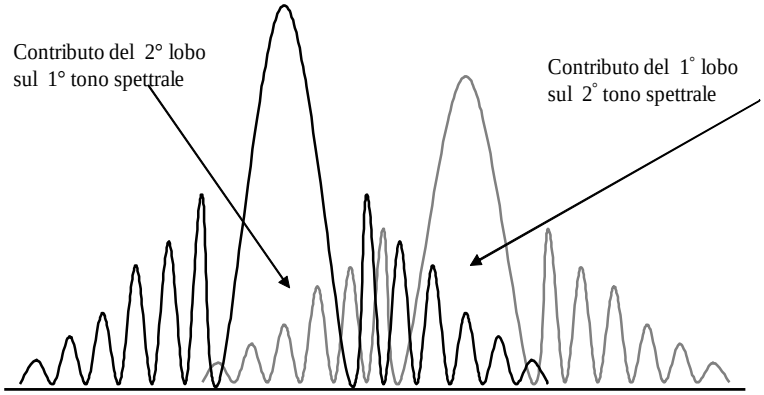


Figura 56: interferenza armonica, lobi interagenti

L'utilizzo di opportune finestra *g(n)* può ridurre in maniera consistente la dispersione spettrale e di conseguenza l'interferenza armonica. In molti casi questo non è possibile, e bisogna ricorrere ad altri metodi. La sovrapposizione dei lobi principali e secondari può in taluni casi rendere indistinguibili componenti armoniche di fatto distinte.

Si riportano le definizioni di due importanti indici usati per quantificare i fenomeni descritti, usati poi per dimensionare le finestre g(n)

a) Attenuazione del primo lobo laterale α_{SL}.

Essa è definita come il rapporto, espresso in decibel (dB), tra il picco del lobo principale $|G(0)|$ e quello del primo lobo laterale $|G(f_1)|$, ove f_1 è la frequenza alla quale si ha il massimo del primo lobo laterale; in tal caso

$$\alpha_{SL} = 20 \log_{10} \frac{G(0)}{G(f_1)} \tag{7.3}$$

Da notare che spesso il valore di f_1 non è noto, ed allora si considera il picco del primo lobo laterale al centro di tale lobo laterale, ovvero alla frequenza $f_{0,}$ in tal caso

$$\alpha_{SL} = 20 \log_{10} \frac{G(f)}{G(f_0)} \qquad (7.4)$$

In genere $f_1 \cong f_0$, e quindi le due definizioni sono in pratica coincidenti.

b) Decadimento asintotico dello spettro di ampiezza, noto come "roll-off".

È la pendenza con cui decadono i picchi dei lobi laterali, o, più precisamente, l'inviluppo dello spettro di ampiezza. Questo parametro è espresso in dB/ottava.

Capitolo 8

ZRLC

8.1 Introduzione

In questo capitolo presentiamo uno strumento virtuale capace di misurare l'impedenza di un generico bipolo. E dunque, sebbene superfluo da precisare, resistenza, capacità e induttanza, oltre naturalmente alla rappresentazione grafica del modulo e della fase (oltre che numerica). Inoltre, è possibile acquisire sequenze di misurazioni nel dominio del tempo e della frequenza, ottenerne grafici memorizzabili ed esportabili, e non ultimo effettuare il calcolo dell'incertezza.

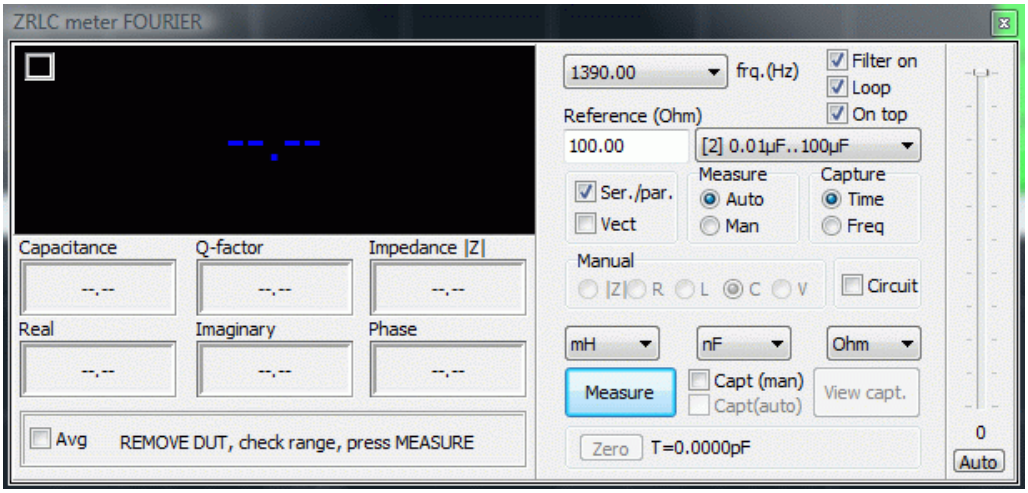


Figura 57: strumento ZRLC

Senza molta fantasia abbiamo chiamato questo strumento “ZRLC” (vedi fig. 57); esso necessita di un hardware dedicato che deve essere aggiunto alla scheda di acquisizione dati. Invero, l’hardware necessario è decisamente semplice, come vedremo più avanti, ma di esso ne presenteremo una versione più completa nel prossimo capitolo, sviluppata in collaborazione con la rivista “Nuova Elettronica” [41] che include anche la scheda di acquisizione. I concetti che verranno esposti nel presente capitolo si applicano indifferentemente al circuito “semplice” abbinato alla scheda di acquisizione e al circuito “completo” appositamente sviluppato.

Il capitolo precedente ha messo in luce le varie cause di incertezza che l’uso dell’algoritmo FFT (e/o anche DFT) può introdurre; a queste si aggiungano le cause di incertezza “generali” individuate nel cap. IV, ed altre specifiche del sistema di acquisizione dati a due canali che usiamo per implementare gli strumenti virtuali oggetto di questo lavoro. In questo capitolo presenteremo le metodologie adottate per ridurre o eliminare le cause di incertezza (comprese nelle categorie elencate nel capoverso precedente) che abbiamo individuato come “errori sistematici” e dovute all’hardware; sceglieremo un algoritmo alternativo a quelli tipicamente utilizzati in letteratura per la misura dell’impedenza (normalmente LMS, vedi apposito paragrafo) che fa uso dell’FFT, e dunque cercheremo per esso di ridurre al minimo possibile le cause d’incertezza appositamente individuate nel capitolo precedente.

In ultimo, si osservi che questo strumento si presta ad essere usato anche per la misura di impedenze generiche, ossia non necessariamente costituite da un bipolo semplice: è infatti possibile misurare le impedenze di amplificatori e circuiti complessi in genere. Le misure possono avvenire in tutto il range di frequenze permesso dalla scheda di acquisizione/generazione, consentendo di evidenziare il variare dell’impedenza al variare della frequenza di test.

8.2 Il circuito

Il semplice circuito che dovremo abbinare alla scheda di acquisizione a due canali (ad elevata impedenza d’ingresso) è il seguente (fig. 58):

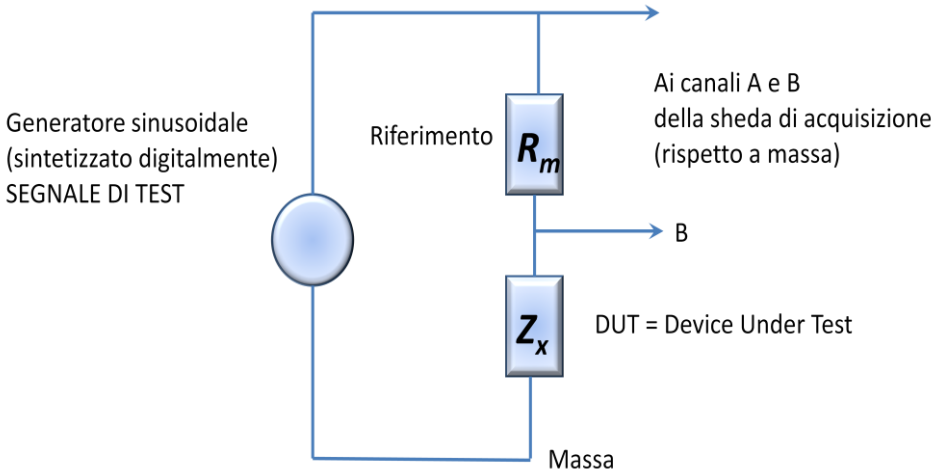


Figura 58: il circuito aggiuntivo

Esso è relativamente semplice; la misura dell'impedenza incognita Z_x avviene tramite la lettura delle tensioni rilevate al nodo A e B rispetto a massa; il segnale di test è sinusoidale e generato dal programma (usando le "uscite" della scheda) ed applicata ai capi del partitore resistivo formato dalla serie di Z_x ed R_m; quest'ultimo è il "resistore di riferimento" e si suppone di valore noto. Vedremo in seguito che l'accuratezza con cui si conosce il valore di quest'ultimo resistore determinerà in maniera non trascurabile l'incertezza di misura associata a Z_x. A tal proposito si osservi che il valore della resistenza può essere ricavato da quanto dichiarato dal costruttore (valore nominale e tolleranza) oppure misurato con uno strumento che presenti un'incertezza di misura nota e migliore di quella dichiarata dal costruttore del componente. Su questo argomento torneremo a più riprese in vari paragrafi.

8.2.1 Misura e relazione usata

La relazione che consente di calcolare l'impedenza (in modulo) è la seguente:

$$|Z_x| = \frac{|V_{zx}|}{|V_{rm}|} R_m \qquad (8.1)$$

V_{zx} si ricava immediatamente (dati letti da canale B) mentre V_{rm}, ossia la tensione ai capi del resistore di riferimento, si può ottenere come differenza tra le tensioni lette ai nodi A e B ossia:

$$V_{rm} = V_A - V_B \tag{8.2}$$

La conoscenza "completa" dell'impedenza si può conoscere calcolando lo sfasamento tra V_{rm} e V_{zx} essendo la prima la tensione ai capi della resistenza di riferimento, e dunque proporzionale alla corrente che scorre nell'impedenza incognita ($I = V_{rm} / R_m$), la seconda la tensione ai capi della stessa. Per calcolare lo sfasamento tra le tensioni è necessario applicare un algoritmo che consenta la stima di esso in maniera quanto più accurata possibile. Una possibilità sarebbe quella di individuare il passaggio per lo zero delle tensioni (zero crossing) e dunque calcolare lo sfasamento relativo tra le due come differenza tra i due valori rilevati. In questo modo si ottiene la fase desiderata, ma si va incontro ad una serie di problemi; primo tra i quali che lo sfasamento si può ottenere con una accuratezza limitata dalla "grana" con cui è nota la forma d'onda, ossia dipendente dalla frequenza di campionamento usata. Inoltre, essa sarebbe nota con accuratezza proporzionalmente peggiore in funzione della frequenza utilizzata per il segnale sinusoidale di test; i punti acquisiti per secondo restano gli stessi, mentre la frequenza del segnale di test aumenta/diminuisce.

Per fissare le idee, si consideri un segnale sinusoidale di test a 1000 Hz campionato a 40960 Hz e con un buffer di acquisizione punti pari a 4096. Per esso si avrebbero 40960 punti al secondo, e dunque, per la frequenza considerata 40960/1000 = 40.96 punti per ciclo. Considerando 360 gradi a ciclo possiamo dire che 360/40.96 = 8.78 gradi è la distanza tra due punti consecutivi. Dunque, la fase relativa tra due segnali sinusoidali può essere nota con una risoluzione di 8.78 gradi (e dunque con incertezza non trascurabile).

Immaginiamo ora di voler calcolare la capacità di un condensatore tramite il nostro strumento, e avendo determinato il modulo dell'impedenza e la fase tra la tensione e la corrente che scorre nel bipolo, abbiamo:

$$C = \frac{1}{2\pi(Im)} \tag{8.3}$$

dove $Im = Z_x \cdot Sin(\varphi)$ e φ è lo sfasamento che abbiamo calcolato con la risoluzione di 8.78 gradi, e Z_x l'impedenza calcolata con la 8.1. Si osservi che la non idealità del componente capacitivo, comporta che la fase può non essere di 90 gradi esatti, ed inoltre nelle misure pratiche il circuito sotto misura può non essere necessariamente un componente solo, e dunque in generale presentare uno sfasamento anche sensibilmente differente da 90 gradi.

Supponiamo dunque di aver rilevato i seguenti valori:

$$Z_x = 100\ ohm;\ \varphi = 8\ gradi \tag{8.4}$$

Il che ci porta a ottenere immediatamente $C = 0.00159\ F$; nel caso peggiore la fase presenta un'incertezza che può arrivare a 8.78 gradi e dunque 89 - 8.78 = 80.22, ossia la capacità risulta C = 0.00161 F ossia approssimativamente un 2% di differenza. Si consideri che, a rigore, l'incertezza associata ad una tale misurazione verrebbe caratterizzata con una distribuzione uniforme e dunque con una deviazione standard e conseguente incertezza sensibilmente inferiore al 2% stimato.

Si consideri invece di lasciare invariati i parametri sinora considerati, eccetto la frequenza del segnale di test che porremo pari a 15 kHz. In tal caso avremo 40960/15000 = 2.73 punti per ciclo e dunque 360/2.73 = 131.86 è la distanza tra due punti consecutivi in gradi. Ossia, l'incertezza associata a questa (disastrosa) risoluzione deve fare i conti con una indeterminazione della fase che può arrivare a 131 gradi! Risulta evidente come non sia possibile utilizzare questo metodo, almeno per determinate frequenze del segnale di test. Queste considerazioni verranno utilizzate anche in altre sezioni.

8.3 Il metodo LMS di Widrow-Hoff

In letteratura e in rete è possibile reperire strumenti virtuali realizzati secondo un metodo ben consolidato, cui facciamo un rapido accenno in questo paragrafo e nei

successivi sottoparagrafi. Per passare direttamente alla descrizione del metodo proposto ed usato per lo strumento ZRLC si salti al paragrafo 8.4. Si osservi che, prima di definire le linee principali del metodo proposto in questo lavoro, abbiamo preferito riprodurre i risultati [42] descritti in questo paragrafo (ossia abbiamo scritto un programma funzionante) al fine di valutare le prestazioni ottenibili. Successivamente abbiamo esplorato altre strade, e ci siamo accorti che si potevano ottenere risultati nettamente migliori rispetto a LMS; e dunque abbiamo preferito abbandonare quest'ultimo. LMS è acronimo di "Least Mean Square" ossia "metodo dei minimi quadrati" ed è un metodo genericamente utilizzato per ridurre l'errore in un processo di calcolo minimizzando la somma dei quadrati degli errori. Nella sua versione detta di Widrow-Hoff [43] essa prevede l'uso di un algoritmo ricorsivo utilizzato per minimizzare la somma dei quadrati dell'errore e dunque giungere al calcolo delle quantità desiderate a meno di una certa quantità (in teoria) arbitrariamente piccola. Vedremo nel prossimo paragrafo come alcuni autori propongono di usarlo per la misura dell'impedenza, nella sua forma "completa" a ponte (8.3.1) e nella forma "ridotta" (8.3.2).

Si osservi che nel testo di Widrow e Stearns [43], oramai divenuto un classico, si da una descrizione esaustiva di meccanismi che partono dal seguente schema (tratto liberamente da [43]):

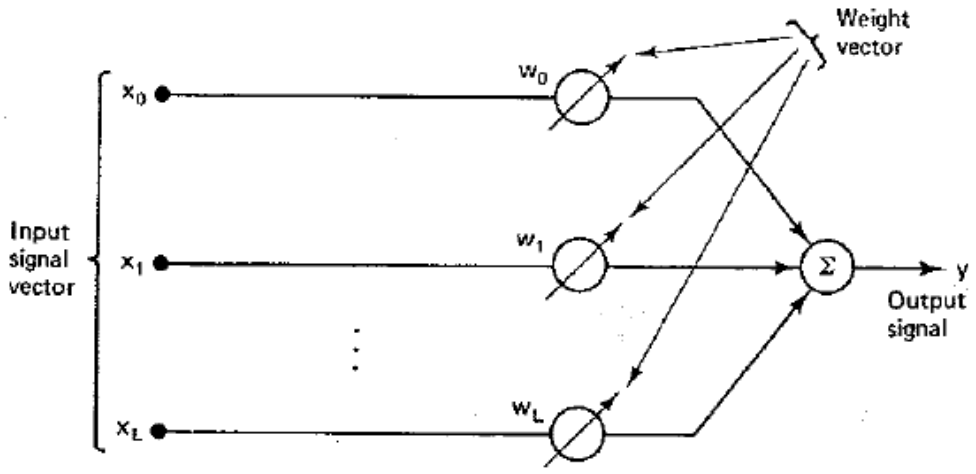


Figura 59: adaptive linear combiner "parallelo"

Che è relativo ad un “Adaptive Linear Combiner” in cui gli “L” campioni d’ingresso sono considerati temporalmente generati al medesimo istante, e vengono linearmente combinati, a mezzo dei coefficienti “*w*” per ottenere l’uscita *Y*. Gli stessi, possono indifferentemente essere considerati come generati ad istanti successivi (ad esempio ottenuti da un campionamento) e dunque essere combinati secondo quest’altro schema:

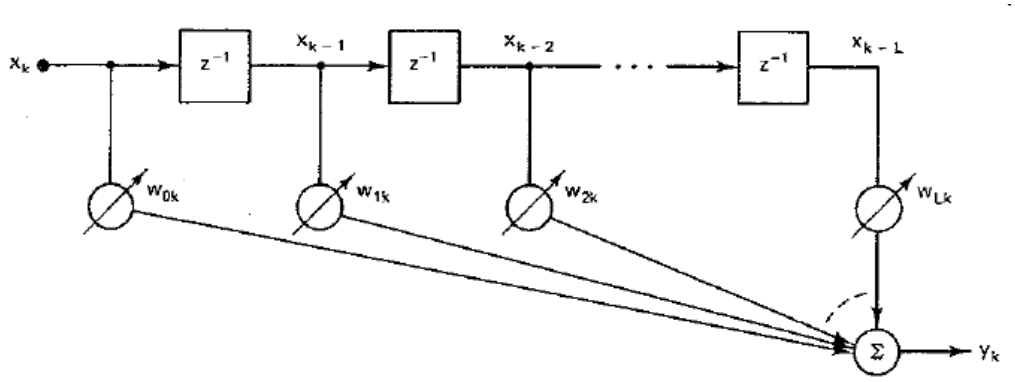


Figura 60: adaptive linear combiner "temporale"

Successivamente si può ricavare un segnale d’errore per confronto con un segnale di cui si vuole seguire l’andamento, ossia effettuare un’operazione descritta da questa figura, per semplicità facendo riferimento al primo dei modelli usati:

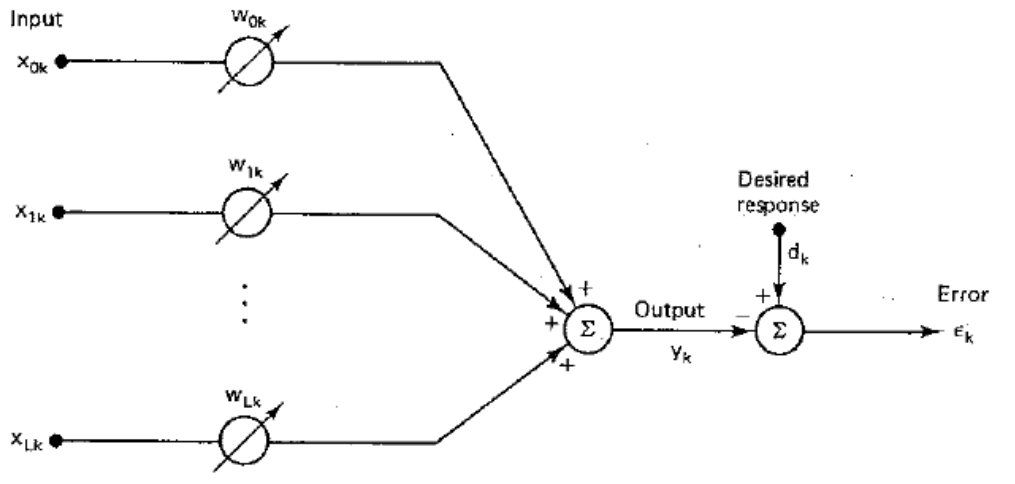


Figura 61: adaptive linear combiner che "insegue" il segnale desiderato

Tramite il segnale d'errore e_k ed opportuni algoritmi (LMS) saremo dunque in grado di "aggiustare" i coefficienti W al fine di ottenere un segnale Y_k che sia uguale al "Desired response".

Si osservi tuttavia quanto ironicamente riportato nel testo di Widrow:

"*We note, however, that considerably ingenuity is often required to find a suitable signal, since if the actual desired response were available one would generally not need the adaptive system*".

Il che è genericamente vero, ma come vedremo nei prossimi paragrafi, ricaveremo si un segnale identico a quello "desiderato" che già abbiamo, ma scomposto nelle sue componenti costitutive, direttamente legate alla parte reale e immaginaria dell'impedenza. In altre parole, i coefficienti W rappresenteranno proprio la parte reale e immaginaria della grandezza incognita che vogliamo misurare, sebbene il segnale complessivo che ci consentono di calcolare sia proprio il segnale di riferimento gia noto.

8.3.1 Metodo completo "Automatic AC Bridge"

Esso è illustrato dettagliatamente in [44], riportiamo qui le linee essenziali relative a sistemi tempo discreto, evidentemente di nostro maggiore interesse. In particolare gli autori implementano quello definito genericamente come "Automatic AC Bridge" per la misurazione di impedenze. Il lavoro testè accennato è stato poi fonte di ispirazione per una discreta quantità di lavori "pratici" che hanno avuto come risultato la produzione di programmi per DSP e più recentemente per PC [42], oggetto del prossimo paragrafo. Riportiamo la figura del circuito proposto dagli autori (fig. 62) e notiamo come esso sia essenzialmente un ponte che può essere bilanciato variando la tensione del ramo facente riferimento alla tensione V_x. L'impedenza incognita è Z_x, Z_r è un'impedenza di valore noto (di riferimento) ed anche la tensione Vr è di valore noto. La tensione V_x può essere espressa come somma di una componente in fase ed in quadratura, ossia:

$$V_x = V_{xi} + V_{xq} = m_i V_r + m_q V_q \tag{8.5}$$

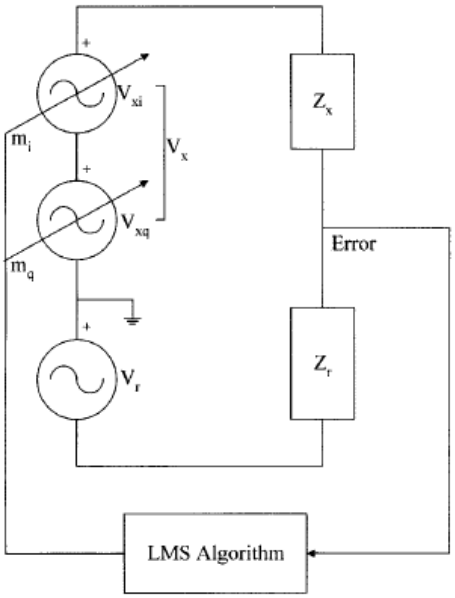


Figura 62: Ponte automatico in AC per la misura di impedenze

L'algoritmo proposto dagli autori parte dall'assunto di controllare il ponte (bilanciamento) agendo sulla tensione V_x, o meglio variando la componente in fase e in quadratura, tramite l'algoritmo LMS di Widrow-Hoff e la rilevazione dell'errore ("Error" in figura). E' facile dimostrare che al bilanciamento del ponte, l'impedenza incognita si può esprimere nella forma:

$$Z_x = m_i R + J m_q R \tag{8.6}$$

Avendo posto $Z_r = R$, ossia utilizzando come resistenza di riferimento una impedenza puramente resistiva. Questa scelta che implica l'importante conseguenza che il calcolo dell'impedenza incognita non involva calcoli con numeri complessi. Gli autori propongono la fig. 63 nella versione tempo continuo dell'algoritmo, e la corrispondente versione tempo discreto, rappresentata in fig. 64.

Si osservi che il riferimento (reference) indica come V_r e V_q rispettivamente la componente in fase ed in quadratura della tensione che nella figura 62 erano invece indicati come V_{xi} e V_{xq}; analogamente in figura 62 la tensione nota è indicata come V_r

mentre nelle altre figure indicata come V_p. Consideriamo questo fatto come una svista.

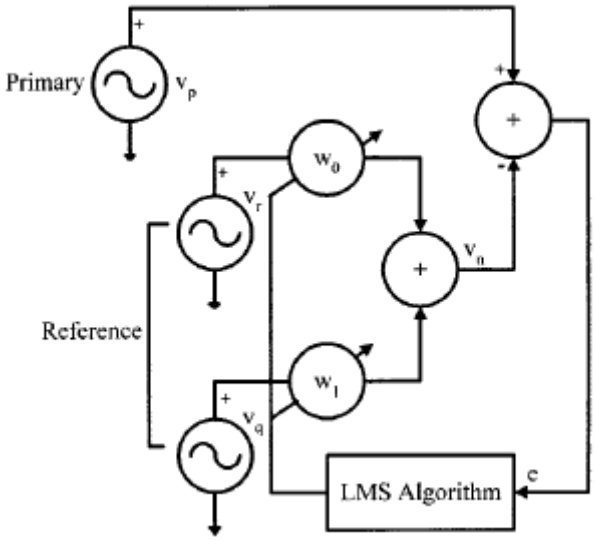


Figura 63: algoritmo tempo continuo

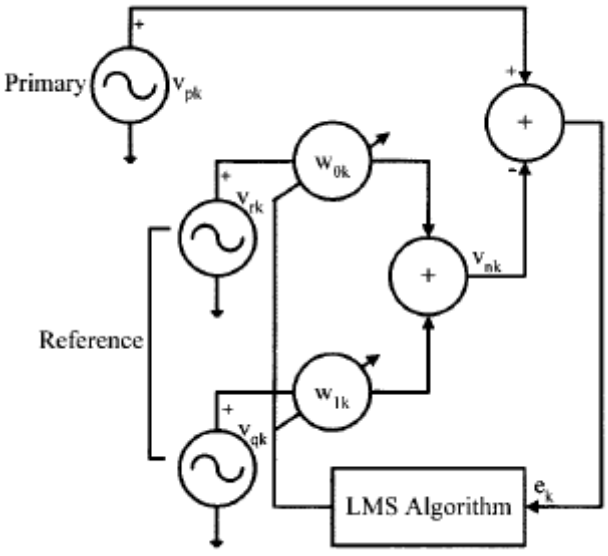


Figura 64: algoritmo tempo discreto

Il calcolo di m_i ed m_q (indicati da ora in poi come w_0, w_1, giacchè ad essi legati da relazioni di semplice proporzionalità) nel caso tempo discreto è realizzato tramite un

algoritmo “a discesa del gradiente” (steepest discendent algorithm); in particolare, definiti i vettori:

$$\mathbf{W} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} v_r \\ v_q \end{bmatrix} \tag{8.7}$$

Si applica l’algoritmo ricorsivo:

$$\mathbf{W}_{k+1} = \mathbf{W}_k + 2\mu e_k \mathbf{R}_k \quad k = 0, 1, 2 \ldots \tag{8.8}$$

Dove:

e_k = errore al k-simo istante;

W_{k+1} = valore del “peso” all’istante k+1

W_k = valore del “peso” all’istante k

μ = fattore di convergenza

scrivendo la precedente per il k-simo istante, ed applicando l’algoritmo di integrazione di Eulero, possiamo infine scrivere per ogni campione:

$$w_{i(k+1)} = 2\mu e_k r_{ik} + w_{ik} \tag{8.9}$$

In sintesi, l’algoritmo prevede che venga assegnato il fattore di convergenza μ; vengano lette e generate le due tensioni V_r e V_x; si applica la 8.9 iterando su tutti i punti disponibili sino ad ottenere un valore “stabile” del vettore $\boldsymbol{W}$ e dunque si inseriscono i valori ottenuti dalla 8.9 nella 8.6 e si ricavano tutti i parametri dell’impedenza (modulo fase, etc.). Si osservi dunque come sia necessario generare e leggere tre tensioni, oppure due sole, ed in tal caso di una è necessario effettuare la scomposizione in fase e in quadratura con qualche algoritmo.

Alcuni autori propongono invece il metodo semplificato che viene descritto nel prossimo paragrafo.

8.3.2 “Automatic AC bridge” in formato semplificato

Alcuni autori propongono la versione semplificata del “ponte” descritto nel paragrafo precedente, riportata in fig. 65 sotto forma di circuito completo, che essenzialmente ricalca quello “di principio” riportato in 8.2 e che costituisce la base del circuito usato, come vedremo, per realizzare ZRLC.

Gli amplificatori operazionali indicati in figura hanno la sola funzione di elevare il più possibile l’impedenza d’ingresso e dunque perturbare il meno possibile l’impedenza sotto misura (DUT).

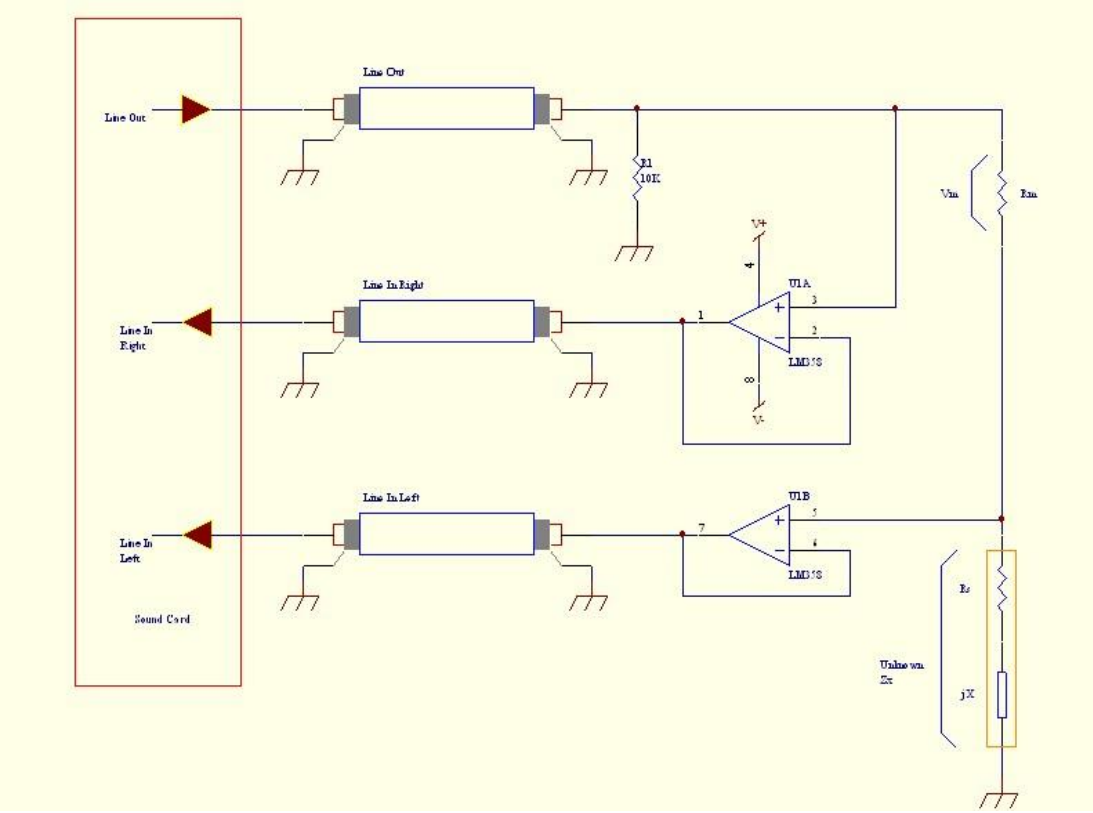


Figura 65: ponte semplificato

In particolare, viene generata una sola tensione applicata alla serie formata da R di riferimento e Z_X incognita; vengono poi lette le tensioni applicate ai capi di R e Z_X; infine il ponte viene bilanciato usando i dati letti in un ciclo di acquisizione (es. un buffer di VA) e ottenendo l’uguaglianza della tensione ai capi di R (letta) con quella

ai capi di Z_x (determinata algoritmicamente). Il vettore $\boldsymbol{W}$ ricavato consente di ricavare l'impedenza incognita tramite la 8.6

Si osservi che a rigore la tensione ai capi della resistenza di riferimento è ricavata come differenza delle due tensioni lette; e che dalla tensione generata e applicata ai capi del partitore viene ricavata la componente in fase ed in quadratura tramite il seguente schema (fig. 66):

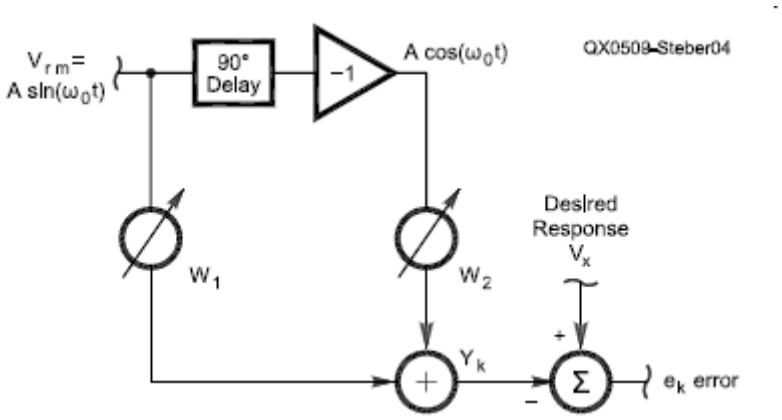


Figura 66: componenti in fase e quadratura

E dunque la componente in fase ed in quadratura vengono generate a partire da una sinusoide unica e applicando un ritardo "grezzo" di 90 gradi, ottenuto semplicemente ritardando la sinusoide generata di un numero discreto di campioni, ossia con un "passo" dipendente dalla frequenza di campionamento usata.

8.3.3 Confronto

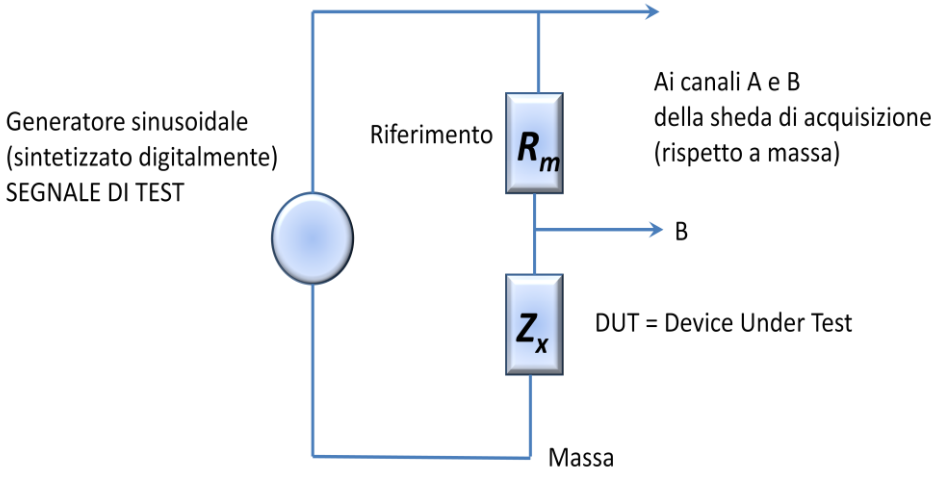
Il metodo proposto in 8.3.1 prevede dunque la generazione di tre tensioni, di cui due servono a generare una tensione su cui è possibile "agire" per tramite dei parametri del vettore $\boldsymbol{W}$ ossia per determinare i coefficienti "m" e dunque l'impedenza incognita. Inoltre si devono leggere almeno due tensioni per ricavare il segnale d'errore. L'algoritmo agendo sui parametri $\boldsymbol{W}$ riduce l'errore e_k al di sotto di una soglia prefissata. Le criticità intrinseche di questo metodo sono la generazione delle

tensioni il cui errore in fase e/o ampiezza è determinante al fine dell'accuratezza complessiva.

Il metodo proposto in 8.3.2 è più semplice in quanto prevede solo la lettura di due tensioni e la generazione di una; dalla quale però viene generata la coppia "fase/quadratura" in una maniera intrinsecamente imprecisa (vedi 8.2.1). Inoltre il bilanciamento è puramente virtuale, ossia non si agisce fisicamente su una tensione che va effettivamente a bilanciare un ponte, ma si ricava quella tensione che "bilancerebbe" il ramo del ponte cercando di calcolare il vettore *W* che "determinerebbe" l'uguaglianza tra la tensione ai capi dell'impedenza incognita e della resistenza di riferimento.

8.4 Il metodo proposto

Il metodo proposto prevede come circuito aggiuntivo alla scheda di acquisizione quello riportato nel par. 8.2 qui ripetuto per comodità, insieme alla relazione "base" 8.1 usata per la determinazione del modulo dell'impedenza incognita.



$$|Z_x| = \frac{|V_{zx}|}{|V_{rm}|} R_m \qquad (8.1)$$

Che eventualmente può essere trasformato in quello riportato nel paragrafo 8.3.2 fig. 65 in caso di schede di acquisizione con impedenza d'ingresso non sufficientemente elevata.

L'idea alla base di ZRLC è quella di usare la trasformata di Fourier, implementata tramite l'algoritmo "veloce" FFT; essa e stata già usata altrove nel programma per implementare altri strumenti virtuali (analizzatore di spettro e frequenzimetro). Il segnale di test è un segnale sinusoidale puro, a frequenza arbitraria (nei limiti delle capacità dell'hardware impiegato) e generato via software tramite lo strumento virtuale di VA "generatore di funzioni". Il fatto di essere generato dalla stessa scheda usata per l'acquisizione dei campioni comporta dei vantaggi non trascurabili che illustreremo a breve.

Tramite i due canali di acquisizione A e B vengono lette le sequenze di campioni rappresentative delle due tensioni ai nodi A e B rispetto al terminale indicato come massa (e come ovunque nel programma a gruppi di "n" punti, dove gli "n" punti sono necessariamente una potenza di due); successivamente da esse vengono ricavate le tensioni ai capi dei due bipoli, ossia le due sequenze di esse rappresentative. Di queste sequenze ne viene ricavata la trasformata di Fourier per tramite della FFT dalla quale si ricavano le ampiezze e la fase relativa. Si osservi che, come più volte riportato nel presente capitolo, la tensione ai capi della resistenza di riferimento è proporzionale, per tramite del valore della resistenza stessa, alla corrente che scorre nell'impedenza incognita. Dunque la fase tra le due tensioni rappresenta la fase tra la tensione e la corrente applicata all'impedenza incognita. La parte reale ed immaginaria dell'impedenza potranno dunque essere calcolate per tramite delle ben note relazioni:

$$Re = |Z|cos(\varphi)$$

$$Im = |Z|sin(\varphi) \tag{8.2}$$

Dove φ è calcolata tramite FFT e $|Z|$ per tramite della relazione 8.1 i cui valori di tensione sono calcolati dalle ampiezze fornite dalla FFT.

I problemi cui si va incontro, e che peggiorano l'accuratezza di misura aumentando l'incertezza, sono:

1. FFT presenta i noti problemi, descritti nel capitolo precedente, di aliasing, dispersione spettrale, interferenza armonica (par. 8.4.1);
2. i due canali di acquisizione della scheda possono non essere identici e dunque introdurre un errore relativo (tra canali) di guadagno e di fase (par. 8.4.2);
3. in molte schede di acquisizione viene usato un solo convertitore A/D multiplexato e che dunque introduce un errore sistematico di fase pari ad uno o più passi di campionamento (par. 8.4.2);
4. l'uso di un generatore ad una determinata frequenza pone dei problemi di sincronizzazione con la FFT (vedi avanti) che peggiorano i fenomeni accennati in (1) (par. 8.4.1).

8.4.1 Aliasing, dispersione, interferenza (1)(4)

I problemi riportati in (1) sono stati ampiamente descritti nel capitolo precedente; ad essi si aggiunga che, sino a non molto tempo fa, la potenza di calcolo disponibile era tale che anche utilizzando un algoritmo intrinsecamente veloce come la FFT essa provocava comunque un discreto "assorbimento" di potenza di calcolo; gli errori dovuti alle tre cause di (1) oltre a (2) (3) e (4) scoraggiava l'uso di FFT e parimenti vedeva il proliferare di metodi "alternativi" come quello di LMS che evitava una consistente mole di calcoli, sebbene non risolvesse i problemi di (2) e (3).

Le risorse di calcolo, con i moderni sistemi che hanno, in molti casi, duo o più microprocessori e molte operazioni matematiche definite direttamente in hardware (es. funzioni trigonometriche principali), rendono questo problema automaticamente superato, cosi come in generale è successo per tutti gli algoritmi dell'Elaborazione Numerica divenuti presto appannaggio di normali PC (cfr. cap. I,III). A titolo di esempio si consideri che con una frequenza di campionamento "canonica" di 44100 Hz ed un buffer di 4096 punti, si ottiene un buffer di campioni circa ogni 100 mS, riuscendo tuttavia a calcolare una FFT in meno di 5 mS.

Il problema dell'aliasing può ovviamente risolversi con un adeguato filtro (analogico) anti-aliasing, normalmente presente in un sistema di acquisizione pre-confezionato. Invero, come evidenziato anche nel capitolo precedente, la DFT (o FFT) risulta completamente esente da aliasing solo nel caso in cui il segnale sia completamente periodico; condizione nel nostro caso senza dubbio verificata perché usiamo un segnale di test monofrequenziale e rigorosamente periodico. Ricordiamo che, qualora il segnale non fosse periodico, rimarrebbe comunque una componente di aliasing che il filtro antialiasing certamente eliminerebbe, ma modificando la forma d'onda (ossia riducendone il contenuto armonico) e dunque introducendo comunque una alterazione del segnale. Vediamo ora come è possibile migliorare, con semplici accorgimenti, il processo di eliminazione dell'aliasing.

Nel caso della scheda che presenteremo nel prossimo capitolo, è teoricamente possibile usare frequenze di campionamento sino a 48 kHz (e dunque con banda passante sino a 24kHz) con un filtro antialiasing dimensionato di conseguenza e già compreso nei circuiti integrati adoperati. Da prove pratiche ci si è accorti che il convertitore A/D interno riesce ad essere pilotato sino ad oltre 80 kHz di frequenza di campionamento; purtuttavia il filtro antialiasing rimane "inchiodato" a poco più di 20 kHz, come è logico aspettarsi. In tal guisa, potrebbe sembrare inutile forzare il circuito a funzionare a frequenze di campionamento così elevate se poi la risposta in frequenza complessiva rimane la stessa. In realtà, si ottiene comunque un grosso vantaggio, e cioè una drastica riduzione dell'aliasing residuo, sempre presente specialmente se il circuito utilizzato è di basso costo.

In altri termini, è come se lavorassimo in "oversampling" e dunque spostando ben al di la della regione "critica" del filtro la frequenza di Nyquist, e quindi assicurandoci che ogni componente armonica non consentita sia effettivamente eliminata. La nostra scelta, per lo strumento presentato nel prossimo capitolo, è di usare una frequenza di campionamento di 81920 Hz e un buffer di 8192 punti, ed un filtro anti-aliasing a circa 20 kHz.

La dispersione spettrale, anch'essa esaustivamente descritta nel capitolo precedente, è pure ridotta ai minimi termini tramite delle semplici scelte. Infatti, essa si presenta

se non operiamo un campionamento sincrono con la trasformata di Fourier; cosa che in generale non può avvenire con un segnale arbitrario, anche se periodico. Se infatti il segnale che vogliamo visualizzare con un analizzatore di spettro non è a priori noto, non possiamo adattare la frequenza di campionamento in maniera da ottenere la voluta sincronicità; anche se esso fosse, pur se arbitrario, perfettamente noto, sorgerebbero comunque delle difficoltà, perché generalmente non è possibile imporre una frequenza di campionamento qualsiasi ma normalmente a "passi" predefiniti. Ad esempio nei sistemi Windows è possibile gestire le schede di acquisizione (anche quelle sonore) impostando un valore intero della frequenza di campionamento. Se dunque il segnale sotto misura ha una frequenza qualsiasi, ossia il cui valore è genericamente un numero reale (ossia un numero decimale) anche conoscendone esattamente la frequenza non potremo adattare precisamente la frequenza di campionamento per raggiungere la condizione appunto nota come "*campionamento sincrono*". Sebbene, a rigore, in questo caso saremmo in grado di calcolare con esattezza l'incertezza introdotta.

La soluzione adottata è oltremodo semplice; essa prevede l'uso di frequenze di test, sinusoidali, che sono un multiplo della risoluzione della FFT. Così, se usiamo una frequenza di campionamento di 81920 Hz ed un buffer di 8192 punti, abbiamo una risoluzione di 81920/8192 = 10Hz. Imponendo di poter usare come frequenze di test una frequenza multipla di 10, ossia = *n*10* dove *n* intero maggiore di zero, otteniamo comunque un range di frequenze utilizzabile pressocchè arbitrario e un campionamento decisamente sincrono.

Si evita dunque l'uso di "finestraggi" di ogni tipo e problemi ad essi relativi. Inoltre, se si usa, come abbiamo fatto nella scheda proposta, generare il segnale di test via software e con un convertitore D/A pilotato dallo stesso segnale di clock che pilota il convertitore A/D si ottiene una ulteriore drastica riduzione della dispersione spettrale.

Da prove sperimentali infatti risulta che, anche lavorando in condizioni di campionamento sincrono, ma usando segnali generati da un hardware distinto da quello usato per la lettura, permane una certa quantità di dispersione spettrale causata

dalle inevitabili e impercettibili differenze tra la frequenza del segnale generato e la frequenza “attesa” dalla FFT.

In altre parole, se il segnale necessario per ottenere il campionamento sincrono è (poniamo) di 1000 Hz perché usiamo la relazione *n*10* con *n=100*, un suo eventuale valore anche leggermente differente di pochi decimi o persino millesimi di Hz sarebbe automaticamente compensata da una deriva nello stesso senso del convertitore A/D se usati sinergicamente (ossia stesso clock) altrimenti lo spettro risultante, teoricamente una linea verticale “stretta”, finisce per essere un lobo simile alla trasformata di un impulso e sull’oscilloscopio si nota un evidente “scorrimento” della sinusoide.

Queste considerazioni sono derivate da osservazioni puramente sperimentali, che riportiamo nelle figure che seguono; sia nella prima che nella seconda figura usiamo i medesimi valori (81920 Hz di campionamento, 8192 punti di buffer, 1000 Hz di frequenza di segnale di test) ma nella prima usando convertitori A/D e D/A pilotati dallo stesso clock, e nella seconda da clock diversi (seppur nominalmente uguali).

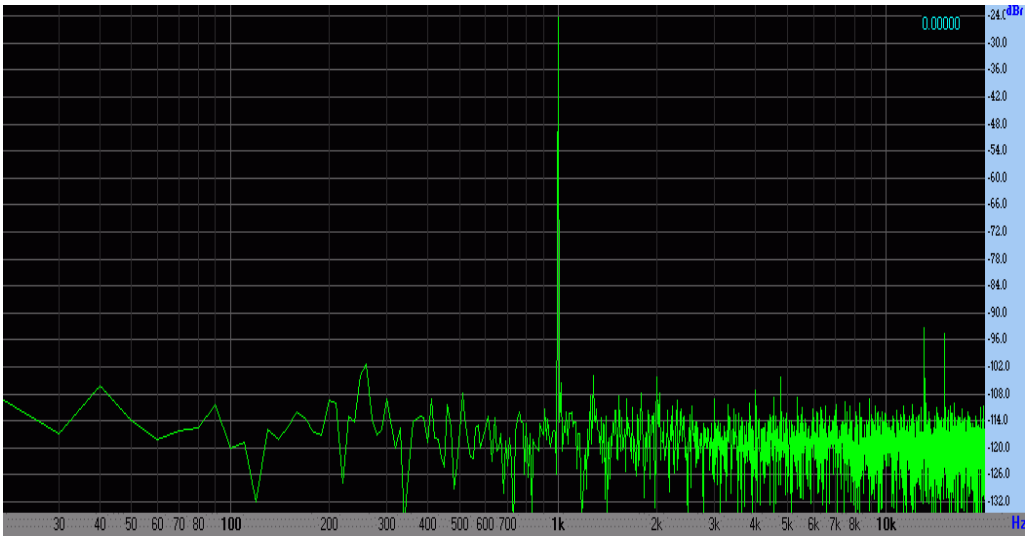


Figura 67: spettro con stesso clock

Nella prima si osserva che, pur non usando alcuna finestratura, la linea spettrale è ben definita, ed il resto del segnale è semplice rumore, e inoltre il segnale sull’oscilloscopio e perfettamente “triggerato”; nel secondo di osserva una evidente

dispersione spettrale residua ed il segnale sull'oscilloscopio "deriva" anche se molto lentamente (i segnali dell'oscilloscopio non sono riportati).

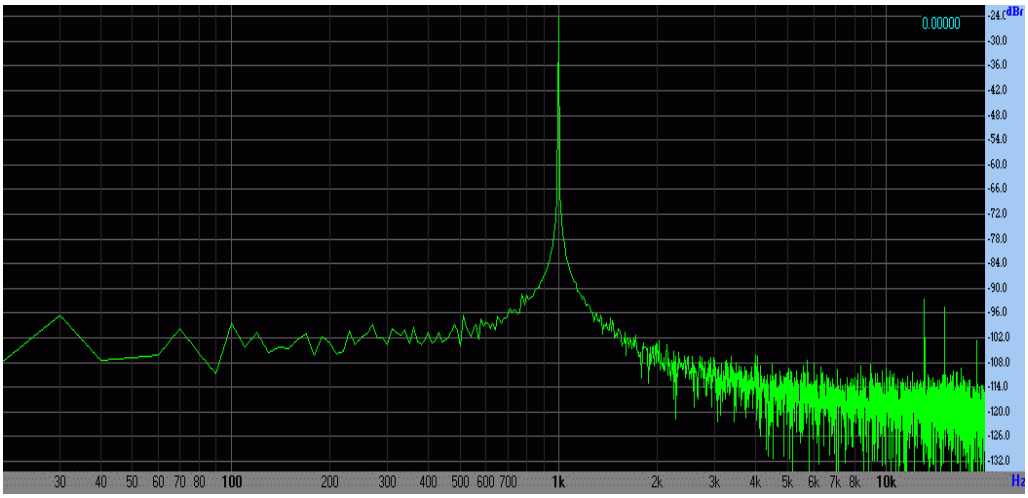


Figura 68: spettro con diverso clock

L'interferenza armonica è in questo caso un falso problema; nel nostro strumento non avremo mai l'esigenza di generare frequenze eccessivamente vicine e dunque non andremo mai incontro al fenomeno; si osservi comunque che, sebbene non strettamente necessario, abbiamo provveduto a filtrare il segnale, prima di essere utilizzato nei vari algoritmi, con un filtro di "notch inverso" ad elevatissima pendenza. Esso si adatta automaticamente ogni qual volta si seleziona una frequenza di test nella listbox corrispondente.

8.4.2 Errore di guadagno e fase (2) (3)

Ogni qualvolta si utilizzano strutture hardware virtualmente identiche, si va incontro ad incertezze dovute alla non eguaglianza fisica delle medesime. In questo caso stiamo parlando essenzialmente dei circuiti di condizionamento dei due canali, giacchè in moltissimi casi (ed è per esempio il caso della scheda da noi presentata) il convertitore A/D usato è il medesimo per entrambi i canali, e dunque evidentemente identico.

Iniziamo dunque evidenziando una prima e macroscopica causa d'incertezza, dovuta ad un errore sistematico; se il convertitore utilizzato è unico, giocoforza i due canali saranno sfasati di una quantità temporalmente uguale ad un passo di campionamento; che si traduce in uno sfasamento in gradi il cui valore è tanto peggiore quanto più elevata è la frequenza generata/letta. In generazione è un problema che non ci riguarda, in quanto stiamo utilizzando un solo canale; diverso è il discorso in fase di acquisizione, giacché usiamo i due canali presupponendone la completa uguaglianza; che si traduce essenzialmente nel supporre la medesima risposta in frequenza (ampiezza e fase).

Semplicemente, è stata implementata una coda software come indicato in figura 69.

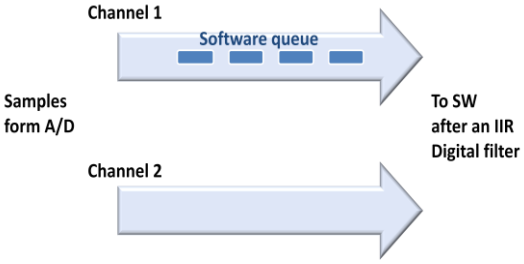


Figura 69:ritardo tramite coda

Che può realizzare un ritardo arbitrario di "n" campioni indifferentemente su un canale o l'altro; essa può essere impostata tramite la finestra generale di "Settings", nel "tab" principale, riportato in fig. 70:

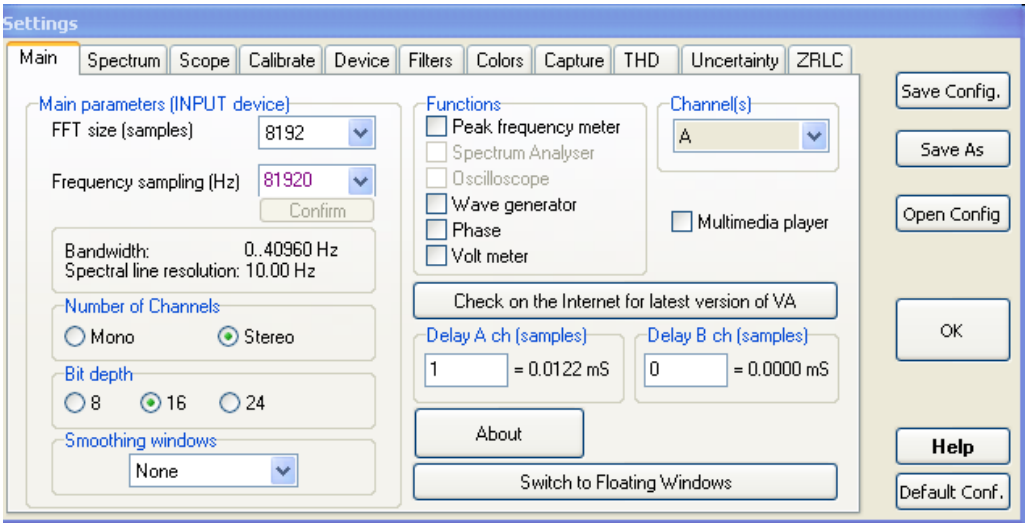


Figura 70: finestra di "Settings", "tab" principale

Con la coppia scelta di frequenza di campionamento e buffer, ed inserendo per esempio il ritardo sul canale A di un solo campione, esso equivale ad un ritardo temporale di 0.0122 mS.

In questo modo si compensa il "grosso" del ritardo di fase tra i due canali. Invero, prove sperimentali, hanno dimostrato che permane un ritardo (anche significativo) tra i due canali, dovuto ai circuiti di condizionamento, in questo caso semplici amplificatori/attenuatori e circuiti di protezione. La misura implica l'applicazione della relazione 8.1, ossia il calcolo delle due tensioni applicate ai capi della resistenza di riferimento e dell'impedenza da misurare; in particolare la prima grandezza è determinata come differenza tra le tensioni lette dai due canali, grandezza che quindi è sensibile alla differenza di fase tra le due, sia istante per istante che in modulo. Inoltre, la parte reale e immaginaria dell'impedenza è calcolata tramite le 9.2 e dunque usando lo sfasamento calcolato tra le due tensioni, anch'esso naturalmente influenzato (ed in maniera correlata) dallo sfasamento tra i canali.

Le prove sperimentali evidenziano uno sfasamento che può arrivare dai decimi di grado sino all'ordine del mezzo grado; ed inoltre proporzionalmente maggiore in funzione della frequenza del segnale di test.

La soluzione proposta per la correzione dell'errore sistematico "residuo" di fase tra i due canali è la seguente. Si osservi che la seguente nota relazione trigonometrica:

$$A\sin(\omega t + \varphi) = A\,(\cos(\varphi)\sin(\omega t) + \sin(\varphi)\cos(\omega t)) \qquad (9.3)$$

esprime un semplice fatto: un segnale sinusoidale arbitrario di ampiezza "*A*" può essere espresso come la somma di un segnale sinusoidale ed uno co-sinusoidale, ossia come la somma di una componente che diremo in "fase" ed una in "quadratura" evidenziando in tal modo uno sfasamento tra le due componenti di 90 gradi. La stessa relazione può dunque essere espressa per i segnali a tempo discreto, ossia quelli di nostro interesse, e riscriversi come:

$$v(k) = A\sin(\omega t_k + \varphi) = A\,(\cos(\varphi)\sin(\omega t_k) + \sin(\varphi)\cos(\omega t_k)) \qquad (8.4)$$

dove *v(k)* è il campione al generico istante k con k intero positivo, e t_k è il valore del tempo dipendente dalla frequenza di campionamento, ossia vale ovviamente la relazione:

$$t_k = \frac{1}{F_c} \cdot k \tag{8.5}$$

Dove F_c è la frequenza di campionamento in Hz, *t* il tempo in secondi e *k* intero positivo.

Nella 8.4 emerge un fatto importantissimo. La relazione esprime la dipendenza del campione k-simo dalla frequenza del segnale, dal valore di *k* e dallo sfasamento φ. Quest'ultimo è rappresentato da una variabile continua; dunque se di un generico segnale sinusoidale tempo discreto conosciamo la rappresentazione in funzione di seno e coseno così come espresso dalla 8.4 automaticamente potremo esprimere esso con uno sfasamento arbitrario applicando appunto da 8.4 ed imponendo il valore di φ desiderato.

Dunque, nel calcolo dell'impedenza, potremo correggere la fase su uno dei due canali semplicemente calcolando lo sfasamento residuo tra i due canali e esprimendo il segnale letto da uno dei due canali tramite l'applicazione della 8.4. In altre parole, un canale verrà utilizzato direttamente e l'altro "ricalcolato" tramite la 8.4.

A tal proposito si veda la fig. 71; come indicato in (1) si genera un segnale sinusoidale; si applica ai due canali e si calcola con un qualsiasi metodo lo sfasamento residuo (per esempio con una FFT). Successivamente, si può utilizzare l'angolo di sfasamento testè calcolato come indicato in (2) di fig. 71; si ricordi infatti che nella misura dell'impedenza vengono applicate delle sinusoidi ad entrambi i canali, ed il problema è applicare la correzione ad un solo canale.

Allora, immaginiamo che uno dei due canali legga una sinusoide (in (2) indicata come "Sin"); essa rappresenta la componente in "fase" da introdurre nella relazione 8.4; la componente in quadratura la otterremo sfasando di 90 gradi la stessa componente in fase, con un metodo qualsiasi (per esempio con un filtro "allpass" come indicato in figura; oppure altro, vedi resto del paragrafo) e dunque calcoleremo

il campione all'istante k-simo tramite la 9.4, inserendo i campioni in fase e quadratura calcolati, e lo sfasamento correttivo φ.

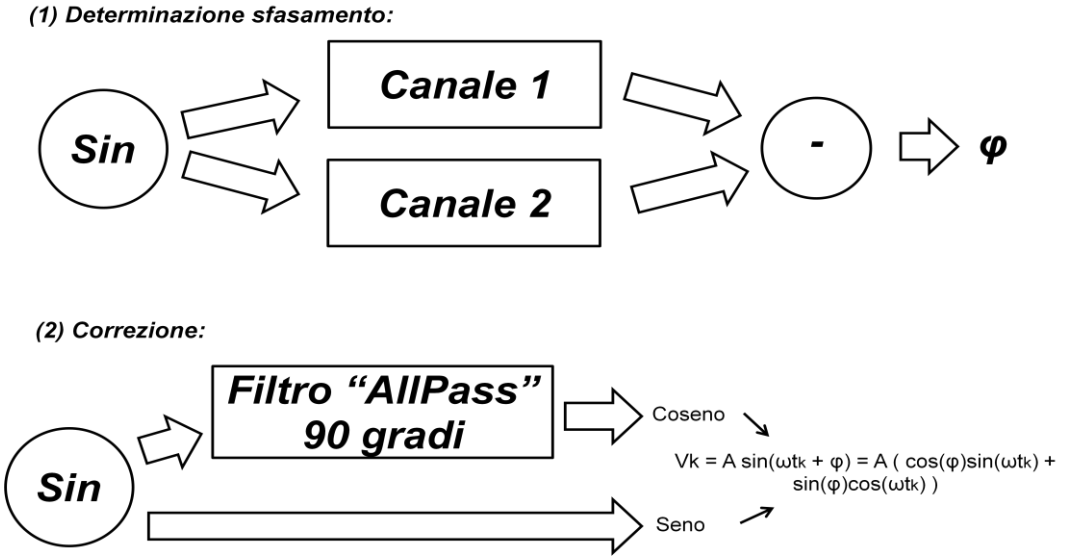


Figura 71: sfasamento, determinazione e correzione

A questo punto bisogna osservare che abbiamo semplicemente "spostato" il problema; ma apparentemente esso non è stato ancora risolto. Infatti, si pone ora il nuovo problema di calcolare esattamente un segnale sfasato di 90 gradi a partire da un segnale noto. In letteratura è possibile trovare svariati metodi per ottenere da un segnale sinusoidale una "coppia" seno-coseno; il più classico tra i quali è applicare una trasformata di Hilbert. La cui espressione più immediata potrebbe essere semplicemente utilizzare il metodo "grezzo", ossia calcolare un ritardo discreto ottenuto tramite una semplice coda, come effettuato in fig. 69. In questo caso otterremmo, come ampiamente discusso in altra sede, un errore inaccettabile, soprattutto se confrontato con l'ordine di grandezza in gioco dello sfasamento residuo. Si rende dunque necessaria l'applicazione di una trasformata di Hilbert completa, e non tramite un semplice ritardo.

Normalmente una trasformata di Hilbert viene approssimata tramite un filtro FIR, il cui numero di "tappi" (coefficienti) determina la maggiore o minore accuratezza del risultato ottenuto. Alcuni autori propongono l'uso di un filtro IIR, soluzione che

abbiamo preferito non utilizzare per motivi di stabilità. Ricordiamo a grandi linee cosa si intende per trasformata di Hilbert, rimandando a [45] per i dettagli. Dato un segnale $x(t)$ si dice Trasformata di Hilbert di $x(t)$ il segnale $\underline{x}(t) = H\{x(t)\}$ ottenuto dal transito attraverso la seguente funzione di trasferimento:

$$H_H(f) = \begin{cases} j & f < 0 \\ 0 & f = 0 \\ -j & f > 0 \end{cases} \tag{8.6}$$

Si dimostra che dato un segnale $x(t)$ reale ed a valor medio nullo, il segnale e la sua trasformata di Hilbert sono ortogonali; ossia:

$$\langle \underline{x}(t)\, x(t) \rangle = 0 \tag{8.7}$$

Il che significa che la trasformata di un segnale puramente cosinusoidale è ortogonale ad esso, ossia è una sinusoide, come è possibile osservare dalla figura 72:

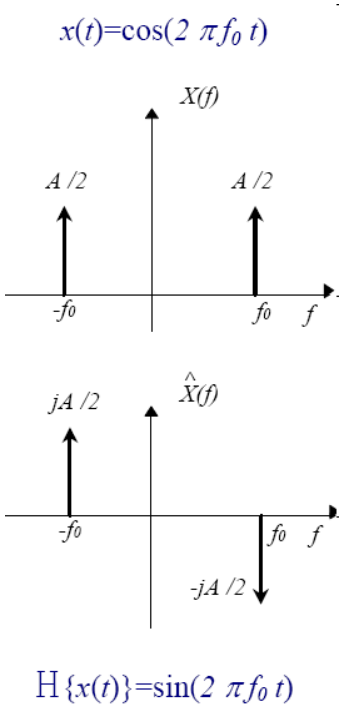


Figura 72: trasformata di Hilbert di un segnale co-sinusoidale

L'estensione al caso tempo discreto dei risultati di cui alle righe precedenti è immediata, e la omettiamo per semplicità. Sono state effettuate delle prove tramite l'approssimazione (che in letteratura è possibile trovare per esempio in [45]) della trasformata di Hilbert con filtri FIR a "n" coefficienti. In particolare test con 64 coefficienti, 128 e 256 non hanno fornito risultati soddisfacenti, ottenendo sfasamenti con margini che potevano essere dell'ordine del grado. Abbiamo dunque preferito adottare la soluzione già proposta nello schema a blocchi di figura 71 (2); ossia l'uso di un filtro "allpass", ossia "passa tutto". Come dice il nome esso ha una banda passante unitaria (piatta) in ampiezza e arbitraria (entro certi limiti) in fase. Quest'ultima caratteristica ci ha permesso di calcolare un filtro che presentasse una funzione di trasferimento che non alterasse l'ampiezza (modulo) del segnale in tutto il range di frequenze consentite dall'hardware e uno sfasamento identicamente uguale a 90 gradi (per ogni frequenza del segnale di test). In particolare abbiamo utilizzato i risultati di [31] che propone un filtro IIR, a sua volta ottenuto dal campionamento di un equivalente filtro analogico, tramite una trasformazione bilineare. La funzione di trasferimento del filtro, usando la trasformata "z" è (utilizzabile per filtri qualsiasi):

$$H(z) = \frac{b_0+b_1z^{-1}+b_2z^{-2}}{a_0+a_1z^{-1}+a_2z^{-2}} \tag{8.8}$$

Normalizzando rispetto ad a_0 otteniamo:

$$H(z) = \frac{\frac{b_0}{a_0}+\frac{b_1}{a_0}z^{-1}+\frac{b_2}{a_0}z^{-2}}{1+\frac{a_1}{a_0}z^{-1}+\frac{a_2}{a_0}z^{-2}} \tag{8.9}$$

E' dunque immediato ricavare la corrispondente classica equazione ricorsiva nel dominio del tempo:

$$y[n] =$$

$$\frac{b_0}{a_0}\cdot x[n] + \frac{b_1}{a_0}\cdot x[n-1] + \frac{b_2}{a_0}\cdot x[n-2] - \frac{a_1}{a_0}\cdot y[n-1] - \frac{a_2}{a_0}\cdot y[n-2] \tag{8.10}$$

Che risulta una relazione senza dubbio facile da implementare in un calcolatore. I coefficienti del filtro sono stati calcolati tramite le seguenti relazioni:

$$A = \sqrt{10^{\frac{dBgain}{20}}}$$

$$\omega_0 = \frac{2 \cdot \pi \cdot f_0}{F_s}$$

$$\alpha = \sin(\omega_0) \cdot \sinh\left(\frac{\ln(2)}{2} \cdot \Delta \cdot \frac{\omega_0}{\sin(\omega_0)}\right) \quad (8.11)$$

Dove Δ e ω_0 sono dei parametri che ci consentono la messa a punto della risposta in fase, consentendo di impostare uno sfasamento di 90 gradi "esatti" alla frequenza voluta. I coefficienti sono alla fine calcolati come:

$$b_0 = 1 - \alpha$$

$$b_1 = -2 \cdot \cos(\omega_0)$$

$$b_2 = 1 + \alpha$$

$$a_0 = 1 + \alpha$$

$$a_1 = -2 \cdot \cos(\omega_0)$$

$$a_2 = 1 - \alpha \quad (8.12)$$

Dai quali è immediato ottenere i corrispondenti normalizzati rispetto ad a_0. Una nota circa la metodologia utilizzata per "calibrare" il filtro ad ogni frequenza usata da ZRLC come segnale di test. Abbiamo usato un taglio eminentemente pratico; il parametro Δ è stato fissato ad un valore costante, e abbiamo variato ω_0 ("frequenza di taglio"), in maniera da ottenere uno sfasamento quanto più possibile vicino a 90 gradi per la frequenza del segnale di test (che è in generale diversa dalla frequenza di taglio ω_0). In particolare abbiamo aggiunto nel programma una procedura iterativa che, partendo dal valore della frequenza di taglio del passo precedente (al primo giro posta alla metà della frequenza del segnale da generare), incrementa ω_0 di una quantità predefinita arbitrariamente piccola (nei limiti della rappresentazione usata del numero reale, nel nostro caso IEEE floating point a 80 bit). In tal modo, una tantum ed in relazione alla frequenza di campionamento usata, abbiamo determinato i valori di ω_0 che consentono di ottenere, frequenza per frequenza, la migliore approssimazione possibile. Per fissare le idee, riportiamo una tabella comparativa

che riporta un estratto delle frequenze che è possibile usare come segnale di test (frequenza di campionamento 81920 Hz e 8192 punti di buffer) e il valore di sfasamento ottenuto sia con il filtro "allpass" calcolato come testé indicato e con il filtro FIR a 256 tappi tramite il quale abbiamo approssimato la trasformata di Hilbert. Si nota l'evidente superiorità della soluzione proposta, che dunque è stata adottata ed ha comportato semplicemente un pre-calcolo "una-tantum" dei coefficienti che sono stati "inglobati" nel codice eseguibile.

Frequenza(Hz)	*All pass*	*Hilbert*
500	90.002	91.232
1000	90.001	90.434
2000	90.011	90.621
4000	90.001	91.934
6000	90.004	90.259
8000	90.003	91.632
10000	90.002	91.316
12000	90.007	91.232
14000	90.002	90.429
16000	90.001	90.012
18000	90.006	90.223
19000	90.002	90.545

Tabella 2: comparazione tra soluzione con filtro allpass e trasformata di Hilbert

Concludiamo questo paragrafo dando un rapido accenno al metodo usato per bilanciare i due canali in relazione all'ampiezza relativa. In tal caso il procedimento adottato è stato vieppiù semplice e lineare; è stata "iniettata" la stessa sinusoide nei due canali, come già fatto per il calcolo dello sfasamento, e successivamente sono state calcolate le ampiezze di picco dei due segnali; è stata quindi calcolata l'ampiezza relativa e dunque ricavato un fattore correttivo da applicare ad uno dei due canali per correggere le eventuali differenze di amplificazione tra canali. In altre parole, una costante (compresa nell'intervallo *(0..1]*) per cui ogni campione di uno dei due canali deve essere moltiplicato.

8.5 Il calcolo dell'incertezza

Descriveremo in questo paragrafo come è stato effettuato il calcolo dell'incertezza composta per lo strumento ZRLC. Data la semplicità della relazione che è stata utilizzata per il calcolo dell'impedenza in modulo, sarebbe stato possibile implementare un calcolo dell'incertezza "white box" così come proposto nella stessa GUM [22], in particolare nella sezione "H" relativa agli esempi. Esempio ripreso anche in [32]. Tuttavia, si deve notare che per il calcolo della fase è stata utilizzata l'FFT così come per il calcolo delle ampiezze delle due tensioni. Anche per essa invero è possibile un calcolo "white-box" dell'incertezza cosi come proposto nel capitolo VI. Dunque sarebbe stato possibile calcolare l'incertezza complessiva tramite un approccio completamente "white box". Data la natura "real-time" del programma, abbiamo preferito utilizzare un approccio "misto" che prevede un calcolo di incertezza di tipo "A" in tempo reale, più l'aggiunta dell'incertezza dovuta alla resistenza di riferimento e delle componenti "sistematiche" residue che non possono essere comprese nel calcolo dell'incertezza di tipo A. Quest'ultima infatti calcola effettivamente solo l'incertezza relativa ad errori casuali, e nulla può dire in relazione ad errori sistematici che semplicemente "spostano" in maniera non casuale il valore atteso delle misurazioni effettuate. Peraltro, l'eliminazione degli errori sistematici più significativi è stata effettuata (vedi paragrafi precedenti) e quelli

residui (che possono ridursi ad errore di gain e offset “residuo”) possono essere schematizzati con opportune distribuzioni ed aggiunti al calcolo finale come incertezze di tipo “B”. In particolare, gain e offset, come detto generalmente da considerare sistematici (anche se non necessariamente), sono stati desunti dalle caratteristiche fornite dal costruttore relativamente al convertitore A/D e D/A. Si osservi altresì che, come riportato anche da altre fonti [36], molti convertitori commerciali presentano la possibilità di compensare via hardware questi errori. In ogni caso, si tenga presente che il programma consente di impostare tutti i valori relativi alle varie cause d’incertezza che possono essere imputati all’hardware di acquisizione (cfr. cap. IV) e dunque in possesso di questi effettuare il calcolo corretto. Si veda a tal proposito la figura seguente (73).

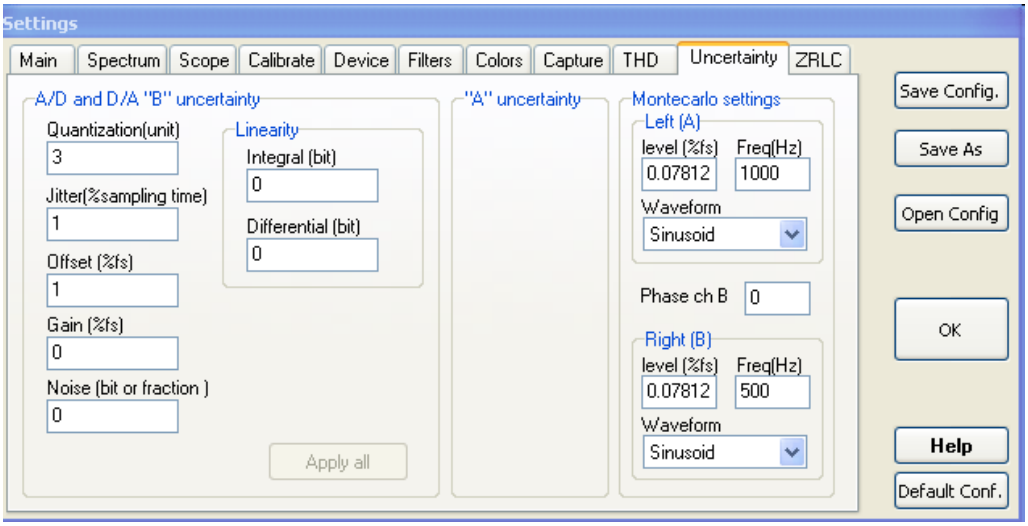


Figura 73: la finestra di impostazione delle incertezze

Inoltre, in generale Visual Analyser consente il calcolo dell’incertezza di tipo A per ogni strumento implementato; in alcuni casi poi, come quello di ZRLC, essa è stata specializzata (ossia sviluppata una procedura specializzata per il calcolo dell’incertezza statistica appositamente scritta per ZRLC) e aggiunta ad altre valutazioni. La valutazione generale dell’incertezza di tipo A avviene tramite una funzione, invocabile per tutti gli strumenti, che consente di visualizzare in una finestra ed in tempo reale l’istogramma delle frequenze dei dati (e quindi di valutare approssimativamente la distribuzione ottenuta) ed il calcolo della media, varianza, deviazione standard e dunque dell’incertezza statistica associata alle misurazioni.

Essa si presenta come in figura 74. Si osservi che essa è resa possibile per il fatto che Visual Analyser e tutti gli strumenti da esso virtualizzati, effettuano la misura operando su un buffer di campioni acquisito sequenzialmente nel tempo (e su base virtualmente infinita); in altri termini vengono effettuate misure ripetute in ragione dei parametri scelti. Per esempio se si usa una frequenza di campionamento di 40960 Hz e un buffer di 4096 punti si effettua una misura ogni 100 mS (40960/4096) ossia 10 misure al secondo. E' possibile dunque effettuare un numero consistente di misure in pochi secondi e calcolare una buona approssimazione dello scarto tipo della media (oltre a quello del campione). La generica finestra è riportata nella seguente figura (test effettuato con SCOPE):

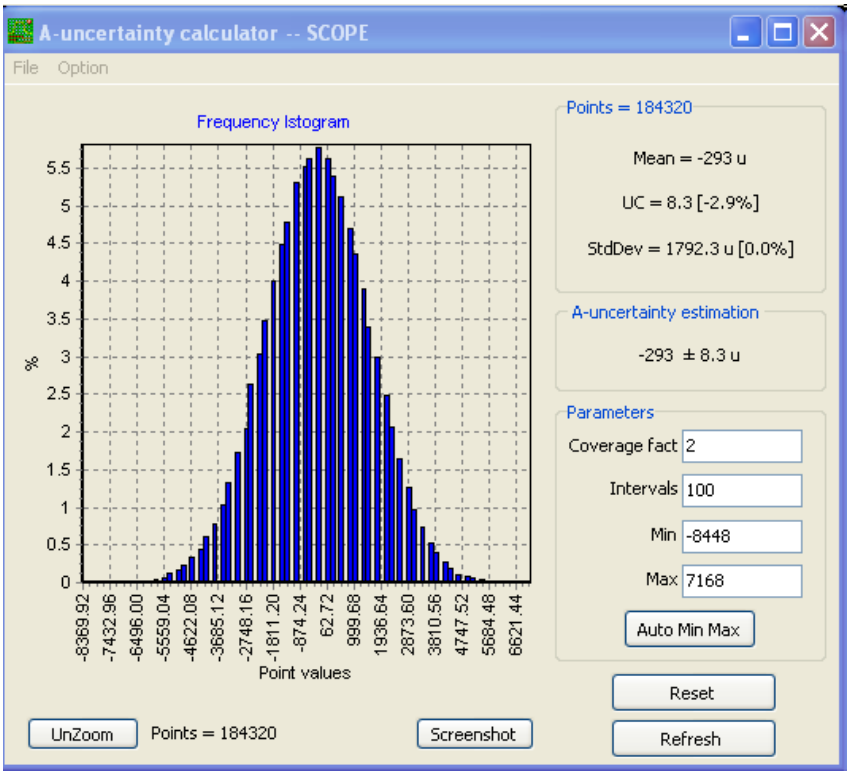


Figura 74: finestra generale per la determinazione incertezza di tipo "A"

Ogni singola finestra per la determinazione dell'incertezza è istanziata separatamente per ogni strumento e realizzata come un thread separato (laddove non prevista specifica, ossia integrata nello strumento di misura, come in ZRLC e Voltmetro)), per ottenere la minore interferenza possibile con il funzionamento in tempo reale del

programma ed al contempo consentire un calcolo efficiente (e anch'esso in tempo reale) di questa componente dell'incertezza.

8.5.1 Calcolo dell'incertezza dovuta alla resistenza di riferimento

Il metodo usato per il calcolo dell'impedenza incognita, implica l'uso di un elemento circuitale preso come riferimento, e senza perdere di generalità per esso è stato utilizzato un bipolo puramente resistivo, detta "resistenza di riferimento". In pratica, come visto in questo stesso capitolo, è stata usata la relazione 8.1, qui ancora una volta riportata per comodità:

$$|Z_x| = \frac{|V_{zx}|}{|V_{rm}|} R_m \tag{8.1}$$

Dalla quale è facile evincere la stretta dipendenza dall'elemento R_m, e dunque dalla incertezza ad essa associata. Se il valore della resistenza di riferimento fosse noto con incertezza nulla, nullo sarebbe il contributo da essa apportato al calcolo dell'incertezza complessiva.

L'idea è quella di portare di conto il contributo di questo elemento, semplicemente a partire dalla conoscenza della sua "tolleranza", parametro normalmente noto di qualsiasi resistore commerciale (normalmente disponibili nei valori di 10%, 5% e 1%).

Oppure, è possibile misurare il resistore di riferimento con un misuratore di precisione, di cui sia nota l'incertezza, ed impostare nel programma i valori misurati. Per impostare i parametri relativi alla resistenza di riferimento, è stata prevista un'apposita finestra, riportata nella figura che segue.

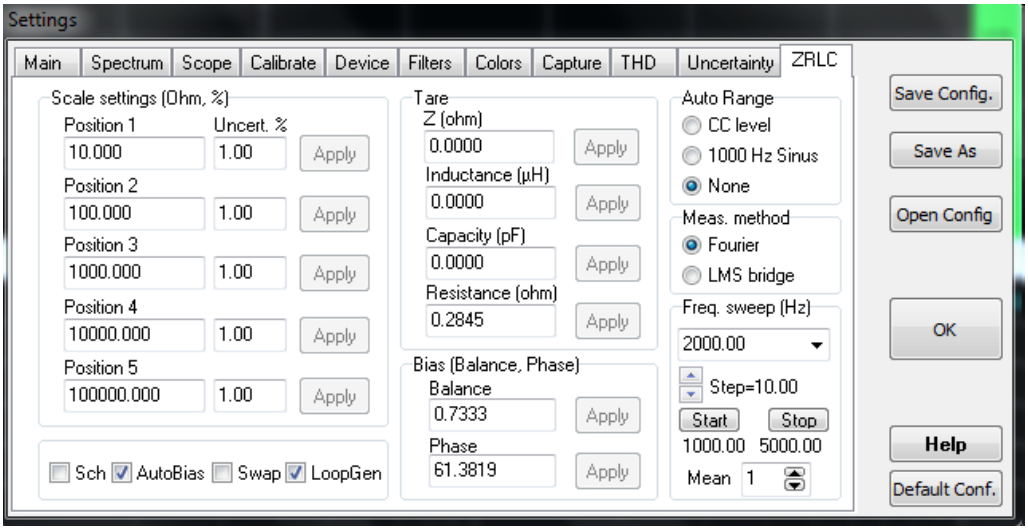


Figura 75: la finestra per impostare l'incertezza della resistenza di riferimento

In essa, tra le varie opzioni, è possibile impostare il valore delle varie resistenze di riferimento e l'incertezza ad esse relativa, che come detto, possono essere direttamente i parametri dichiarati dal costruttore o quelli misurati.

Allo scopo di evitare un calcolo di tipo "white box" ed implementare tecniche che meglio si prestano a uno strumento di questo tipo, caratterizzato da un numero consistente di misure ripetute e calcoli eseguiti in tempo reale, si è optato per un calcolo (eseguito durante la misura) del "delta" di misura che si otterebbe considerando i valori agli estremi della resistenza di riferimento, ossia considerando il valore nominale di resistenza impostato e la tolleranza (incertezza) associata al valore nominale stesso. Il "delta" così calcolato viene ad essere utilizzato per individuare un intervallo di valori all'interno della quale ricade con elevata probabilità la misura; successivamente associamo a questo ventaglio di valori una variabile aleatoria cui assegneremo per semplicità una distribuzione uniforme, la cui media sarà il valore nominale della resistenza. In pratica, essa è pertanto modellato secondo una distribuzione uniforme la cui deviazione standard u^2, dato δx come intervallo, può semplicemente calcolarsi come:

$$u^2 = (\delta x)^2 / 12 \qquad (8.13)$$

da cui è facile calcolare l'incertezza tipo come:

$$u = 0{,}29\ \delta x \qquad (8.14)$$

La quantità δx verrà calcolata per tramite della 8.1 impostando in essa i valori minimi e massimi della resistenza di riferimento. Detta $y = f(x)$ la 8.1, dove la variabile indipendente "x" rappresenta la resistenza di riferimento R_m, e dato un valore di $x=100\Omega$ ed una tolleranza di 5%, otteniamo che $\delta x = f(105) - f(95)$ e dunque è immediato calcolare la quantità "u" per tramite della 8.14.

In pratica il calcolo del δx è effettuato in tempo reale ad ogni misura, e lo stesso dicasi per il calcolo dell'incertezza, che viene aggiornato ad ogni misura e presentato a video sotto forma di incertezza percentuale o assoluta. Il calcolo complessivo dell'incertezza, che tiene di conto di tutti i contributi è aggiornato secondo la relazione presentata nel paragrafo 8.5.4.

8.5.2 Incertezze comprese nel calcolo dell' incertezza statistica

Sebbene la GUM indichi esplicitamente che le incertezze di tipo A e B vanno tra loro sommate quadraticamente (vedi 8.5.4) per il calcolo complessivo dell'incertezza, in effetti molteplici cause d'incertezza sono palesemente già contenute nel calcolo dell'incertezza statistica, ossia appunto quella di tipo A, data la loro natura eminentemente casuale. La stessa GUM, nel paragrafo 4.3.10 recita:

"*E' importante non contare due volte le componenti dell'incertezza. Se una componente dovuta ad un certo effetto è ottenuta da una valutazione di categoria B, essa dovrebbe essere inclusa come componente indipendente nel calcolo dell'incertezza tipo composta, solamente nella misura in cui l'effetto non contribuisce alla variabilità osservata delle osservazioni. Ciò in quanto l'incertezza dovuta alla parte dell'effetto che contribuisce alla variabilità osservata è già inclusa nella componente d'incertezza ottenuta dall'analisi statistica delle osservazioni*"

Di esse consideriamo senza alcun dubbio di tipo casuale gli errori di quantizzazione, di jitter, e noise. L'errore di linearità, differenziale ed integrale, può essere ragionevolmente considerato anch'esso di tipo casuale e quindi compreso in un calcolo statistico.

8.5.3 Incertezza dovuta alla risoluzione

Questa incertezza è generalmente poco considerata, o più spesso dimenticata. La "finitezza" della lunghezza di parola del convertitore A/D, e la scala conseguente ad ogni portata (per esempio, resistore di riferimento di R_{rif} Ω, si è calcolato un range di misura che va da R_{min} a R_{max}) e dunque, dati "k" bit di parola del convertitore A/D si può calcolare la risoluzione δx:

$$\delta x = (R_{max} - R_{min})/2^k \quad \Omega \tag{8.15}$$

Appare immediatamente evidente che la misura sarà affetta da una ulteriore incertezza data dal numero di cifre del display digitale che si può quantificare con il numero δx appena calcolato. Per esempio, se (esagerando solo per fare un esempio) si avesse una risoluzione δx = 10 Ω, una eventuale misura di un valore qualsiasi avrebbe e comunque una incertezza ulteriore di 10Ω . Come riportato dalla GUM in F.2.2.1 detta δx la risoluzione, il valore della sollecitazione che produce una indicazione data X può giacere con uguale probabilità in qualunque punto dell'intervallo compreso tra $X - \delta x/2$ e $X + \delta x/2$. La sollecitazione è allora descritta da una distribuzione di probabilità rettangolare di ampiezza δx con varianza $u^2 = (\delta x)^2/12$, vale a dire un'incertezza tipo $u = 0.29\ \delta x$ per qualsiasi indicazione.

8.5.4 Incertezze rimanenti

L'errore di Guadagno e di Offset presentano invero una duplice natura di sistematicità e casualità. La prima è confermata, oltre che dalla interpretazione teorica di questi fenomeni, anche dal fatto che taluni dispositivi di conversione A/D prevedono appositi meccanismi per la loro automatica compensazione. Parimenti, come più volte ricordato, essi presentano una oggettiva difficoltà di completa compensazione, e la deriva delle caratteristiche dei componenti hardware, oltre a una certa sensibilità alle condizioni di misura (temperatura, umidità, etc.) fanno ritenere per queste grandezze una natura di tipo casuale.

L’esistenza nel programma di una procedura interna per la valutazione dell’incertezza tramite analisi di tipo Monte Carlo semplificata hanno mostrato lo scarso contributo che queste variabili casuali apportano al calcolo dell’incertezza. Inoltre, sarebbe necessario, qualora si volesse far “propagare” il contributo dell’incertezza di Gain e di Offset nella relazione 8.1 (cosa certamente fattibile in questo caso in cui il modello matematico è di assoluta semplicità) effettuare un calcolo “ibrido” che preferiamo al momento evitare. Pertanto, al momento, trascuriamo deliberatamente i contributi di queste grandezza, la cui influenza è comunque mitigata dalle procedure di “azzeramento” previste dallo strumento.

8.5.5 Calcolo complessivo e modalità di visualizzazione

Per finire, tutti i contributi dell’incertezza (A e B) sono utilizzati per il calcolo dell’incertezza complessiva (detta incertezza tipo composta), tramite la classica relazione proposta anche nella GUM (v. cap. II); essa prevede di effettuare la somma dei quadrati delle varie incertezze ed estrarne la radice quadrata, in maniera da esprimere il valore totale dell’incertezza tipo composta:

$$U_{tot} = \sqrt{U_A^2 + U_{b1}^2 + \cdots U_{bm}^2} \tag{8.15}$$

Supposto di avere “*m*” cause d’incertezza di tipo B. Successivamente è possibile definire un fattore di copertura adeguato per determinare l’incertezza estesa. Il fattore di copertura è impostabile nelle preferenze.

8.6 Strumento in funzionamento reale con calcolo incertezza

Per concludere, nella figura 76 è possibile vedere una schermata dello strumento ZRLC in funzione, mentre si misura un condensatore di valore nominale 100nF (il condensatore, tra le varie cose, è molto vecchio) la cui misura è stata effettuata in condizioni standard per VA, ossia frequenza di campionamento di 40960 Hz, numero

di punti per buffer pari a 4096, e frequenza di test pari a 1050 Hz. Queste condizioni permettono, in pochi secondi di misurazione, una notevole quantità di misure; infatti con i parametri scelti si ottengono 10 misure al secondo. La misura è stata ottenuta dopo poco più di 15 secondi (154 misure) ma già dopo pochi secondi i risultati sono stati dello stesso ordine di grandezza di quelli visibili a video. L'incertezza tipo composta, con un fattore di copertura k = 2 (estesa), viene espressa in forma percentuale e assoluta. Essa è stata calcolata secondo quanto esposto in questo capitolo; si osservi che i parametri di funzionamento avrebbero potuto essere scelti in funzione delle esigenze del caso e delle risorse disponibili; per esempio l'uso di un numero di punti del buffer più elevato avrebbe consentito una maggiore accuratezza delle singola misura ma parimenti un numero minore di misure al secondo; alzando la frequenza di campionamento (se il dispositivo di acquisizione lo consente) si può invece aumentare il "tasso" di misure al secondo. Ancora alzando contemporaneamente entrambi questi parametri si possono aumentare accuratezza e numero di misure al secondo, al prezzo di maggior onere computazionale.

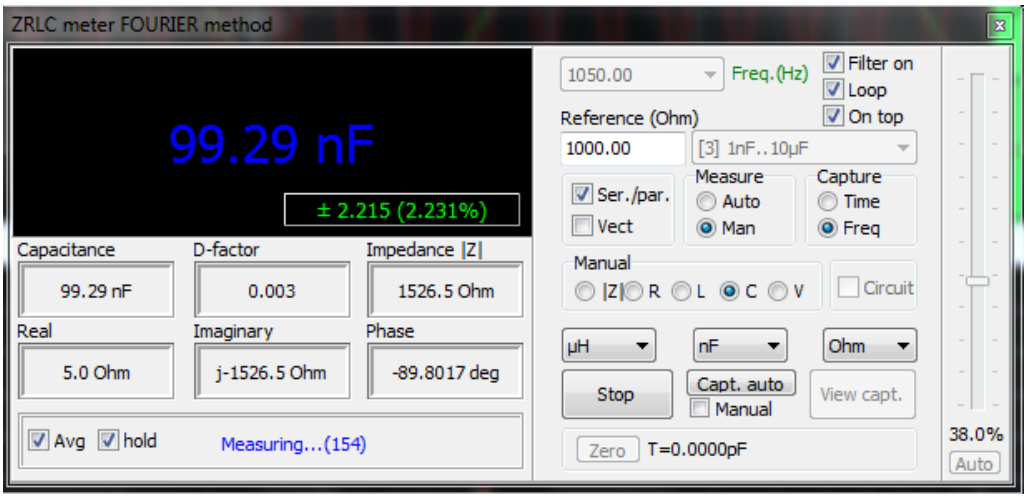


Figura 76: ZRLC e incertezza

La scelta della frequenza di test è un altro parametro particolarmente significativo che può portare ad esplorare diversi aspetti e fenomeni relativi al bipolo sotto misura. In tal senso è possibile effettuare una misura con uno sweep automatico in frequenza (con tanto di calibrazione automatica ad ogni frequenza) che porta alla produzione di un completo diagramma della misura in funzione della frequenza (funzione "capture"

di ZRLC); per ogni misura, anche in sweep, è possibile effettuare misure ripetute in numero arbitrario e calcolarne la media.

Per esempio nella fig. 77 è stato effettuato un sweep in frequenza con i seguenti parametri, oltre quelli già usati per la misura precedente (frequenza di campionamento e dimensioni buffer):

- frequenza di test: da 500 a 3000Hz con passo di 100Hz (quindi 26 punti totali, compreso il primo a 500 Hz);
- ogni misura ad ogni singola frequenza è stata ripetuta e mediata 5 volte (valore impostabile a piacere, senza limitazioni altro che il tempo e le risorse a disposizione);
- punti acquisiti catturati in apposita finestra, e raccordati con metodo di interpolazione "cubic spline", con conseguente possibilità di memorizzazione, stampa e importazione diretta in formato Word/Excel (tramite clipboard).

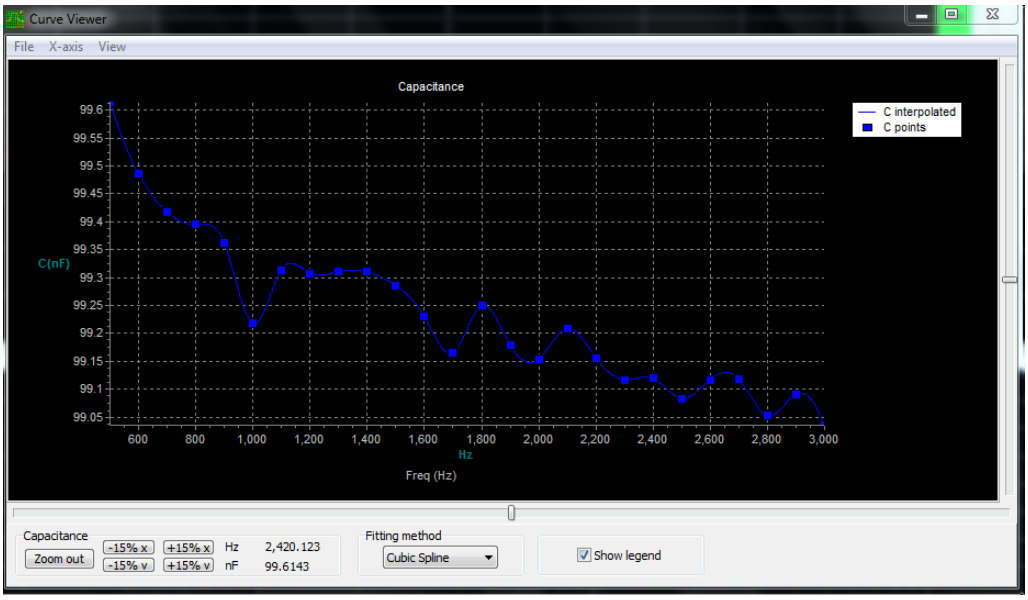


Figura 77: sweep di misure automatiche

Per la stessa misura, la finestra di "cattura incertezza statistica" dopo poche decine di punti acquisiti presenta un istogramma del tipo visibile in fig. 78;

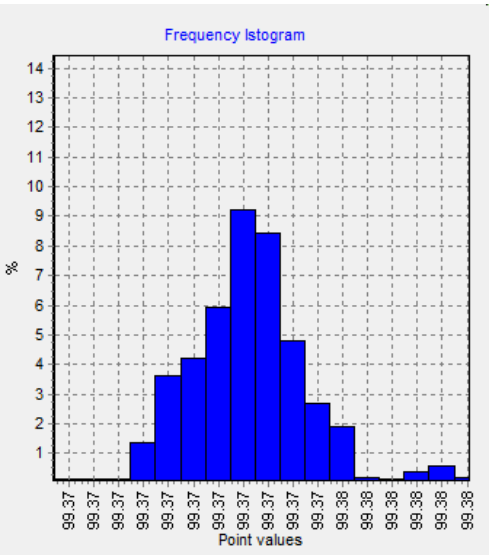


Figura 78: incertezza statistica

essa sembra avvicinarsi ragionevolmente ad una distribuzione di tipo gaussiano (teorema del limite centrale) data la molteplicità di variabili aleatorie che concorrono alla definizione della variabile aleatoria risultato della misura.

Ancora, nella figura 79 è visibile l'andamento delle due tensioni rilevate dallo strumento, una delle quali proporzionali alla corrente che scorre nel bipolo.

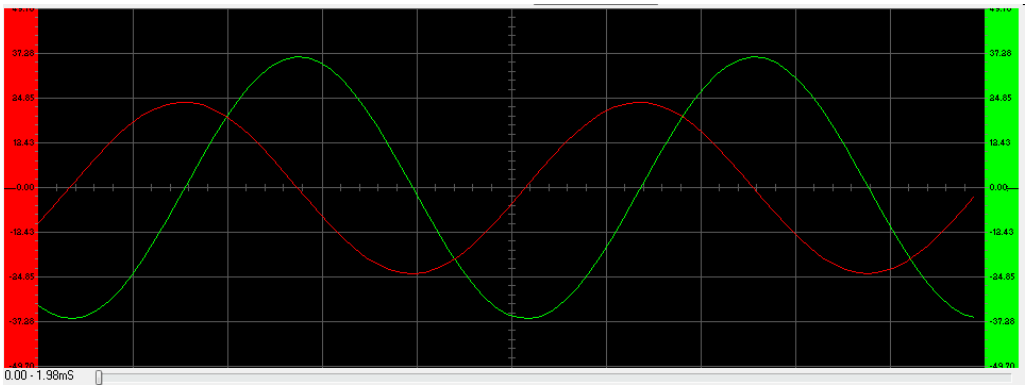


Figura 79: sfasamento tra tensione e corrente nel dominio del tempo

Congruentemente con quanto indicato dallo strumento di misura è possibile rilevare uno sfasamento di circa 90 gradi, che riportiamo nell'ulteriore figura

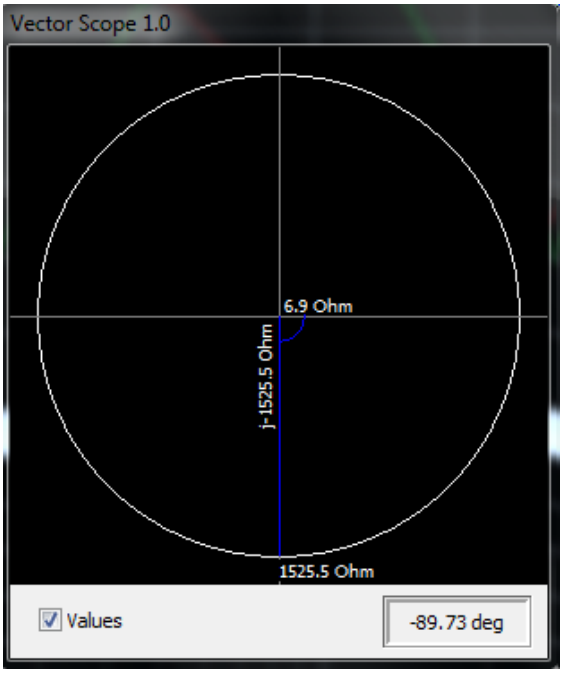


Figura 80: sfasamento con vettorscopio

80 riportante l'indicazione del "vettorscopio" in tempo reale. La leggera discrepanza che si può notare sul valore della parte reale dell'impedenza rispetto a quanto indicato nella schermata di ZRLC è dovuta al fatto che essa è stata acquisita in un secondo tempo, ossia è relativa ad una misura eseguita in un momento diverso (giorni di distanza) e in condizioni di misura generalmente differenti (temperatura, umidità, e persino scheda di acquisizione fisicamente diversa, anche se stesso modello).

Capitolo 9

Hardware per ZRLC

9.1 Introduzione

In questo capitolo verrà presentato un hardware progettato "ad hoc" per l'uso con VA e la funzione di misuratore d'impedenza (ZRLC). Come ampiamente descritto nel cap. VIII, l'implementazione di ZRLC richiede la messa a punto di un semplice circuito suppletivo da affiancare alla scheda di acquisizione/generazione a due canali; esso consta di un partitore resistivo e due amplificatori operazionali usati come elevatori d'impedenza.

Tramite l'aggiunta di pochi altri componenti, è stata realizzata una vera e propria scheda completamente "autonoma", ossia capace di effettuare anche le operazioni di acquisizione e generazione del segnale; e che dunque ci consente avere un circuito completo, a basso costo, e facilmente utilizzabile su macchine diverse, mantenendo costanti i parametri di calcolo dell'incertezza e consentendo di avere il massimo controllo su tutta la catena di misura. Ed inoltre, come sempre accade per i circuiti "esterni" ad un PC, di limitare in maniera consistente i rischi di danneggiamento dello stesso in caso di malfunzionamento. A rigore un completo disaccoppiamento sarebbe effettuabile solo con dei fotoaccoppiatori, ma in questo caso, visti i livelli di correnti e tensioni in gioco, non si è ritenuto necessario adottare una simile soluzione.

Il circuito dunque comprende il convertitore A/D, quello D/A, oltre naturalmente al filtro anti-aliasing ed i circuiti di supporto per il bus di comunicazione con il PC; ed inoltre, tutta la componentistica necessaria per ricavare la tensione di alimentazione da quella eventualmente disponibile nel PC stesso. Per inciso, i componenti utilizzati sono volutamente "commerciali" e di basso costo, e nonostante ciò le prestazioni prima stimate e poi misurate sul campo sono decisamente di ottimo livello (elevata accuratezza), dimostrando la validità degli algoritmi utilizzati sia per il calcolo delle grandezze che per la minimizzazione degli errori sistematici. In altre parole, dimostrando la validità "dell'hardware realizzato in software" (cfr. capitolo I).

Il presente progetto è stato fatto in collaborazione con la nota rivista di elettronica amatoriale "Nuova Elettronica", la quale da tempo usa VA come software di base per numerosi strumenti di misura e per articoli didattici.

Inoltre, la tedesca ROGA-instruments ha adottato Visual Analyser come software di base "freeware" per le sue schede di misura professionali (vedi fig. 81); essa ci ha inoltre fornito apparecchiature di elevatissimo livello (es.: ROGADAQ2) tramite le quali testare, migliorare e mettere a punto il programma oltre aggiungere nuove funzioni e consentirci un calcolo esatto dell'incertezza (la scheda è caratterizzata metrologicamente). Essa inoltre è accoppiata in continua, caratteristica che ha permesso lo sviluppo di questa ulteriore funzione in VA, sia in entrata che in uscita (in questo secondo caso è possibile generare anche una tensione continua).

Figura 81: scheda professionale RODAdaq2 della ROGA-instruments, utilizza VA come software di base

Lo schema presentato in questo capitolo è stato pubblicato sulla rivista Nuova Elettronica, numero 242 pagg. 8..44, con esplicito riferimento alla collaborazione con l'Università di Roma "Tor Vergata"; la maggior parte delle figure sono state tratte liberamente dal medesimo articolo con esplicita autorizzazione della rivista stessa.

9.2 Schema di principio

Lo schema di principio per la misura, già ampiamente descritto nel cap. VIII è riportato per comodità di lettura nella figura 82 con l'aggiunta di ulteriori dettagli. Si ribadisce che le figure e gli schemi usate in questo capitolo sono liberamente tratte dall'articolo della rivista Nuova Elettronica scritto e pubblicato in collaborazione con la stessa.

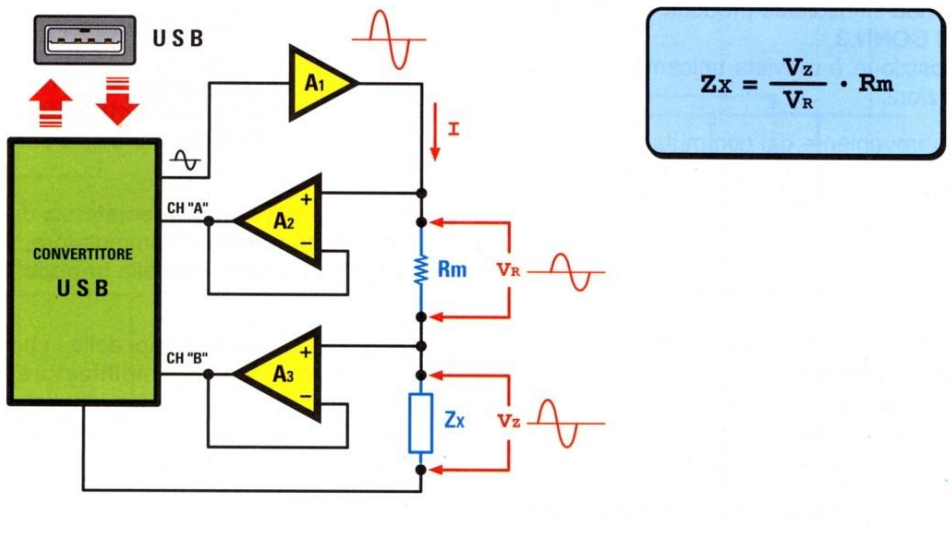


Figura 82: Schema di principio

In esso è possibile distinguere chiaramente gli elementi costitutivi di base, quali la resistenza di riferimento, l'impedenza sotto misura, i segnali applicati e letti più i buffer elevatori d'impedenza (A2, A3), ed infine l'amplificatore di potenza del

segnale di test applicato al partitore (A1) . E' inoltre riportata la formula utilizzata per il calcolo del modulo dell'impedenza, dalla quale, tramite il calcolo della fase tra le due tensioni V_Z e V_R è possibile ricavare l'impedenza del DUT (Device Under Test, in questo caso Z_x). Si osservi poi che è stato aggiunto il blocco denominato "convertitore USB", il quale comprende essenzialmente:

- i convertitori A/D e D/A;
- il filtro anti-aliasing;
- interfaccia USB, usata come bus di comunicazione con il PC;
- circuiti per la generazione delle tensioni necessarie a partire dalla tensione disponibile sulla porta USB.

La scelta di utilizzare l'interfaccia USB è stata effettuata data l'enorme diffusione che essa ha nel mondo dei personal computer, la disponibilità di circuiti integrati dedicati, e la possibilità di prelevare da essa l'alimentazione necessaria per tutti i componenti attivi utilizzati nel circuito; elemento affatto trascurabile, che consente di eliminare una quantità consistente di circuiti aggiuntivi di prelievo e trasformazione dell'energia (alimentatore), oppure di ricorrere a pile. Infatti, lo standard USB comprende due terminali per l'invio e la ricezione dei dati, un terminale di massa ed un terminale di alimentazione, da dove è possibile prelevare una tensione di 5 Volt con una corrente massima di 500 mA. In taluni casi, se è necessaria una corrente maggiore, è possibile sfruttare un'altra porta USB libera dalla quale prelevare ulteriore potenza.

Per poter misurare impedenze nel range più ampio possibile, come vedremo nel prossimo paragrafo, sono state utilizzate più resistenze di riferimento la cui selezione è effettuata con un commutatore rotativo a cinque posizioni; questo perché (cap. VIII) la misura risulta più accurata se l'ordine di grandezza della resistenza campione è dello stesso ordine di grandezza del DUT. Inoltre, è necessario utilizzare un amplificatore che porti il livello del segnale di uscita del convertitore D/A (usato per la generazione del segnale di test, A1) ad un livello adatto al pilotaggio del carico costituito dalla serie della resistenza di riferimento e dal DUT. In altri termini, per cercare di abbassare il più possibile l'impedenza d'uscita del generatore di tensione,

in maniera da renderlo più vicino ad un generatore “ideale” e ridurre così la perturbazione introdotta nel sistema sotto misura. Da non confondere con la perturbazione provocata dalla circuitazione preposta al “prelievo” del segnale, per la quale è invece auspicabile una impedenza più elevata possibile.

9.3 Schema elettrico completo

Il circuito elettrico completo e riportato in fig. 83, mentre la lista componenti nella fig. 84.

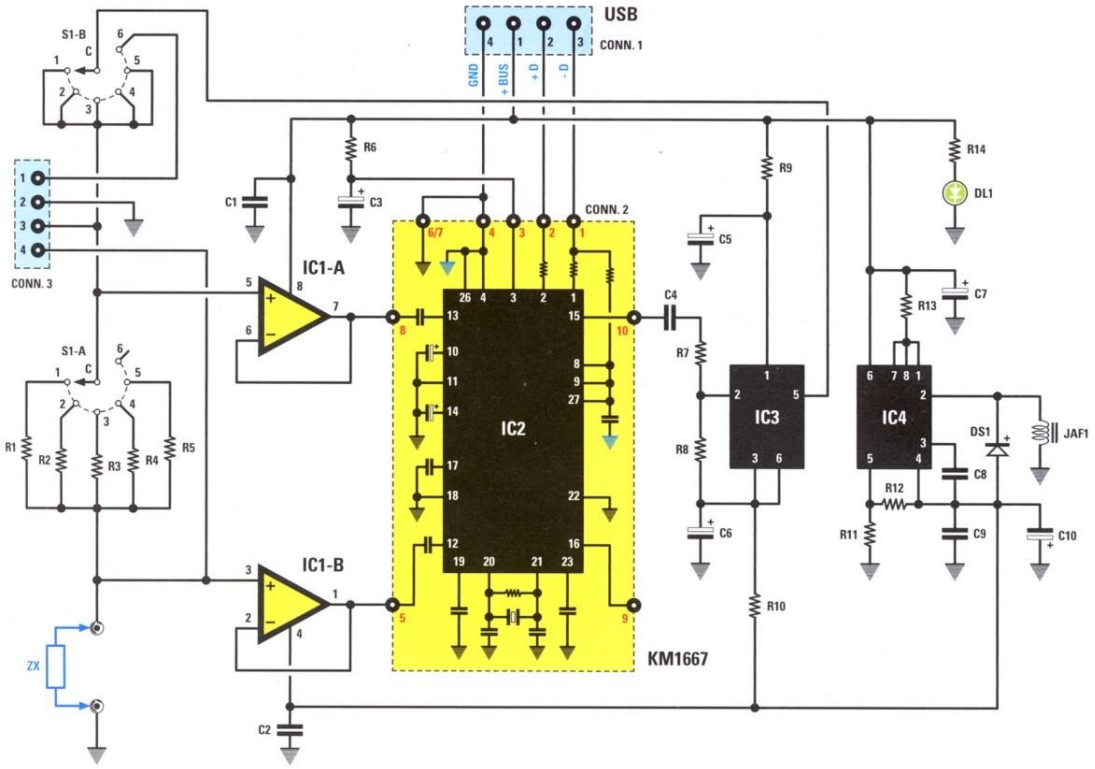


Figura 83: il circuito completo

In esso possiamo innanzitutto notare la presenza di un partitore resistivo “variabile” a mezzo di un commutatore rotativo a 5 posizioni; tramite esso è possibile modificare il valore della resistenza di riferimento, e dunque modificare la portata dello strumento. Le cinque resistenze utilizzate saranno a bassa tolleranza (tipicamente 1%). Il valore nominale delle resistenze viene impostato in una apposita sezione del

programma, insieme alla tolleranza dichiarata dal costruttore; oppure questo stesso valore può venire misurato con uno strumento che fornisce anche l'intervallo di fiducia (o l'incertezza), da impostare anch'essa nel programma al posto della tolleranza dichiarata dal costruttore. Essa deve essere espressa in percentuale ± rispetto al valore nominale impostato. Tramite la conoscenza di queste grandezze, il programma sarà in grado di calcolare la componente di incertezza che la resistenza campione aggiunge al calcolo dell'incertezza complessiva. Chiaramente, se si ha la disponibilità di uno strumento di misura, la cui incertezza è migliore dell'incertezza che il costruttore dichiara per le resistenze campione, sarà possibile spuntare una incertezza composta complessiva migliore. Nel caso dunque si disponga di un ohmmetro di elevata qualità, effettueremo le misure delle resistenze di riferimento, annotando anche il valore dell'incertezza; successivamente dovremo impostare, nell'apposita finestra delle impostazioni (cfr. cap. V, VIII) i valori misurati, con le cifre decimali significative, ed il valore di incertezza che lo strumento dichiara per quella misura e quella portata. Quest'ultima quantità è normalmente calcolata con una formula, fornita dal costruttore, che tiene conto di tutti i parametri necessari (portata, fondo scala, etc.); si osservi che il miglioramento dell'incertezza associata alle misure sarà attendibile se lo strumento usato è "calibrato" e "tarato", ossia se il suo certificato di taratura è ancora in corso di validità.

R1 = 10 ohm 1%
R2 = 100 ohm 1%
R3 = 1.000 ohm 1%
R4 = 10.000 ohm 1%
R5 = 100.000 ohm 1%
R6 = 10 ohm
R7 = 33.000 ohm
R8 = 680 ohm
R9 = 1 ohm
R10 = 1 ohm
R11 = 8.200 ohm
R12 = 2.700 ohm
R13 = 0,22 ohm
R14 = 680 ohm
C1 = 100.000 pF poliestere
C2 = 100.000 pF poliestere
C3 = 100 microF. elettrolitico
C4 = 1 microF. poliestere
C5 = 100 microF. elettrolitico
C6 = 100 microF. elettrolitico
C7 = 100 microF. elettrolitico
C8 = 1.500 pF poliestere
C9 = 100.000 pF poliestere
C10 = 470 microF. elettrolitico
JAF1 = impedenza 100 microHenry
DS1 = diodo tipo BYW100
DL1 = diodo led
IC1 = integrato tipo NE5532
IC2 = circuito SMD tipo KM1667
IC3 = integrato tipo TDA7052
IC4 = integrato tipo MC34063A
S1 = commutatore 2 vie 6 pos.
CONN.1 = connettore USB
CONN.2 = connettore 10 pin
CONN.3 = connettore 4 pin

Figura 84: lista componenti

Gli integrati IC1-A e IC1-B (NE5532, amplificatore operazionale doppio) svolgono la funzione di amplificatori a guadagno unitario utilizzati per elevare l'impedenza d'ingresso degli stadi successivi, e dunque ridurre al minimo possibile la perturbazione introdotta nel sistema sotto misura. Le impedenze d'ingresso di IC1-A e IC1-B vengono infatti a trovarsi in parallelo al partitore; in particolare IC1-B viene a trovarsi direttamente in parallelo al DUT, mentre IC1-A si trova in parallelo alla serie costituita dal resistore di riferimento e dal DUT. La condizione ideale, ossia a perturbazione nulla, si verificherebbe se l'impedenza di ingresso di IC1A/B fosse infinita. I segnali in uscita da IC1A/B vengono poi applicati all'integrato PCM2902 tramite un condensatore di disaccoppiamento, che elimina la componente continua eventualmente presente; l'integrato PCM2902 non è infatti in grado di gestire segnali non variabili nel tempo (continui). In tal senso abbiamo condotto numerose prove che hanno confermato questa caratteristica. Il PCM2902 è stato acquistato in una configurazione premontata (SMD) su una piccola basetta che contiene i componenti essenziali al suo corretto funzionamento, così come indicato sul "datasheet" stesso nel circuito applicativo d'esempio (la basetta è stata fornita dalla ditta Nuova Elettronica, con la sigla KM1667, vedi appendice II per uno stralcio del "datasheet" dell'integrato), tra i quali il quarzo che fornisce il segnale di clock. Essa viene fornita con un connettore a 10 piedini (v. fig 85). Questo integrato è infatti un completo dispositivo di acquisizione/generazione dati, in grado di effettuare la conversione A/D e D/A dei segnali, effettuare il filtraggio antialiasing e gestire l'interfaccia USB tramite il noto e affidabilissimo protocollo HID (Human Interface Device).

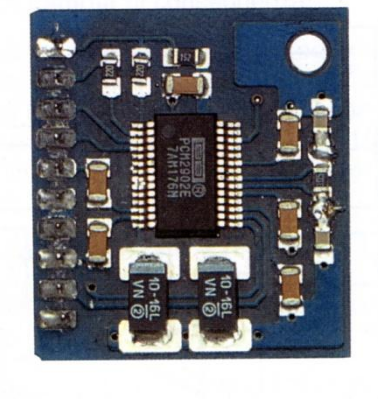


Figura 85: il modulo KM1667

Il connettore “conn 1” collegato ai piedini 1,2,3,4 del modulo KM1667 consente di collegare un cavo USB standard, dal quale prelevare/inviare i dati (piedini 1 & 2); inoltre esso fornisce l’alimentazione al modulo KM stesso ed a tutto il circuito: la massa (GND) è collegata al piedino 4 mentre il positivo al piedino 3 tramite la resistenza R6 ed il condensatore elettrolitico di filtro C3.

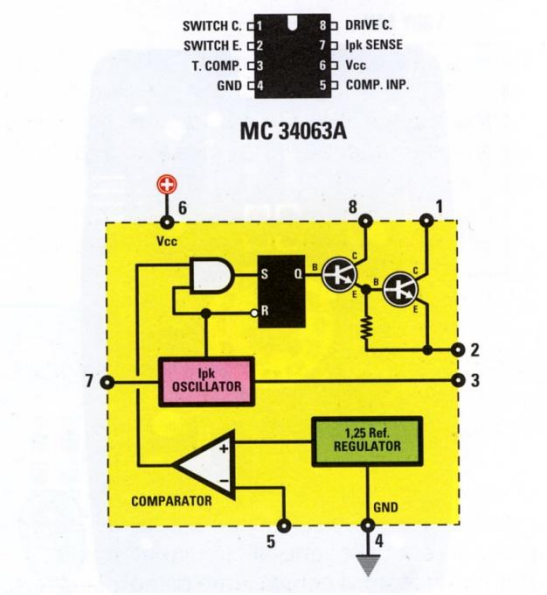


Figura 86: integrato MC34063A

Questa stessa alimentazione è utilizzata dall’integrato IC4, siglato MC34063A (v. fig. 86) il quale altro non è che un regolatore switching che consente di ottenere una tensione di -5 Volt partendo dalla tensione di alimentazione di +5 Volt fornita dalla porta USB.

In tal modo è possibile alimentare gli amplificatori operazionali IC1A/B in maniera duale, semplificando il circuito d’ingresso e potendo per esso migliorare la risposta in frequenza (non è necessario applicare ulteriori condensatori in ingresso). Analogamente, l’alimentazione duale è fornita all’integrato IC3 (TDA7052, vedi fig. 87) che è in pratica un amplificatore di potenza miniaturizzato da 1 watt.

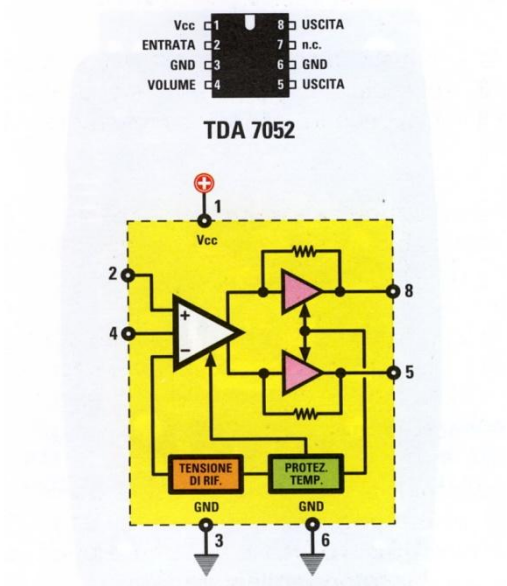


Figura 87: integrato TDA7052

Tra i suoi pregi, quello di essere a basso costo e di richiedere per il suo funzionamento un numero di componenti veramente ridotto. L'uso di questo amplificatore, lo ricordiamo, è necessario per poter fornire la giusta corrente e tensione (e dunque la potenza) per il pilotaggio del partitore resistivo tramite il quale si effettua la misura; ossia, in altri termini, per rendere il più possibile il segnale di test costante ed indipendente dal carico. In ultimo, da notare che il led DL1 in serie alla resistenza R14 segnala la presenza di alimentazione e dunque l'accensione del circuito. In fig. 88 è possibile vedere il circuito montato su circuito stampato e nella sua versione commerciale in apposito contenitore.

Figura 88: versione completa di ZRLC fatta in collaborazione con la rivista Nuova Elettronica

9.4 Uso dello strumento ed eliminazione di un errore sistematico macroscopico

L'uso del dispositivo proposto è particolarmente (e volutamente) semplice; non è richiesta installazione, dato che ogni sistema operativo della famiglia Windows è in grado di riconoscere la periferica come "USB audio CODEC". Dunque è necessario solamente inserire il cavo USB e successivamente mandare in esecuzione il programma. Alternativamente, è possibile inserire il connettore USB dopo aver lanciato il programma, ed allora sarà sufficiente "cercare" il dispositivo tramite appositi meccanismi di ricerca automatica presenti in VA o manualmente.Il programma è stato sviluppato nell'ottica di essere immediatamente utilizzabile, senza dover passare per una serie interminabile di settaggi come spesso accade per molti strumenti virtuali; tuttavia, rimane solo un settaggio semplice da effettuare "una-tantum", e che serve per eliminare una delle cause di incertezza più grossolane che l'uso dell'integrato PCM2902 inevitabilmente introduce. Esso infatti è capace di gestire due canali in ingresso e altrettanti in uscita (sebbene in uscita abbiamo utilizzato solo un canale, il canale rimanente verrà utilizzato per future applicazioni in corso di messa a punto). Questi due canali, come spesso accade per molti dispositivi di questo tipo, sono gestiti da un solo convertitore A/D e D/A utilizzato in multiplexing. Ossia, i campioni relativi ai due canali vengono convertiti serialmente, determinando lo sfasamento di un campione (cfr. cap. VIII) di un canale rispetto all'altro. Come ampiamente spiegato in VIII questo comporta una differenza di fase tra i due canali che a sua volta comporta un errore sulla misura di entità assolutamente inaccettabile; si vedano le considerazioni fatte nel medesimo capitolo. Questo errore può a ragione essere annoverato tra quelli di tipo "sistematico" ed in questo caso di tipo macroscopico. Il problema può essere tuttavia agevolmente eliminato ritardando un canale di un campione; VA implementa una coppia di code capaci di gestire ritardi di dimensioni arbitrarie su entrambi i canali; i valori di questi parametri possono essere impostati per tramite della finestra di "Settings" ampiamente illustrata nel capitolo V relativo all'architettura del programma, di cui riportiamo in figura 89 la schermata principale per comodità di consultazione.

L’eliminazione di questo errore è dunque possibile, ma non elimina del tutto l’errore di fase complessivo tra i due canali; esso infatti non è dovuto esclusivamente al convertitore A/D, ma dipende anche dagli amplificatori presenti nel percorso del segnale e dal cablaggio stesso. Si veda il cap. IV per una trattazione esaustiva.

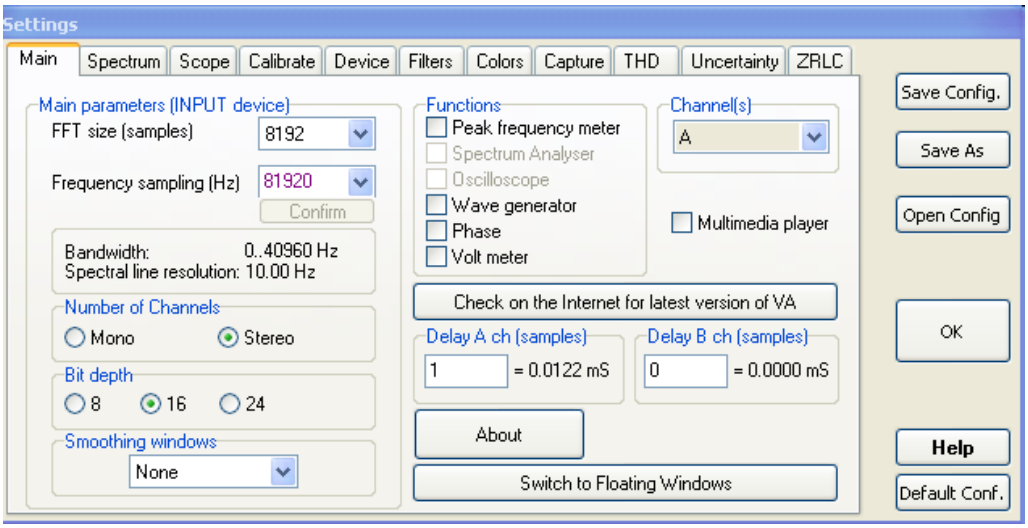


Figura 89: Finestra di “settings” e compensazione dei ritardi

9.5 Caratterizzazione metrologica

La caratterizzazione metrologica di uno strumento completo, o di una scheda di acquisizione, non può essere effettuata tramite semplici calcoli o comunque misure effettuate su di un singolo oggetto; esso richiede attrezzature specifiche e la disponibilità di numerosissimi esemplari della scheda/strumento per poter condurre analisi statisticamente significative.

Tramite la realizzazione di questo circuito abbiamo tuttavia potuto caratterizzare alcuni aspetti dell’incertezza ad esso associata desumendoli dalle caratteristiche dichiarate dai vari costruttori dell’integrato PCM2902, degli amplificatori operazionali d’ingresso e dell’amplificatore d’uscita. In particolare la conoscenza dell’errore di quantizzazione, di gain, e del jitter per tramite dei datasheet del quarzo utilizzato, più la linearità e il rapporto segnale-rumore (SNR) ci ha consentito, impostando questi valori nel programma, di abbozzare un primo calcolo dell’incertezza. Affiancando le informazioni ottenute tramite questi dati decisamente

di “tipo B” ad un’analisi statistica (e dunque una incertezza di “tipo A”), quale quella che può fare VA, possiamo giungere ad una stima dell’incertezza non eccessivamente lontana dalla realtà. Si veda in generale per i dettagli la descrizione relativa ai vari strumenti implementati; in particolare per ZRLC si consulti il capitolo appositamente dedicato.

Appendice 1

Visual Analyser: introduzione originale

Il punto di partenza del presente lavoro è un programma, denominato ***Visual Analyser,*** che dimostra come sia possibile utilizzare un personal computer per simulare un completo set di strumenti per il laboratorio Elettronico, che usi esclusivamente la dotazione hardware di base ed un sistema operativo standard. In altre parole, senza praticamente alcun hardware aggiuntivo (che al massimo si limita all'uso di un set di cavi ed opzionalmente un partitore resistivo) e nessuna modifica al sistema operativo (Windows o Linux). Dimostrando così come un semplice personal computer sia effettivamente una macchina universale adatta agli impieghi più diversi e complessi.

Gli strumenti di misura ed analisi dei segnali elettrici, normalmente utilizzati nei laboratori di Elettronica, prelevano il segnale elettrico generalmente tramite delle "sonde" o "puntali" li amplificano e li elaborano tramite appositi circuiti detti "di condizionamento" (es. amplificatori, filtri, protezioni, etc.) e li visualizzano su un apposito display, che può essere un semplice galvanometro od uno schermo a raggi catodici/lcd o altri dispositivi meno sofisticati. Per esempio un multimetro è generalmente usato per misurare valori di tensione e corrente sia continua che alternata (in questo secondo caso ne misurano il valore RMS o true-RMS) o per la misura di valori di resistenza, e fa generalmente uso di un galvanometro opportunamente graduato, mentre un oscilloscopio è usato per visualizzare in tempo reale i segnali elettrici, tramite uno schermo su cui si visualizza l'ampiezza del segnale in funzione del tempo. Ancora, un analizzatore di spettro serve a scomporre un segnale nelle sue componenti armoniche in ampiezza e fase e rappresentato a

video in due grafici separati, detti diagramma delle ampiezze e diagramma di fase, entrambi in funzione della frequenza.

In generale gli strumenti di misura si dividono grossolanamente in due grandi categorie: analogici e digitali. I primi si limitano ad acquisire il segnale direttamente, lo elaborano con circuitazione analogica, e lo visualizzano ancora in forma analogica (vedi il classico multimetro analogico con strumento a bobina mobile o l'oscilloscopio). I secondi effettuano prima una conversione da segnale analogico (ossia tempo continuo) a segnale digitale (ossia tempo discreto e quantizzato) in maniera da poter utilizzare per l'analisi ed elaborazione uno o più microprocessori e/o hardware specifico, ottenendo prestazioni in genere nettamente superiori ai primi o comunque con migliore versatilità e/o aggiornabilità. Questa seconda tipologia a di strumenti sfrutta un teorema fondamentale, e dunque la relativa circuitazione che lo implementa, detto "Teorema del campionamento" o "Teorema di Shannon-Nyquist" tramite il quale è possibile trasformare ogni segnale elettrico analogico (ossia un segnale la cui variabile temporale è un numero reale) in un segnale discreto (la variabile temporale è un numero intero, l'ampiezza ancora continua). Successivamente il segnale così ottenuto viene "digitalizzato" (quantizzato), ossia ad ogni campione viene associato un numero (intero) trasformando di fatto dei segnali elettrici tempo continuo in una sequenza di numeri "comprensibili" ad un normale elaboratore (per esempio anche un PC) e dunque memorizzabili in RAM come sequenze di numeri .

Ogni personal computer, quasi senza eccezione, è dotato di una scheda sonora, e quindi dei necessari circuiti di amplificazione, campionamento e digitalizzazione dei segnali, più altri accessori particolarmente utili quali ingressi multipli, mixer e filtri, e talune persino preamplificatori particolari (es. preamplificatori con curva di equalizzazione RIAA per dischi in Vinile). E con la possibilità di acquisire via software applicativo i campioni numerici (tramite API di sistema), accumularli in RAM ed effettuare su di esse opportune elaborazioni.

Il programma base da cui partiamo fa uso della scheda sonora come strumento di acquisizione dei segnali elettrici; esso pertanto ricade nella categoria degli strumenti di tipo digitale, ed utilizza come strumento di visualizzazione il monitor del PC. E' facile immaginare le enormi potenzialità di un software di questo tipo, i cui limiti sono dettati esclusivamente dalla qualità della scheda sonora e dalla potenza di calcolo dell'hardware del PC stesso. Il PC diventa così terreno fertile per la pratica applicazione di tutti i concetti della materia nota come "*elaborazione numerica dei segnali*" e "*analisi numerica*". Infatti, la disponibilità di un segnale numerico in memoria ed un hardware veloce, in sinergia con l'utilizzo di algoritmi sofisticati, consentono di simulare un numero enorme di "circuiti virtuali" che rendono l'elaboratore ancora più flessibile ed adatto a simulare la maggior parte degli strumenti di misura. Tra questi è notevole la famosissima trasformata di Fourier (tramite la quale effettuare analisi armonica), od i filtri digitali (FIR o IIR tramite i quali implementare filtri le cui prestazioni sono impensabili nel mondo analogico) e molti altri ancora. Ed infine con la possibilità di automatizzare sinergicamente l'uso dei vari strumenti: per esempio si può generare un segnale di test, inviarlo all'ingresso di un DUT (Device Under Test) da analizzare ad intervalli di tempo predefiniti, contemporaneamente effettuare la lettura, ed infine il confronto e visualizzazione su schermo dei risultati ottenuti.

Pur tuttavia è da notare come le moderne schede sonore siano comunque limitate al campo delle basse frequenze: nelle più sofisticate si arriva a frequenze di campionamento dell'ordine di 192 kHz, e quindi a bande passanti pari a circa 96 kHz e che generalmente non comprendono la componente continua. E' da notare altresì che un numero consistente di circuiti elettronici tratta segnali esclusivamente ad audio frequenza, e che in ogni modo Visual Analyser è predisposto per l'uso di schede di acquisizione dati dedicate (e quindi con frequenze di campionamento paragonabili a strumenti professionali). Inoltre, l'uso del paradigma Object Oriented rende facile la modifica del programma in funzione di schede progettate "ad hoc" per specifiche applicazioni.

Pagina lasciata intenzionalmente bianca

Appendice2

Datasheet integrato pcm2902

Si riporta nelle pagine seguenti uno stralcio del datasheet dell'integrato Texas Instruments utilizzato nell'hardware progettato in collaborazione con la rivista Nuova Elettronica per lo strumento virtuale ZRLC descritto nell'ultimo capitolo.

SLES035E–MARCH 2007–REVISED MARCH 2008

STEREO AUDIO CODEC WITH USB INTERFACE, SINGLE-ENDED ANALOG INPUT/OUTPUT AND S/PDIF

FEATURES

- PCM2900: Without S/PDIF
- PCM2902: With S/PDIF
- On-Chip USB Interface:
  - With Full-Speed Transceivers
  - Fully Compliant With USB 1.1 Specification
  - Certified by USB-IF
  - Partially Programmable Descriptors (1)
  - USB Adaptive Mode for Playback
  - USB Asynchronous Mode for Record
  - Bus Powered
- 16-Bit Delta-Sigma ADC and DAC
- Sampling Rate:
  - DAC: 32, 44.1, 48 kHz
  - ADC: 8, 11.025, 16, 22.05, 32, 44.1, 48 kHz
- On-Chip Clock Generator With Single 12-MHz Clock Source
- Single Power Supply: 5 V Typical (V_{BUS})
- Stereo ADC
  - Analog Performance at V_{BUS} = 5 V
    - THD+N = 0.01%
    - SNR = 89 dB
    - Dynamic Range = 89 dB
  - Decimation Digital Filter
    - Pass-Band Ripple = ±0.05 dB
    - Stop-Band Attenuation = 65 dB
  - Single-Ended Voltage Input
  - Antialiasing Filter Included
  - Digital LCF Included
- Stereo DAC
  - Analog Performance at V_{BUS} = 5 V
    - THD+N = 0.005%
    - SNR = 96 dB
    - Dynamic Range = 93 dB
  - Oversampling Digital Filter
    - Pass-Band Ripple = ±0.1 dB
    - Stop-Band Attenuation = –43 dB
  - Single-Ended Voltage Output
  - Analog LPF Included
- Multifunctions:
  - Human Interface Device (HID) Volume ± Control and Mute Control
  - Suspend Flag
- Package: 28-Pin SSOP

(1) The descriptor can be modified by changing a mask.

APPLICATIONS

- USB Audio Speaker
- USB Headset
- USB Monitor
- USB Audio Interface Box

DESCRIPTION

The PCM2900/2902 is Texas Instruments' single-chip USB stereo audio codec with USB-compliant full-speed protocol controller and S/PDIF (only PCM2902). The USB protocol controller works with no software code, but the USB descriptors can be modified in some areas (e.g., vendor ID/product ID). The PCM2900/2902 employs SpAct™ architecture, TI's unique system that recovers the audio clock from USB packet data. On-chip analog PLLs with SpAct architecture enable playback and record with low clock jitter and with independent playback and record sampling rates.

This integrated circuit can be damaged by ESD. Texas Instruments recommends that all integrated circuits be handled with appropriate precautions. Failure to observe proper handling and installation procedures can cause damage.

ESD damage can range from subtle performance degradation to complete device failure. Precision integrated circuits may be more susceptible to damage because very small parametric changes could cause the device not to meet its published specifications.

PACKAGING ORDERING INFORMATION

PCM2900						
PRODUCT	PACKAGE-LEAD	PACKAGE DESIGNATOR	SPECIFIED TEMPERATURE RANGE	PACKAGE MARKING	ORDERING NUMBER(1)	TRANSPORT MEDIA
PCM2900E	SSOP-28	28DB	–25 C to 85 C	PCM2900E	PCM2900E	Rails
					PCM2900E/2K	Tape and reel

(1) Models with a slash (/) are available only in tape and reel in the quantities indicated (e.g., /2K indicates 2000 devices per reel). Ordering 2000 pieces of PCM2900E/2K gets a single 2000-piece tape and reel.

PCM2902						
PRODUCT	PACKAGE-LEAD	PACKAGE DESIGNATOR	SPECIFIED TEMPERATURE RANGE	PACKAGE MARKING	ORDERING NUMBER(1)	TRANSPORT MEDIA
PCM2902E	SSOP-28	28DB	–25 C to 85 C	PCM2902E	PCM2902E	Rails
					PCM2902E/2K	Tape and reel

(1) Models with a slash (/) are available only in tape and reel in the quantities indicated (e.g., /2K indicates 2000 devices per reel). Ordering 2000 pieces of PCM2902E/2K gets a single 2000-piece tape and reel.

ABSOLUTE MAXIMUM RATINGS

over operating free-air temperature range (unless otherwise noted)(1)

			PCM2900/PCM2902	UNIT
V_{BUS}	Supply voltage		–0.3 to 6.5	V
	Ground voltage differences, AGNDC, AGNDP, AGNDX, DGND, DGNDU		±0.1	V
	Digital input voltage	SEL0, SEL1, TEST0 (DIN)(2)	–0.3 to 6.5	V
		D+, D–, HID0, HID1, HID2, XTI, XTO, TEST1 (DOUT)(2), ~~SSPND~~	–0.3 to (V_{DDI} + 0.3) < 4	V
	Analog input voltage	$V_{IN}L$, $V_{IN}R$, V_{COM}, $V_{OUT}R$, $V_{OUT}L$	–0.3 to (V_{CCD} + 0.3) < 4	V
		V_{CCD}, V_{CCP1}, V_{CCP2}, V_{CCI}, V_{DDI}	–0.3 to 4	V
	Input current (any pins except supplies)		±10	mA
	Ambient temperature under bias		–40 to 125	C
T_{stg}	Storage temperature		–55 to 150	C
T_J	Junction temperature		150	C
	Lead temperature (soldering)		260	C, 5 s
	Package temperature (IR reflow, peak)		250	C

(1) Stresses beyond those listed under *absolute maximum ratings* may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under *recommended operating conditions* is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
(2) ( ): PCM2902

ELECTRICAL CHARACTERISTICS

all specifications at T_A = 25 C, V_{BUS} = 5 V, f_S = 44.1 kHz, f_{IN} = 1 kHz, 16-bit data, unless otherwise noted

PARAMETER		TEST CONDITIONS	PCM2900E, PCM2902E			UNIT
			MIN	TYP	MAX	
DIGITAL INPUT/OUTPUT						
Host interface		Apply USB Revision 1.1, full speed				
Audio data format		USB isochronous data format				
INPUT LOGIC						
V_{IH}	High-level input voltage	D+, D–		2	3.3	VDC
		XTI, HID0, HID1, and HID2		2.52	3.3	
		SEL0, SEL1		2	5.25	
		DIN, PCM2902		2.52	5.25	
V_{IL}	Low-level input voltage	D+, D–			0.8	VDC
		XTI, HID0, HID1, and HID2			0.9	
		SEL0, SEL1			0.8	
		DIN, PCM2902			0.9	
I_{IH}	High-level input voltage	D+, D–, XTI, SEL0, SEL1	V_{IN} = 3.3 V		±10	A
		HID0, HID1, and HID2		50	80	
		DIN, PCM2902		65	100	
I_{IL}	Low-level input voltage	D+, D–, XTI, SEL0, SEL1	V_{IN} = 0 V		±10	A
		HID0, HID1, and HID2			±10	
		DIN, PCM2902			±10	
OUTPUT LOGIC						
V_{OH}	High-level output voltage	D+, D–		2.8		VDC
		DOUT, PCM2902	I_{OH} = –4 mA	2.8		
		$\overline{SSPND}$	I_{OH} = –2 mA	2.8		
V_{OL}	Low-level output voltage	D+, D–			0.3	VDC
		DOUT, PCM2902	I_{OL} = 4 mA		0.5	
		$\overline{SSPND}$	I_{OL} = 2 mA		0.5	
CLOCK FREQUENCY						
Input clock frequency, XTI			11.994	12	12.008	MHz
ADC CHARACTERISTICS						
Resolution				8, 16		bits
Audio data channel				1, 2		channel

Appendice3

Altre funzioni

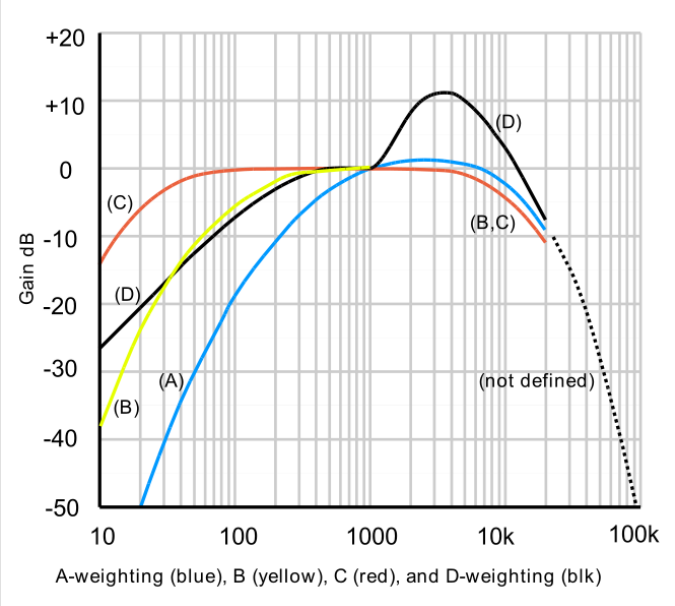
Visual Analyser presenta un consistente numero di funzioni sviluppate in corso d'opera che sono nate come ausiliarie alle principali funzioni oggetto dei precedenti capitoli. Nondimeno, si ritiene utile darne un breve cenno in questa appendice.

La funzione di compensazione

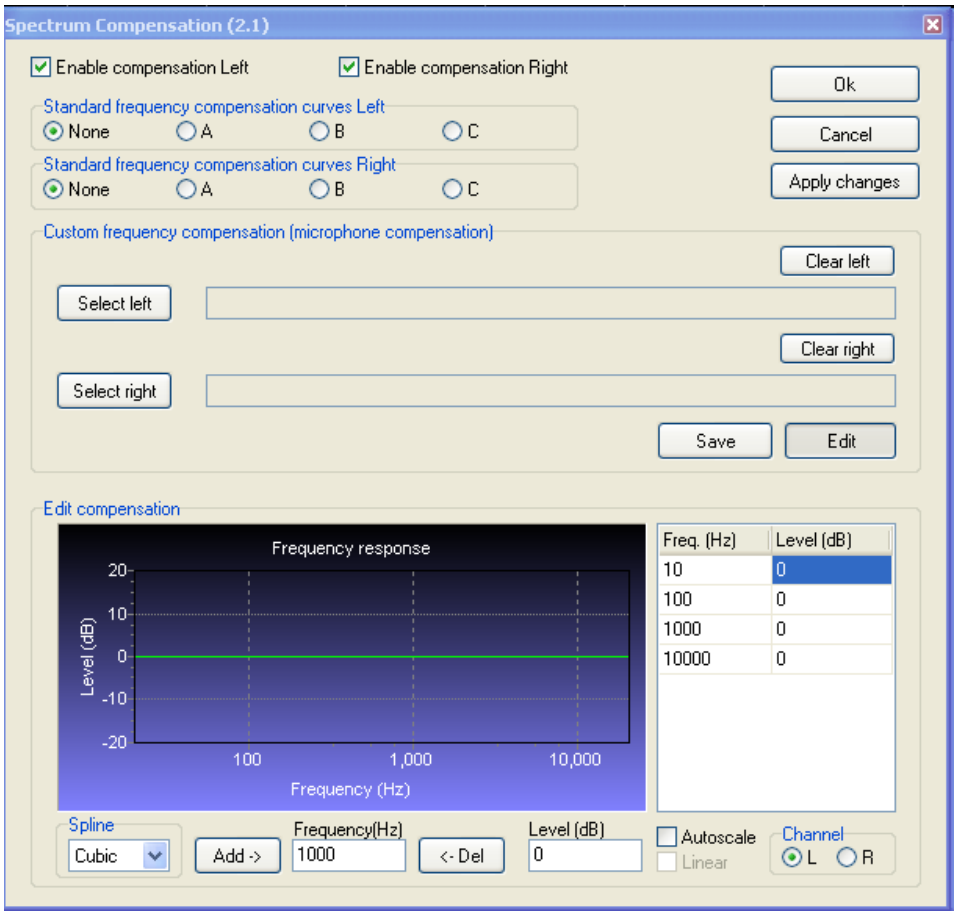
VA consente di impostare una curva di compensazione dello spettro secondo quanto stabilito dalle norme internazionali DIN. In particolare, esse fanno riferimento a particolari "curve di pesatura" dello spettro predefinite, che essenzialmente nascono per specifiche esigenze. Le più note sono le curve A, B, e C.

Esse consistono nell'applicazione di curve di risposta preconfezionate, che consentono di esaltare determinati range di frequenza ed attenuarne altri. Per esempio la curva di pesatura "A" consente di esaltare le frequenze comprese tra 3..6kHz regione in cui l'orecchio umano è più sensibile. L'applicazione di una pesatura "A" ad uno spettro consente di ottenere una evidenziazione dei livelli conforme con la sensibilità dell'orecchio umano.

Nella figura che segue viene riportato un grafico che mostra una nutrita serie di curve di pesature (A, B, C e D) . Visual Analyser consente l'applicazione delle prime tre (A,B,C) e di definire una curva di pesatura del tutto arbitraria.



La seguente figura mostra la finestra di impostazione delle curve di pesatura standard e arbitrarie:



Come è possibile osservare, si può assegnare una curva arbitraria introducendo i punti relativi ad alcune coppie frequenza/ampiezza note (per esempio ottenute dal "campionamento" di una curva continua) e applicare un algoritmo di interpolazione che ne ricavi una curva continua. Le curve di pesatura così ottenute possono essere salvate su file e "sovrapposte" alle curve standard.

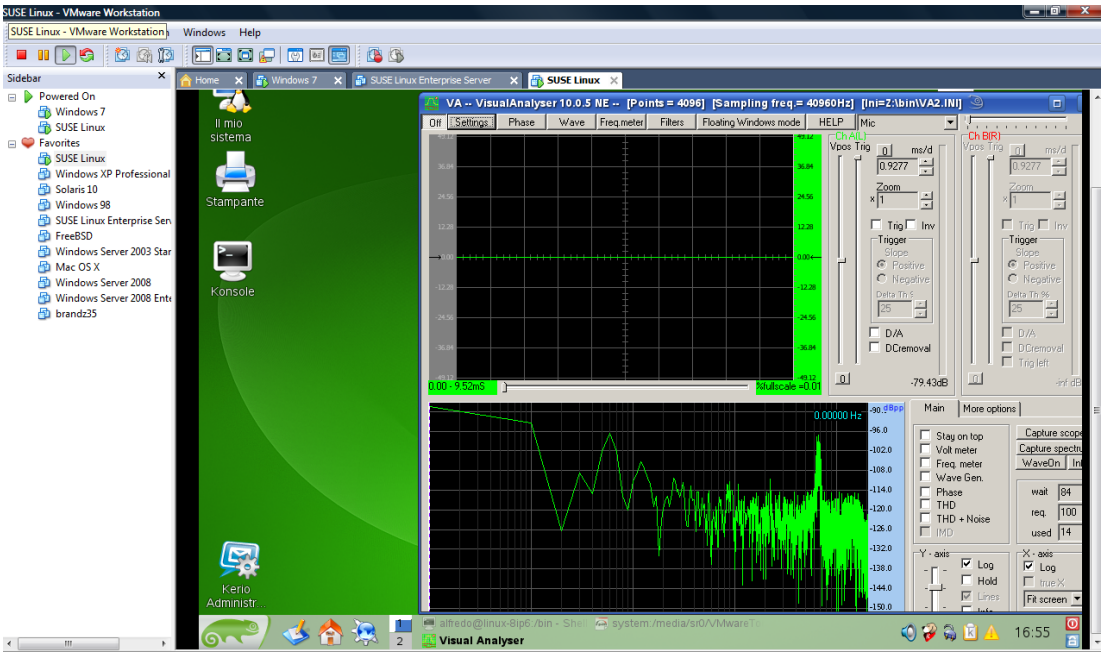
Sicchè, l'applicazione di una curva di risposta standard è utile in molteplici contesti, non ultimo quello inerente le misure: essa potrebbe per esempio essere usata per la compensazione di curve di riposta non volute (si pensi alla linearizzazione di risposte in frequenza di dispositivi hardware).

Sistemi operativi

Il programma è stato sviluppato con il compilatore Borland C++ nella sua versione più recente al momento delle stampe di questo lavoro; esso è dunque utilizzabile dalla quasi totalità dei sistemi operativi di casa Microsoft, esclusi quelli della famiglia 9X; invero per questi ultimi è possibile compilare una versione "ad-hoc" utilizzando una precedente versione del compilatore. Inoltre, per consentire un utilizzo su sistemi operativi Linux è possibile utilizzare con successo un pacchetto software chiamato "Wine" (www.winehq.org), il quale una volta installato su una qualsivoglia versione del sistema operativo Linux consente l'esecuzione di programmi Windows. Questo è reso possibile grazie ad un strato software che emula la maggior parte delle API di Windows traducendole in opportune chiamate alle API di Linux. Si noti che il livello prestazionale non risulta particolarmente degradato; prove "sul campo" hanno dimostrato un degrado non superiore al 10% (worst case).

La gestione ed il test di versioni del programma che possano girare su sistemi operativi differenti, implica naturalmente il collaudo delle stesse su macchine diverse oppure sulla stessa macchina dopo un nuovo "boot"; per rendere questo processo assai più snello e veloce è stato usato un virtualizzatore (VMWARE, www.vmware.com) che ha consentito di testare su di una stessa macchina, e

virtualmente in parallelo, la stessa versione del programma su diversi sistemi operativi (essenzialmente Windows e Linux).



Nella figura soprastante è possibile vedere l'esecuzione di VA sulla "distro" Suse di Linux in esecuzione su di una macchina Windows (nella quale a sua volta era in esecuzione la stessa versione di VA).

Conclusioni

Considerazioni finali

Devo constatare che mi è stato letteralmente impossibile descrivere esaustivamente tutti i risultati ottenuti in questi anni; risultati consistenti in centinaia di migliaia di linee di codice, circuiti fatti e rifatti milioni di volte, e un considerevole numero di notti insonni. Come risultato ho ottenuto funzioni, strumenti, utilità e procedure automatiche comprese in un "pacchetto" che da anni mi ostino a chiamare sempre con lo stesso nome: Visual Analyser. Quello che è stato descritto nei precedenti capitoli è dunque solo un veloce cenno, assai poco completo, di quanto effettivamente realizzato.

Oltre a quanto pianificato agli inizi del lavoro, nuove funzionalità, idee, miglioramenti e ritocchi sono stati aggiunti nel corso di questi tre anni sulla base delle più disparate esigenze, sia per scopi didattici che di ricerca. Di esse nemmeno mi riesce di tenere memoria. Spesso, mentre si seguiva una linea precisa, da essa improvvisamente se ne generava un'altra, dando vita a nuovi universi dotati di vita propria, che spesso hanno dato origine ad altri lavori effettivamente in corso d'opera, o potenziali, in attesa di sviluppo.

Infatti, un software "versatile" come Visual Analyser, le cui uniche limitazioni sono la fantasia di chi lo usa e lo modifica, inserito in un contesto dinamico come quello accademico, ed in particolare nel vasto e complesso mondo delle misure, è una miniera inesauribile di soluzioni e spunti per sempre nuovi e interessanti lavori.

Le idee e i concetti utilizzati in Visual Analyser sono frutto dell'interazione con molteplici persone, ognuna delle quali competente in una o più particolari materie; è impossibile per me ringraziarli e citarli tutti, ma devo dire che senza il loro costante "feedback" e prezioso aiuto pratico il programma ora non sarebbe a questo punto.

Allo stato attuale, che vede il programma essere diventato ben altro rispetto alla "base" da cui siamo partiti (il programma sviluppato per la tesi di laurea specialistica), si è giunti all'introduzione di un sistema di valutazione dell'incertezza, che apre la via verso nuove applicazioni. Sebbene la valutazione dell'incertezza sinora introdotta (prodotta sino alle stampe di questo volume, ma sicuramente più evoluta già da questa sera) sia ancora un primissimo passo, essa apre la strada verso tutta una serie di nuove applicazioni prima impensabili senza una valutazione anche minima di essa.

Visual Analyser non è solo un programma che "fa qualcosa" piuttosto un sistema sofisticato "aperto" e plasmabile, che consente di poter effettuare veloci ed efficaci adattamenti per le più disparate esigenze, che partono dalla didattica sino alla ricerca, e talvolta sconfinano nell'uso professionale. E tutto questo senza doversi appoggiare a librerie, pacchetti e sistemi di sviluppo di terze parti.

Il lavoro su Visual Analyser continua già da ora, momento in cui dichiaro finita la scrittura della tesi, la quale come detto non riesce, a causa del tempo tiranno, a descrivere tutti gli aspetti del lavoro effettivamente prodotto.

Le varie linee relative alle prossime azioni vedono in cantiere materie come "power quality", applicazioni mediche e di telemedicina, ulteriori strumenti di misura oltre alla possibilità, già in parte realizzata, di pilotare in maniera automatica sistemi per la determinazione di parametri di particolari sensori resistivi. Più molti altri che ometto per evitare che, ricordandoli a me stesso, abbia a passare i prossimi anni a trascurare famiglia e affetti più di quanto abbia dovuto fare sinora.

Bibliografia

1. A. Accattatis, tesi di laurea "*Sviluppo di uno strumento virtuale real-time per la generazione, analisi e acquisizione di segnali*", Università di Roma "Tor Vergata", facoltà di Ingegneria Informatica, relatore prof. Salvatore Tucci, Marcello Salmeri.
2. A. Accattatis, M. Salmeri, A. Mencattini, G. Rabottino, R. Lojacono, "*Visual Analyser: un laboratorio didattico di misure virtuali*", IMEKO TC4 Symposium (GMEE 20008), Monte Porzio Catone (RM), Italy, 7-10 Settembre 2008, atti del congresso, pp 317-318.
3. A. Accattatis, M. Salmeri, A. Mencattini, G. Rabottino, R. Lojacono, "*Visual Analyser: a Sophisticated Virtual Measurements Laboratory*", IMEKO TC4 Symposium (IMEKOTC4 '08), Firenze, Italy, September 2008.
4. A. Accattatis, "*Oscilloscopio e Analizzatore di Spettro su PC*", Elettronica In, pp. 37-44, December 200, traduzione in spagnolo dello stesso articolo su http://www.iberfutura.es3.
5. A. Accattatis, "*Visual Analyser: un programma Windows per la simulazione di strumenti di misura e generazione di forme d'onda*", Fare Elettronica, vol. 22, n. 258, pp. 74-82, November 2006.
6. A. Accattatis, "*Visual Analyser. La misura della risposta in frequenza di un amplificatore audio*", Fare Elettronica, vol. 23, n. 259, pp. 90-96, Gennaio 2007.
7. A. Accattatis, "*Elaborazione numerica dei segnali*", Fare Elettronica, vol. 24, n. 280, pp. 22-30, October 2008.
8. A. V. Oppenheim, R.W. Schafer "*Elaborazione numerica dei segnali*", Franco Angeli editore, 1988.
9. Steven W. Smith "*The scientist and Engineer's Guide to Digital Signal Processing*", Analog Device, http://www.dspguide.com.
10. Sen M. Kuo, Bob H. Lee, "Real-*Time digital Signal Processing*", John Wiley & Sons Ltd 2001.

11. C. Britton Rorabaugh, "*Digital Filter Designer's Handbook*", Mc Graw Hill 2000.
12. Sanjit K. Mitra, "*Digital Signal Processing, a computer based approach*", McGraw Hill 1998.
13. R. G. Lions, "*Understanding Digital Signal Processing*", Prentice Hall 2001.
14. A. S. Tanenbaum "*Modern Operating Systems*", 2nd edition ed. Prentice Hall 1998.
15. W. Stallings, "*Operating Systems: Internals and Design Principles*", 4th Ed., Prentice Hall, 2000.
16. J. Hollingworth, Bob Swart, Mark Cashman, Paul Gustavson "*Borland C++ Developer's guide*", ed. Sams, 2002.
17. C. Calvert, "*Borland C++ Builder Unleashed*", ed. Sams, 2002.
18. Sommerville "*Software Engineering 7*" ed. Addison Wesley, 2004.
19. A. Accattatis, "*Visual Analyser 8.0 for windows, a brief introduction*", http://www.electronicslab.com/downloads/pc/016/Visual_Analyzer/index.html.
20. Recensione di Visual Analyser http://matrixsynth.blogspot.com/2006/01/visual-analyser.html;
21. Versione tradotta in ceco di VA, http://cestiny.idnes.cz/pv/visualanalysercz.html;
22. "*Guide to the expression of uncertainty in measurement (GUM)*",1995-01-01. 105 pp, iso guide 98:1995 edition, 2005.
23. S. Caldara, S. Nuccio, C. Spataro, "*Measurement uncertainty estimation of a virtual instrument*", Proc. of Instrumentation and Measurement Technology Conference (IMTC 2000), Baltimore, MD, USA.
24. H. Haitjema, B. Van Dorp, M. Morel, P. H. J. Schellekens, "*Uncertainty estimation by the concept of virtual instruments*", Proceedings of SPIE, the International Society for Optical Engineering, 2001.
25. D. A. Lampasi, L. Podestà, "*A Practical Approach to Evaluate the Measurement Uncertainty of Virtual Instruments*", Proc. of Instrumentation and Measurement Technology Conference, Como, Italy, May 2004.
26. E. Ghiani, N. Locci, C. Muscas, "*Auto-Evaluation of the Uncertainty in Virtual Instruments*", IEEE Transactions on Instrumentation and Measurement, vol. 53, n. 3, June 2004.

27. M. J. Korczynski, A. Hetman, "*A Calculation of Uncertainties in Virtual Instrument*", Proc. of Instrumentation and Measurement Technology Conference, Ottawa, Canada, May 2005.
28. G. Betta, C. Liguori, A. Pietrosanto "*Propagation of uncertainty in a discrete Fourier transform algorithm*", Elsevier Measurement 27 (2000) 231-239.
29. R.I. Becker, N. Morrison, "*The errors in FFT estimation of the Fourier transform*", IEEE Transaction Signal Process. 44 (8) (1996) 2073-2077.
30. R. Petricca,Università degli studi di Cassino, tesi di laurea "*Stime dei parametri dei segnali nel dominio della frequenza in presenza di interferenza armonica*", relatore prof. Consolatina Liguori, AA 02/03.
31. R. Bristow-Johnson, "*Formulae for audio biquad filter coefficients*", rbj@audioimagination.com.
32. N. Locci, "*Dispense del corso di Misure Elettriche"*, Università di Cagliari, disponibile online" su www.diee.unica.it.
33. Lucidi delle lezioni sui convertitori A/D, lezione 21 www.dti.unimi.it.
34. Nicholas "Nick" Gray "*ABCs of ADCs*", Rev 2, August 2004, Copyright © 2003, 2004 National Semiconductor Corporation.
35. Wikipedia, "Dithering."
36. E. Ghiani, C. Muscas, "*Metodi numerici per la valutazione dell'incertezza nelle misure digitali*", Università degli studi di Cagliari, tesi di dottorato, XVI ciclo.
37. National Instrument, Application note n. 007 "*Data acquisition (DAQ) fundamentals*".
38. "*International vocabulary of metrology basic and general concepts and associated terms(vim)*", Technical report, 2008.
39. A. Mencattini, "*Appunti del corso di Elaborazione Segnali di misura*", Università di Roma "Tor Vergata", 2009.
40. Evaluation of measurement data, supplement 1 to the guide to the expression of un-certainty in measurement "*Propagation of distributions using a monte carlo method*", Technical report, 2008.

41. Rivista Nuova Elettronica, "*Impedenzimetro USB per personal computer*" n.24,2 pp.8..44, Dicembre 2009.
42. G. Steber, "*An LMS Impedance Bridge*", QEX, pp. 41..47 sept/oct 2005.
43. B. Widrow, S. D. Stearns, "*Adaptive signal Processing*", Prentice Hall, March 25, 1985.
44. M. Dutta, A. Rakshit, S. N. Bhattacharyya, "*Development and study of an Automatic AC Bridge for Impedance Measurement*", IEEE Transaction on Instrumentation and Measurement, Vol 50, N. 5, October 2001.
45. Rashid Ansari, "*IIR Discrete-Time Hilbert Transformers*", IEEE transactions on acoustics, speech, and signal processing, vol. assp-35, no. 8, August 1987.
46. A. Bertoni, G. Grossi, "*Elaborazione Numerica dei Segnali*", Dispense del corso, Università degli studi di Milano, 2009.
47. C. Carobbi, "*Appunti del corso di Misure Elettriche*", Università degli studi di Firenze, 2008.
48. J. W. Cooley, J. W. Tukey**:** "*An Algorithm for Machine Calculation of Complex Fourier Series*", Math. Computation, Vol. 19, 1965, pp. 297-301.