ALMA MATER STUDIORUM – UNIVERSITA' DI BOLOGNA
SEDE DI CESENA

SECONDA FACOLTA' DI INGEGNERIA CON SEDE A CESENA

CORSO DI LAUREA IN INGEGNERIA ELETTRONICA E TELECOMUNICAZIONI

TITOLO DELL'ELABORATO

Progetto di un'interfaccia utente per la gestione e memorizzazione di forme d'onda mediante microcontrollore

Elaborato in

Elettronica dei sistemi digitali

Relatore	Correlatore	Presentata da
Prof. Ing. Aldo Romani	Dott. Matteo Filippi	Matteo Zazzeri

2° Appello

Sessione III

Anno Accademico 2011/2012

Indice

Introduzione

L'energy harvesting è un processo in cui l'energia ambientale comunemente disponibile viene catturata mediante opportuni trasduttori e circuiti elettronici per essere convertita in energia elettrica utilizzabile.

Il progetto descritto sarà una estensione ed integrazione di un sistema già esistente, per la riproduzione attraverso un sistema elettrodinamico vibrante (shaker), di vibrazioni acquisite dall'ambiente circostante in situazioni di riferimento tipiche (esempio le vibrazioni prodotte da un veicolo in movimento o un uomo in corsa), al fine di caratterizzare trasduttori piezoelettrici per studiarne il funzionamento, le caratteristiche e il loro comportamento.

Lo scopo finale è quello di realizzare un sistema stand-alone che sia in grado di riprodurre e controllare in maniera affidabile le vibrazioni imposte da un sistema vibrante, al fine di realizzare un sistema di caratterizzazione per dispositivi di energy harvesting vibrazionale.

In questo progetto, l'intera gestione del processo viene affidata ad un microcontrollore presente sulla scheda di controllo, il quale consente in tempo reale la visualizzazione delle forme d'onda oggetto di studio mediante un display grafico, l'elaborazione dei dati presenti nel sistema nonché la possibilità di caricare e salvare dei dati significativi sulla memoria del sistema durante le fasi di testing. Le caratteristiche implementate rendono il sistema facile da usare.

Successivamente verranno descritte le specifiche tecniche necessariamente da rispettare per la realizzazione di un sistema che permetta di riprodurre e fornire dati attendibili, la struttura di visualizzazione grafica del sistema, la parte di condizionamento del segnale e i principi teorici del controllo ad anello chiuso.

Capitolo 1 - Generalità

1.1 - Energy harvesting

Il problemi di approvvigionamento energetico nella società moderna occupano un ruolo primario. Infatti nel processo dell'evoluzione tecnologica si è creato un netto squilibrio tra lo sviluppo dell'elettronica e lo sviluppo energetico

Attualmente la ricerca non si sta concentrando solo sullo sviluppo di sistemi più efficienti dal punto di vista energetico mantenendo o addirittura incrementando le prestazioni dei dispositivi elettronici ma anche su progetti di recupero e immagazzinamento di energia prodotta da fonti rinnovabili non convenzionali.

In soddisfazione a questa crescente necessità nasce il concetto di energy harvesting, ossia l'acquisizione di energia da sorgenti alternative disponibili nell'ambiente. Questo consentirebbe di eliminare o limitare la dipendenza da batterie elettrochimiche, e tutte le dirette conseguenze in termini ambientali e di costi di manutenzione e sostituzione.

Ad esempio una semplice passeggiata nel parco potrebbe essere ideale per immagazzinare quantità di energia minime che altrimenti sarebbero disperse, che possono poi essere utilizzate per alimentare dispositivi personali a basso consumo.
Gli esempi più comuni di sorgenti maggiormente sfruttate nell'ambito di Energy Harvesting sono:

- Sorgenti fotovoltaiche;
- Sorgenti piezoelettriche;
- Sorgenti RF (a radiofrequenza);
- Sorgenti termiche.

Il progetto in esame è posto al servizio di uno studio del comportamento dei trasduttori piezoelettrici, che permettono di trasdurre energia meccanica di vibrazione alle quali sono sottoposti in variazioni del campo elettrico.

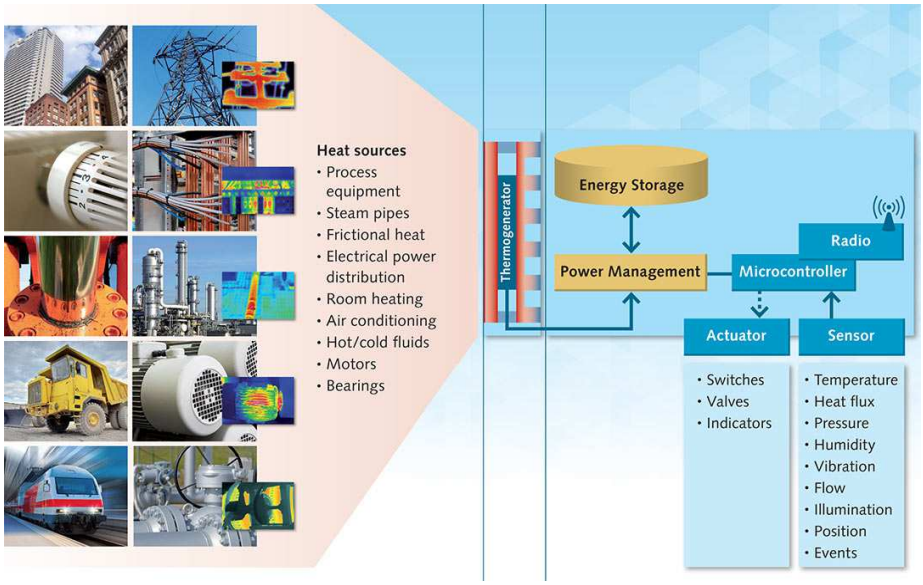

fig. 1.1: Applicazioni Energy Harvesting [1]

1.2 – Descrizione del progetto

Per lo sviluppo e lo studio di trasduttori piezoelettrici ci siamo serviti di uno shaker elettrodinamico. Le vibrazioni trasmesse dal suo movimento generano sui sensori piezoelettrici una potenza proporzionale all'accelerazione istantanea. L'accelerazione alla quale sottoponiamo il sistema deve essere completamente definibile dall'utente (nei limiti imposti dal sistema), per permetterne lo studio nelle più svariate condizioni.
Il progetto in questione *(fig.1.2)* prevede la possibilità di:

- selezionare una vasta gamma di forme d'onda prememorizzate all'interno di una memoria flash disponibile in una development board utilizzata per il progetto.

- mediante un collegamento tramite interfaccia USB ad un computer di impostare il valore di accelerazione efficace desiderato; il sistema provvederà di conseguenza a generare la forma d'onda che andrà a pilotare l'azionamento dello shaker con queste specifiche.

In tale elaborato non saranno trattate le modalità di comunicazione fra il microcontrollore e il PC, ma la sola sezione riguardante la programmazione del sistema di controllo che permetterà una interazione con l'utente in maniera semplice, garantendo comunque il rispetto delle specifiche.

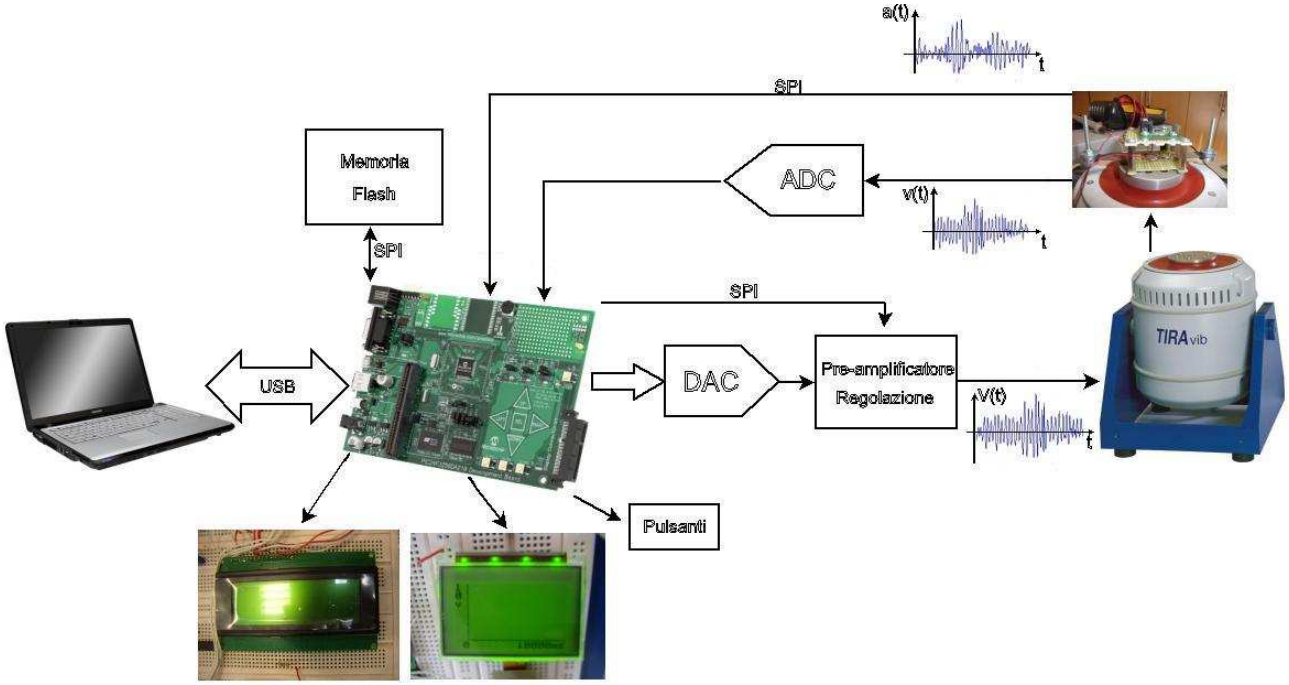

fig. 1.2: Schema a blocchi del progetto completo

Per meglio comprendere il sistema in *fig. 1.2* viene di seguito descritto il funzionamento tipico.

I campioni delle forme d'onda che dovranno essere riprodotte dal sistema, inizialmente sono memorizzati in un PC. Attraverso un'interfaccia USB tali forme d'onda vengono quindi salvate all'interno della memoria flash presente sulla development board del *PIC24FJ256DA210* . In questo modo i dati rimarranno memorizzati anche dopo lo spegnimento della scheda e non dovrà essere necessario il caricamento tramite PC ad ogni accensione. Il cuore del sistema è un microcontrollore a 16 bit disponibile sulla development board. Mediante l'ausilio di un convertitore DAC implementato esternamente alla scheda di sviluppo *(Cap. 2)* si convertono i campioni digitali in un segnale analogico la cui ampiezza viene attenuata/amplificata da un blocco preamplificatore a guadagno configurabile tramite interfaccia SPI. La configurazione del guadagno è affidata al microcontrollore che mediante un controllo a catena chiusa monitora costantemente l'accelerazione efficace tramite un accelerometro, anch'esso collegato mediante interfaccia SPI, e analizzando la differenza di potenziale sviluppata dai sensori piezoelettrici grazie ad un modulo di conversione ADC interno al PIC.

Le periferiche di visualizzazione, permettono la selezione e un controllo grafico durante le fasi di avanzamento della misura, nonché l'interazione con l'utente mediante un menù a scorrimento *(Cap.3)*. Si rende necessario l'utilizzo di pulsanti per permettere una selezione più agevole delle funzionalità di sistema.

Viene inoltre gestita la comunicazione con una memoria non volatile per permettere il caricamento e il salvataggio delle informazioni *(Cap. 4)*.

1.3 – Specifiche di progetto

Durante la fase di progettazione è stato necessario definire alcuni vincoli per garantire un funzionamento migliore possibile fra i diversi componenti, si è quindi selezionato il microcontrollore, i componenti di controllo e misurazione, i display, che verranno pilotati tramite un software per permettere il rispetto delle specifiche.

1.3.1 - Hardware

Durante l'implementazione di un sistema la scelta dell'hardware è uno dei punti cardine per permettere poi il corretto funzionamento di interazione fra le parti. Come si nota in *(fig. 1.3)* il corpo centrale del progetto è il microcontrollore, che deve quindi essere scelto con particolare cura.
La caratteristica di flessibilità del sistema deve permettere di riprodurre forme d'onda per un tempo massimo pari a 10 secondi, nonché il controllo dell'accelerazione media.
Queste caratteristiche richiedono da un punto di vista computazionale una elevata velocità di elaborazione e una quantità di RAM non disponibile, ad esempio, nei tradizionali PIC a 8 bit. Si è infine deciso di affidarsi alla famiglia a 16 bit, in linea con i requisiti richiesti.
La scelta è ricaduta su una development board basata su PIC24FJ256DA210 *(fig.1.4)*. La scheda come viene meglio definito in [5], rende disponibile un microcontrollore che garantisce elevate prestazioni computazionali grazie alla CPU a 16 bit e una frequenza di lavoro a 32 MHz, l'associazione alle caratteristiche di una memoria RAM interna da 96KByte lo rendono ideale per lo sviluppo di questo sistema, considerando quindi la quantità di dati che dovrà essere elaborate per garantire un funzionamento a tempo reale. Inoltre il controllore permette l'implementazione di funzionalità utili allo sviluppo del progetto, come la conversione analogico- digitale ad elevata velocità a 10 bit sfruttando ingressi dedicati, la generazione di interrupt capture mediante timer e ben 3 porte che supportano le configurazioni mediante interfaccia SPI. Comunque grazie alla possibilità di effettuare un Peripheral pin select (PPS) e al package a 100 pin il microcontrollore lascia all'utente un elevato numero di porte libere per il collegamento delle rimanenti periferiche di sistema come gli apparati di visualizzazione e generazione descritti in questa tesi.

Oltre al microcontrollore, sono rese disponibili direttamente sulla development board alcune periferiche utili per l'implementazione progetto e del sistema descritto in questa tesi come:

- Pulsanti programmabili e direttamente connessi al microcontrollore;
- Memoria Flash con una memoria interna di 16 Mbit collegata mediante interfaccia SPI;
- Connettori USB di tipo A, mini-B e micro USB.

fig. 1.3: Scheda di sviluppo Development Board PIC24FJ256DA210 [2]

Le varie caratteristiche che permettono al PIC24FJ256DA210 il collegamento con le periferiche saranno discusse singolarmente nel corso dell'elaborato.

1.3.2 - Software

In unione con l'hardware, vengono stabilite alcune caratteristiche fondamentali per poter realizzare il programma implementato nel PIC.

Innanzi tutto viene stabilito il massimo numero di campioni riproducibili per una forma d'onda, pari a 10000 elementi, viene quindi inizializzato nella RAM del controllore un buffer apposito, dove sarà caricata la forma d'onda in elaborazione. Vengono creati a supporto del primo altri 2 buffer uguali in dimensioni che permettono il salvataggio dell'andamento temporale delle grandezze di controllo del sistema (accelerazione istantanea e livello di tensione di ingresso del sensore piezoelettrico) . Le dimensioni del buffer allocate saranno volutamente superiori a quelle minime necessarie (pari a 20Kbyte, 2 byte per campione), si è infatti deciso di assegnare ai segnali memorizzati una più ampia disponibilità di memorizzazione nonché due campi fondamentali (nome e lunghezza) per facilitare la selezione da parte dell'utente .

Il progetto prevede processi multipli di acquisizione ed elaborazione dati. Questi sono necessariamente da eseguire secondo un ordine prestabilito, si procede prima alla conversione digitale analogica del campione in esame, questo opportunamente adattato sollecita lo shaker, le vibrazioni prodotte sono catturate da sensori piezoelettrici e ci vengono fornite sotto forma di tensione analogica, questa sarà acquisita dal modulo di conversione A/D del pic, per consentire l'elaborazione dell'informazione. Durante queste fasi si deve comunque garantire la visualizzazione dell'andamento della caratteristica sul display grafico e la possibilità di salvarne l'andamento. Quindi risulta necessario che il sistema completi l'intera sequenza prima che sia attivato il processo di elaborazione su un nuovo campione. Per garantire un flusso regolare di informazioni, si è fatto in modo che tutti i processi avvengano con una cadenza prestabilita: con frequenza di acquisizione/campionamento pari a 1 kHz, questo permette la definizione dell'intervallo temporale entro il quale l'intero processo di elaborazione di un campione deve essere completato.

Per definire la finestra temporale si è dovuto ricorrere ad un Timer, il microcontrollore dispone al suo interno di cinque timer programmabili a 16 bit [4], di questi si è selezionato il Timer1. Il contatore una volta avviato il conteggio mediante il set del bit TON nel registro T1CON provvede in maniera automatica all'incremento del valore preconfigurato nel campo TMR1, con frequenza pari a 16 MHz. La scelta della frequenza di incremento corrisponde al tempo impiegato dal microcontrollore a compiere una istruzione, cioè due colpi di clock.

Conoscendo il tempo impiegato dal controllore per incrementare di una unità il registro TMR1 pari a 62.5 ns si può dedurre di conseguenza quanti incrementi deve compiere il contatore per garantire il periodo di 1 ms, questo è pari a 16000.

Per permettere la generazione di un interrupt il registro a 16 bit di Timer1 dovrà andare in overflow, quindi per permettere la definizione di un'intervallo temporale di 1 ms si dovrà provvedere a sottrarre da 65536 pari a 2^{16} il valore di offset calcolato pari a 16000. Ottenendo 49536, il valore con cui il timer inizierà il conteggio, questo dovrà essere caricato nel campo del registro TMR1.

Possono essere definite ulteriori livelli di configurazione del timer mediante il setting di bit dedicati in T1CON [4]. Questi consentono la sincronizzazione e un prescaler del clock ma tali impostazioni non si reputano necessarie nella realizzazione di questo progetto, quindi il setting del campo TMR1 rimane l'unica specifica rilevante nella configurazione di Timer1.

L' implementazione rende possibile la gestione dell'avvio del processo di generazione/ acquisizioni ogni 1 ms mediante il set di un interrupt Flag, generato automaticamente al termine di ogni ciclo di conteggio del timer, questo permette al controllore di concentrarsi esclusivamente sull'elaborazione delle informazioni, delegando l'avvio delle acquisizioni/generazioni del segnale. Solitamente la gestione degli interrupt non risulta subito semplice ma nella famiglia dei pic a 16 bit compare la possibilità di identificare ogni singolo interrupt generato mediante l'uso di vettori univoci contenuti in una Interrupt Vector Table(IVT) [3], questa caratteristica si unisce alla possibilità di impostare fino a otto livelli di priorità per ogni interrupt mediante macro dedicate nel registro di stato (SR) [3], infatti data la particolare importanza che riveste avvio del processo di generazione ed acquisizione comandato dal Timer1 si è deciso di impostare il massimo livello di priorità pari ad uno.

1.3.3 - Schema circuitale

Lo schema in fig. 1.4 rappresenta i collegamenti effettuati per lo sviluppo del sistema descritto in questa tesi. Questo sarà poi integrato con lo schema circuitale di un progetto sviluppato parallelamente, meglio descritto in [15].

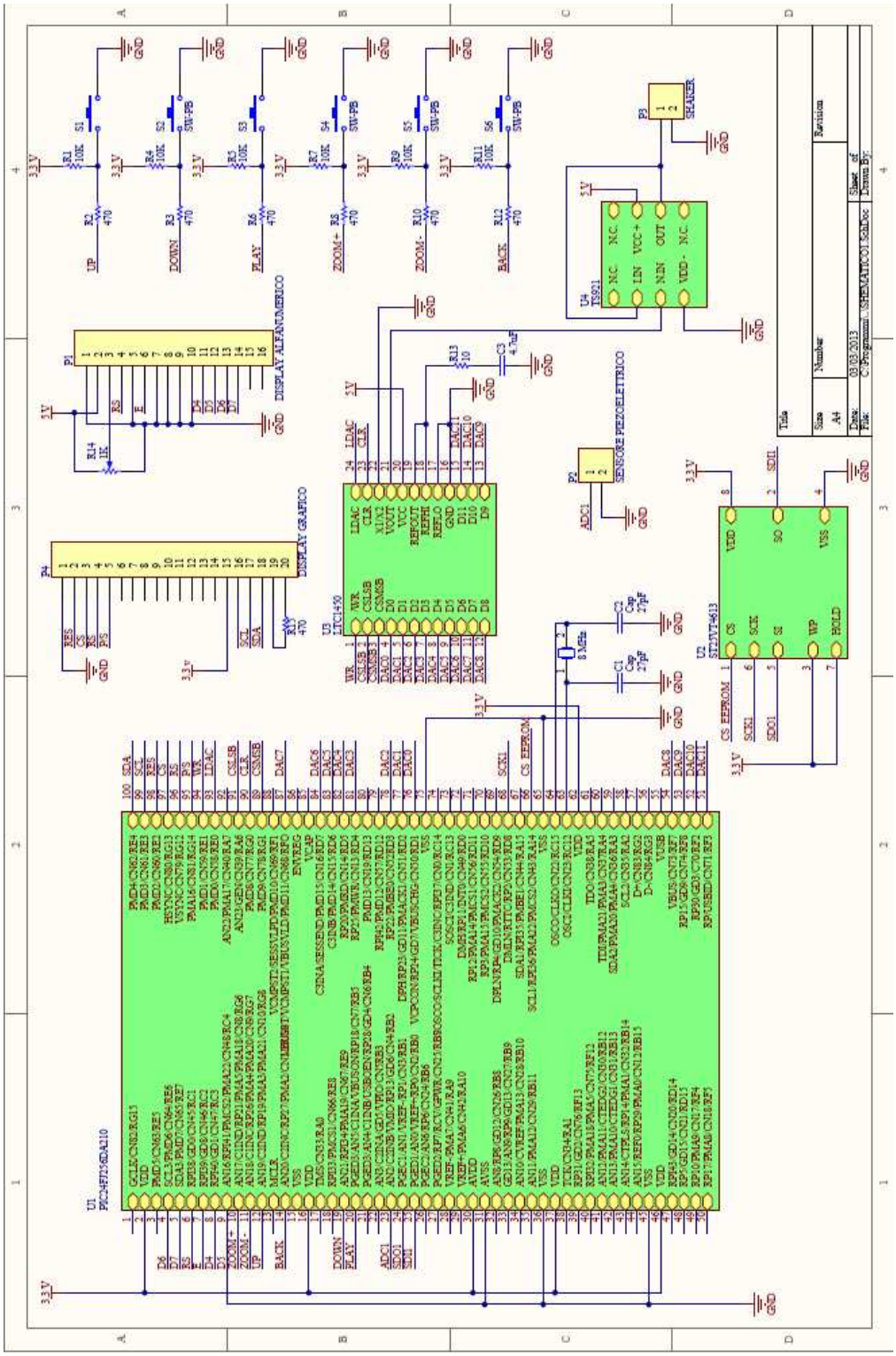

fig. 1.4 Schema elettrico del progetto. Per pin non connessi sono rispettate le configurazioni presenti sul datasheet [5].

Capitolo 2 Conversione dei segnali

Nei sistemi elettronici ad esempio nell'elaborazione e nella trasmissione dei segnali, è necessario convertire i segnali stessi nella forma più opportuna; questa funzione è svolta dai convertitori e in questa sezione sono esaminate le modalità di conversione digitale analogica e analogico digitale.

2.1 DAC

I convertitori digitali analogici, indicati comunemente come DAC (Digital to Analog Converter), sono dispositivi che convertono un codice digitale in una tensione o corrente ad esso proporzionale, permettendo l'interfacciamento del dispositivo con il mondo reale. I DAC si possono differenziare in due macrocategorie :

- Bipolari;
- Unipolari.

Esistono quindi convertitori bipolari caratterizzati da un intervallo simmetrico dei valori assumibili in ingresso rispetto all'asse delle ascisse, il che permette di avere un segnale che può assumere sia valori positivi che negativi. In contrapposizione troviamo i convertitori unipolari, dispositivi la cui tensione di uscita può assumere valori solo positivi.

Nell'implementazione del progetto sarà utilizzato un DAC unipolare, pur essendo infatti possibile fornire allo shaker una forma d'onda che abbia una escursione massima di 5V, indipendente dal suo valore medio; ci si è comunque indirizzati per semplicità ad un andamento 0-5V che permette l'uso di un riferimento fisso (GND). Una ulteriore alternativa, che è stata scartata, consisteva nello sfruttare un dac bipolare e un successivo incremento di un offset del segnale utile. Si sono infine analizzate le specifiche dello stadio di amplificazione dello shaker; questo ha permesso di dedurre la necessità di un segnale di condizionamento in tensione.

Alla luce delle scelte effettuate ci si riferirà successivamente ai soli convertitori DAC unipolari con uscita in tensione.

La tensione all'uscita del convertitore (V_n) è esprimibile in funzione della configurazione binaria del dato in ingrasso e della tensione di fondo scala (V_{FS}) mediante una somma pesata dei soli termini (B_n) che assumono un livello logico alto al momento dell'elaborazione, definibile dall'equazione:

$$V_n = \sum_{i=0}^{n-1} V_{FS} \cdot \frac{B_r}{2_{M-i}} = V_{FS} \cdot \left(\frac{B_{LSB}}{2^n} + \frac{B_1}{2^{n-1}} + ... + \frac{B_{n-1}}{4} + \frac{B_{MSB}}{2} \right)$$ *Eq. 2.1*

Solitamente i convertitori, richiedono due tensioni di riferimento, una positiva (V_{Ref+}) ed una negativa (V_{Ref-}). Ma nel caso di convertitori unipolari è sfruttata la sola tensione di riferimento positiva, dato che il riferimento negativo corrisponde al GND del sistema.

Volendo indicare con *m* la risoluzione in bit dell'DAC, si associa quindi al bit meno significativo *(Vn_{lsb})* un valore di tensione equivalente a :

$$Vn_{lsb} = \frac{V_{ref+}}{2^m}$$ *Eq. 2.2*

Le configurazioni di base più comuni dei convertitori D/A sono:

- Convertitori DAC a resistori pesati;
- Convertitori DAC con Rete a scala.

Le tipologie differiscono tra loro dal tipo di rete resistiva utilizzata. La tipologia a resistori pesati viene utilizzata in convertitori con bassa risoluzione si opta quindi in favore della tipologia con rete a scala, che permette risoluzioni maggiori, come richiesto in questo progetto.

2.1.1 DAC con rete a scala

I convertitori con reti a scala (ladder network), detti anche convertitori R- 2R sono così denominati in conseguenza della struttura della rete di resistenze unite tra loro da un rapporto 2 (fig.2.1).
Ogni bit d'ingresso pilota la commutazione del deviatore corrispondente. Quando il generico bit è a livello logico basso il corrispondente deviatore è connesso a massa, mentre quando è posto a livello logico alto è collegato all'ingresso invertente. La rete a scala è strutturata in modo tale che la resistenza equivalente offerta dalla rete alla sinistra di ciascuno dei suoi nodi valga R. Questa caratteristica può essere verificata considerando il fatto che ogni resistenza 2R è inserita nella rete con un estremo collegato al corrispondente nodo della stessa rete e con l'altro estremo a massa. Consideriamo, quindi, il nodo N_0. La resistenza equivalente è 2R//2R = R. Tale resistenza risulta in serie con la resistenza R collegata al nodo N_1, fornendo un equivalente pari alla serie R + R = 2R. Di conseguenza essa risulta in parallelo all'altra resistenza 2R. L'equivalente di tale parallelo è ancora R. Questa struttura si ripresenta identica in corrispondenza di ogni nodo.

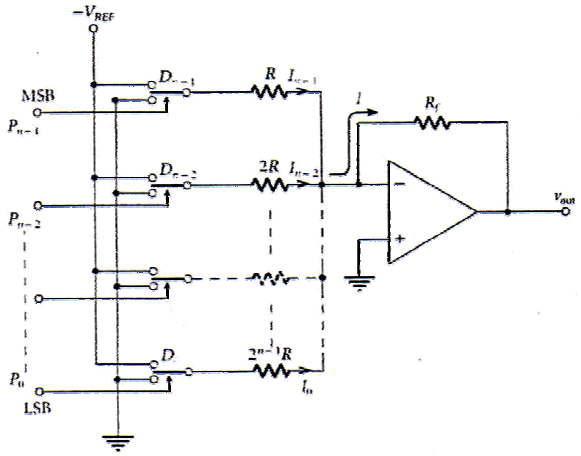

fig. 2.1: DAC con rete a scala *(fonte: [9])*

La tipica funzione di trasferimento di un DAC ideale a 4 bit è rappresentata in *fig. 2.2*. Da esso si deduce come il componente determini una corrispondenza univoca tra l'ingresso digitale e la tensione d'uscita, questa assumerà un numero preciso di valori, multiplo intero di Vn_{lsb} *(Eq. 2.2).*

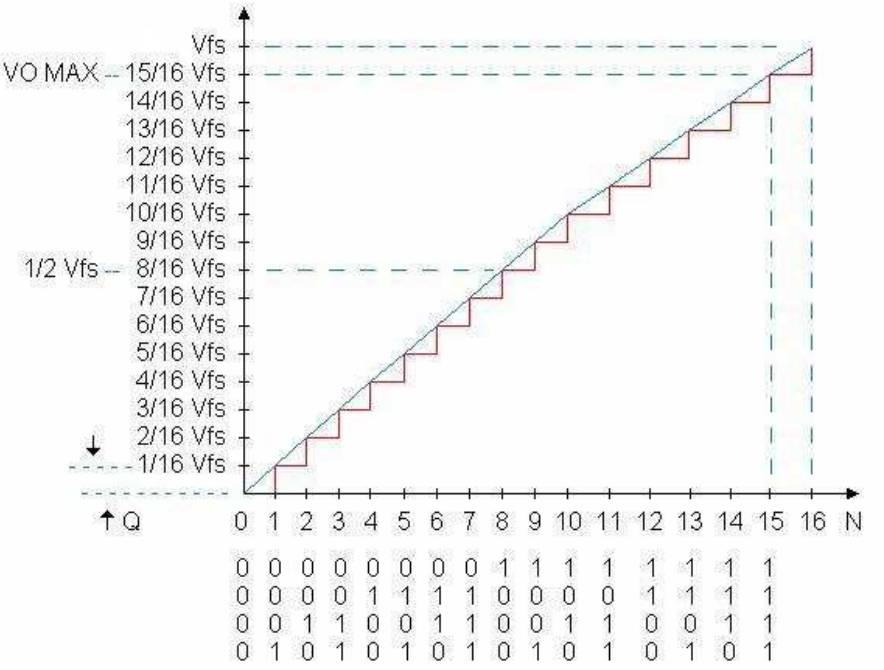

fig. 2.2: funzione di trasferimento ideale di un DAC a 4 bit (fonte: [9])

Parametri caratteristici

Durante la scelta del convertitore D/A adatto, è opportuno conoscere attentamente i parametri caratteristici, fra i quali compaiono la risoluzione e la grandezza d'uscita già precedentemente illustrati nei paragrafi precedenti.

Ma questi non sono sufficienti per definire a pieno la qualità di un componente è necessario considerare anche la differenza tra il valore desiderato e quello in uscita al convertitore, meglio noto come accuratezza del convertitore

Questo parametro viene definito attraverso le principali sorgenti di errore che è possibile riscontrare, come:

- *Errore di Non Linearità,*

  Tale errore è conseguenza diretta della non linearità della caratteristica di trasferimento del componente. Viene quindi compromesso il passo della conversione che non rispetta più le distanze costanti fra due livelli di discretizzazione del segnale. È necessario tenere in assoluta considerazione questa sorgente di errore, che può alterare la monotonia della caratteristica I/O del convertitore.

- *Errore di Riferimento*

  Rappresenta la dispersione del valore effettivo della tensione di riferimento nell'intorno del valore nominale, viene generalmente espresso in funzione della temperatura e della tensione;

- *Errore di Amplificazione di uscita*

  Sempre più frequentemente i convertitori A/D hanno uno stadio di preamplificazione di uscita implementato mediante un'amplificatore operazionale, tale componente può però introdurre ulteriori errori dovuti alla non idealità.;

- *Rapporto segnale - rumore*

  Nel suo acronimo ingelse (SNR, Signal to Noise Ratio), il rumore sovrapponendosi al segnale utile è fonte di un disturbo indesiderato che ne corrompe

  il contenuto informativo, questo rapporto indica quindi l'immunità ai disturbi del nostro componente.

L'incertezza con la quale avviene un processo di misurazione risulta molto importante nello sviluppo del progetto, sarà nostra premura selezionare un componente che garantisca quindi una elevata immunità.

Infine si è scelto di non servirsi di un DAC ad interfaccia di ingresso seriale, perché i caratteristici tempi di latenza nella trasmissione sequenziale del dato avrebbero potuto limitare la rapidità del sistema nella generazione del segnale; si è quindi optato per un' interfaccia di ingresso parallela *(Cap. 2.1.2).*

2.1.2 LTC1450

L'LTC1450 è un convertitore digitale – analogico a 12 bit basato sulla tecnologia con rete a scala, garantisce inoltre elevati standard di risoluzione grazie alla precisione della sua implementazione, che gli permette una maggiore immunità dalle principali sorgenti di errore sempre presenti nei processi di misura. Inoltre il DAC comprende uno stadio rail to rail in uscita ad amplificatore operazionale a guadagno configurabile (1 o 2) ed un doppio latch in ingresso *(fig. 2.3)*, il tutto sfrutta una alimentazione singola.

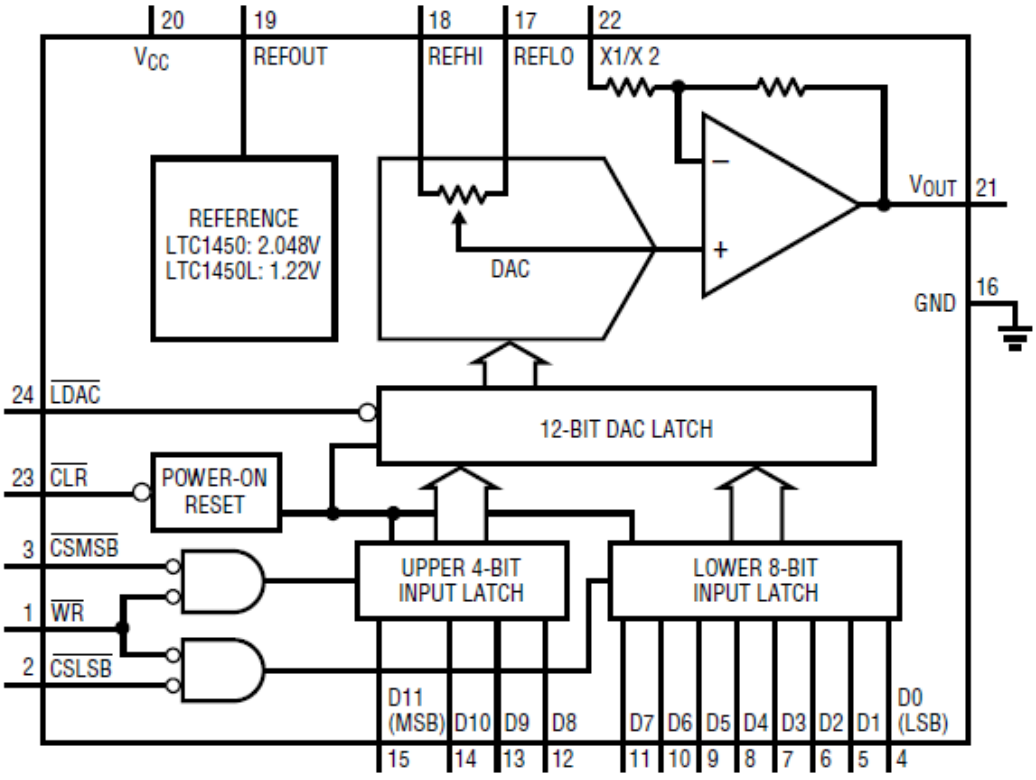

fig.2.3: Schema a blocchi configurazione interna LTC1450 (fonte: datasheet LTC1450 [8])

Interfaccia parallela

Il dato viene impostato all'ingresso del convertitore dal microcontrollore, mediante 12 bit da D0 - D11, si passa poi alla fase di pre-elaborazione in cui i primi 8 bit meno significativi del dato vengono caricati all'interno dei registri d'ingresso del DAC, mediante impostazione del CSLSB e il WR a livello logico basso *(fig. 2.4)*. Il dato viene considerato valido, quindi campionato e caricato all'interno dei registri in corrispondenza del fronte di salita del WR, si pone poi nuovamente il CSLSB a livello logico alto, Avviene poi una sequenza operativa analoga con i rimanenti MSB, che avviene sfruttando il CSMSB.

Al termine della procedura di caricamento il dato viene immesso in registri denominati DAC Latch. In corrispondenza del fronte di salita del segnale LDAC, il quale viene mantenuto a livello logico basso per un tempo minimo definito dal datasheet, il dato convertito viene reso disponibile in uscita *(fig. 2.4)*.

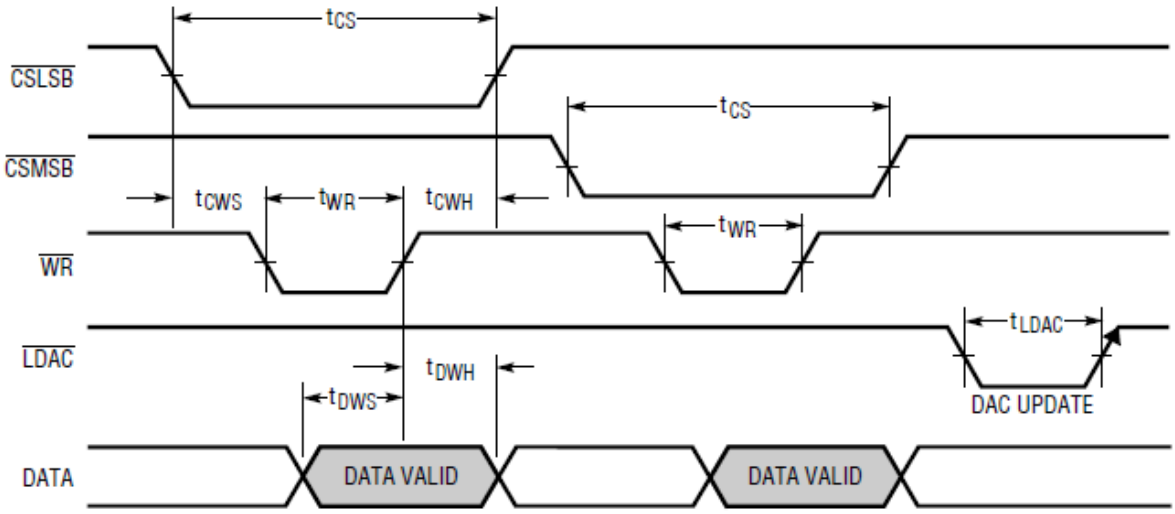

fig. 2.4: Diagramma temporale LTC1450 (fonte: datasheet LTC1450 [8])

Riferimento di tensione

Alla procedura di conversione si affianca un'ottima flessibilità del componente in relazione alle diverse possibilità di impiego, ciò è reso possibile dalla tensione di riferimento interna a 2.048 V, che può permettere al componente valori di tensione di fondoscala pari al doppio (4.096 V), nel caso in cui si sia configurato a 2 il guadagno di uscita dell'amplificatore operazionale tramite un collegamento esterno. Tale configurazione viene implementata anche nel progetto illustrato, permettendo quindi una escursione in uscita del segnale fino a 4.096 V.

Per aumentare l'immunità ai disturbi del riferimento è stato necessario collegare un condensatore di bypass, scelta consigliata anche sul datasheet *[8]*.

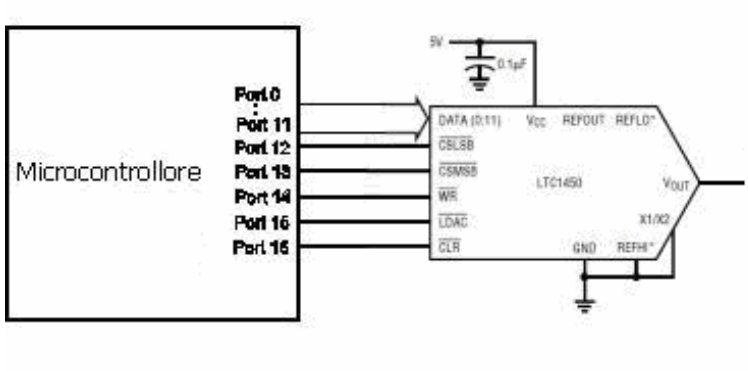

fig. 2.5: Collegamento LTC1450 con microcontrollore

Algoritmo di conversione

Il processo di conversione D/A, viene avviato mediante un interrupt dedicato generato direttamente dal Timer del microcontrollore con frequenza di 1 KHz.

L'algoritmo di conversione elaborato carica i 12 bit del campione sui pin paralleli di ingresso del convertitore ogni millisecondo. Viene quindi attivato il processo di conversione del campione digitale in analogico mediante il codice:

```
void DAC(int Value)
{
// Byte meno significativo
      CSLSB = 0;
      Nop();
      WR = 0;
      Data(da byte7 a byte0)= Value & 0x00FF;
      Nop();
      Nop();
      WR = 1;
      Nop();
      CSLSB = 1;
      Nop();
// Byte più significativo
      CSMSB = 0;
      Nop();
      WR = 0;
      Data(da byte11 a byte8)= Value & 0x0F00;
      Nop();
      Nop();
```

```
      WR = 1;
      Nop();
      CSMSB = 1;
           Nop();
 // conversion Analogic DAC
      LDAC = 0;
      Nop();
      Nop();
      LDAC = 1;
}
```

2.2 ADC

Ogniqualvolta si voglia elaborare un segnale di natura analogica mediante componenti in grado di elaborare solo grandezze digitali, è necessario sottoporre le grandezze analogiche ad un processo di conversione analogico-digitale, per renderla possibile si sfrutta un convertitore analogico digitale (A/D); questo step è necessario, infatti in uscita si può visualizzare un codice composto da più bit, che assumono un valore caratteristico in funzione dell'ingresso analogico.

Una buona sintesi di questi contenuti è riportata in [9], e viene di seguito riassunta e riproposta.
Il convertitore A/D (ADC, Analog to Digital Converter) ha un funzionamento duale rispetto al suo componente DAC visto nel *Cap. 2.1*.
Al fine di eseguire una corretta conversione del segnale ci si avvale di due passi fondamentali:

- *Quantizzazione;*
- *Codifica binaria.*

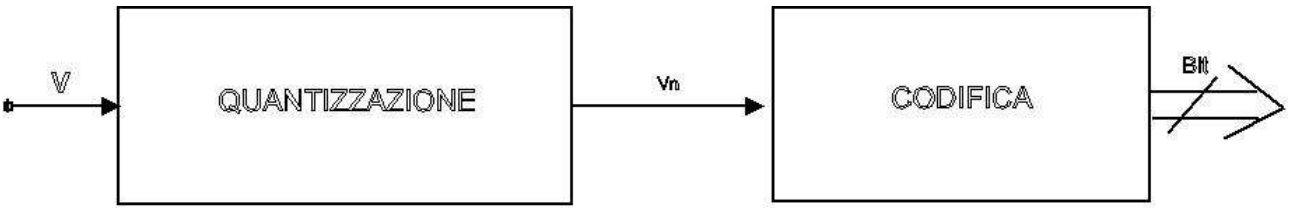

fig. 2.6: Blocchi Fondamentali del processo conversione A/D

Il segnale analogico subisce quindi un'operazione di quantizzazione; essa consiste nel sostituire una grandezza analogica, che può variare con continuità, con una grandezza quantizzata che può assumere una serie discreta di valori.

Si desidera considerare in generale un convertitore ad n bit, Dove possono essere assunti 2^n valori discreti e l'intervallo in cui la grandezza analogica può variare viene distinto nelle categorie bipolari e unipolari. Nel nostro caso i valori assunti sono solo positivi (da 0 al valore di Fondo Scala) e si suddivide quindi l'intervallo in 2^n parti dove il quanto fondamentale vale:

$$V_Q = \frac{V_{FS}}{2^n} \qquad \textit{Eq. 2.3}$$

Il valore corrispondente all'*Eq.2.3* equivale alla minima variazione riscontrabile sull'ingresso analogico definibile anche come risoluzione del convertitore

Nel processo di quantizzazione risulta implicita l'introduzione di un errore, detto rumore di quantizzazione, questo comporta una perdita di informazione proporzionale al passo di quantizzazione. Si può osservare che ad una risoluzione maggiore corrisponde la diminuzione dell'errore associato.

Viene poi associato un codice binario ad ogni livello di tensione in ingresso, il codice viene opportunamente costruito in base al tipo di segnale per rendere il minore possibile l'errore in ricezione di livelli adiacenti. A differenza del processo di quantizzazione la codifica è un processo completamente reversibile, mediante il processo duale offerto dal DAC.

2.2.1 Convertitori ad approssimazioni successive

Nel progetto in esame è stato scelto di realizzare il blocco A/D mediante uno dei convertitori ad approssimazioni successive interni al microcontrollore, al fine di non creare ulteriore ingombro aggiungendo componenti esterni. In seguito si farà riferimento a questa specifica categoria.

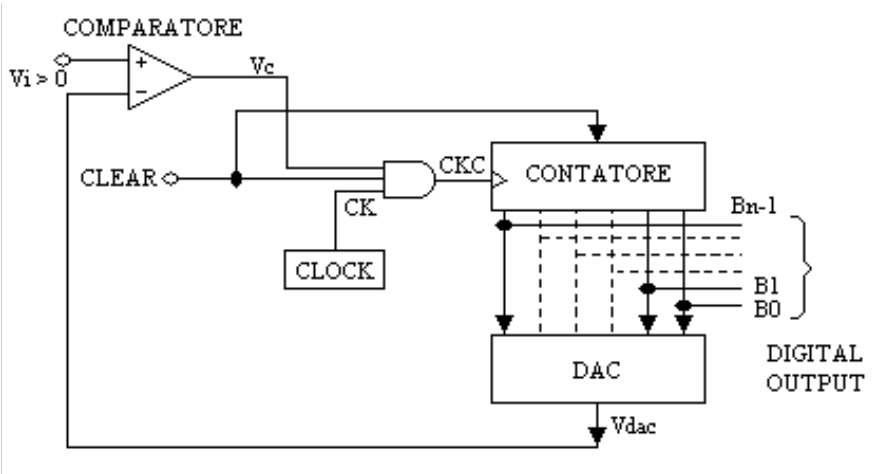

fig. 2.7 : Schema a blocchi di un convertitore ad approssimazioni successive (fonte: [9])

Nello schema a blocchi di *fig.2.7* viene rappresentato il convertitore analogico S.A.R.(Successive Approximation Register).
Il cuore del processo è l'algoritmo di conteggio, ne esistono vari tipi con caratteristiche diverse, ma il più efficiente si avvale di un contatore Johnson.

Risulta necessario conoscere cosa rende il contatore Johnson così performante ancora prima di descrivere i passi fondamentali che questo componente esegue durante il processo di conversione *(fig. 2.8)*,.

La caratteristica vincente del contatore garantisce il trasferimento in uscita di una cifra binaria a livello logico alto ad ogni impulso di clock; questo a partire dal MSB. Con l'ausilio di questa procedura il processo dura sempre N passi per un convertitore ad N bit, mentre con un generico convertitore a conteggio, nel caso peggiore il processo può durare anche 2^N passi.

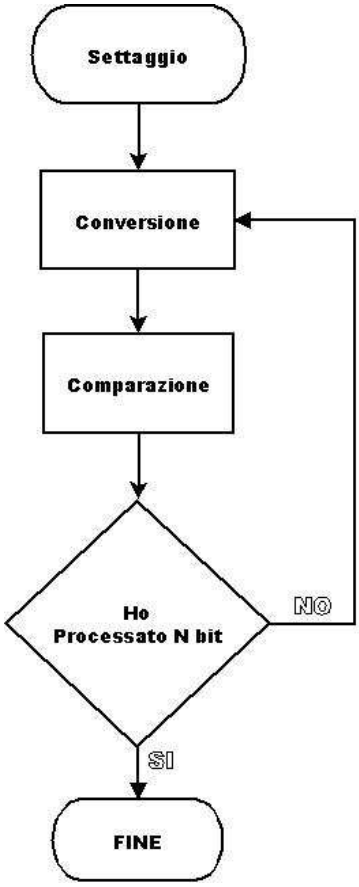

fig. 2.8: flow chart ADC S.A.R.

La corretta interpretazione del processo di funzionamento del convertitore deriva implicitamente dall'algoritmo implementato nella logica di controllo.
Il processo di conversione può essere suddiviso nei seguenti step fondamentali *(fig. 2.8)*:

- *Settaggio*

  Viene abilito il contatore Johnson ad N bit. Il bit più significativo (MSB) all'uscita del contatore viene settato in corrispondenza del primo impulso del clock in uscita al contatore, mentre i rimanenti N-1 bit permangono nella configurazione a livello logico basso;

- *Conversione*

  La configurazione digitale così definita viene fornita in ingresso al convertitore digitale-analogico, che per il tipo di implementazione ne fa corrispondere un'uscita analogica;

- *Comparazione*

  L'uscita analogica del DAC subisce un confronto mediante un comparatore con il valore in tensione del campione da convertire *(fig. 3.6)*.
  Se la tensione dal convertitore D/A risultasse inferiore a quella del campione, il bit più significativo verrebbe memorizzato dalla logica di controllo come livello logico alto. In caso contrario l'MSB sarebbe stato posto a livello logico basso;

- *Bit successivi al primo*

  Un secondo impulso di clock permette la transizione del bit N-1 allo stato logico alto, la nuova condizione si aggiungerà a quella memorizzata precedentemente all'uscita della logica di controllo.
  Il codice così ottenuto viene successivamente sottoposto ai processi di *conversione* e *comparazione*.
  Come si può facilmente intuire il processo sopra descritto si itera ciclicamente per i rimanenti N-2 bit.

Il risultato della conversione è univocamente definito solo se durante l'intero processo di conversione la tensione in ingresso resta costante. Per essere certi che la condizione sia rispettata, si utilizza spesso un circuito di sample and hold, posizionato a monte del convertitore. Inoltre è necessario, che prima dell'avvio di un nuovo confronto il dato abbia completato il processo di operazioni per garantire il controllo in retroazione, quindi, il periodo del clock deve risultare maggiore della somma del tempo di latenza del comparatore e dei tempi richiesti per assegnare l'uscita del DAC ; tali considerazioni limitano superiormente la frequenza del sistema.

2.2.2 Modulo ADC del PIC

Il PIC24FJ256DA210 come specificato in [10] dispone di 24 ingressi analogici, mediante opportuni selettori, configurabili internamente è possibile selezionare il segnale d'ingesso che verrà sottoposto al processo di conversione. Il convertitore A/D interno ha una risoluzione di 10 bit, questa sua caratteristica non compromette la funzionalità del nostro progetto in quanto il campionamento determina 1024 livelli di discretizzazione più che sufficienti per una misura apprezzabile dei segali in esame. Nonostante sia possibile sfruttare un riferimento di tensione esterno, fornito mediante l'aggiunta di due pin analogici ,non è in genere possibile il collegamento diretto fra un microcontrollore e un sensore piezoelettrico. Perchè quest'ultimo può presentare ai sui terminali tensioni nettamente superiori alla tensione massima ammissibile in ingresso. Si ritiene quindi necessario l'implementazione di una rete di condizionamento del segnale proveniente dai sensori piezoelettrici.

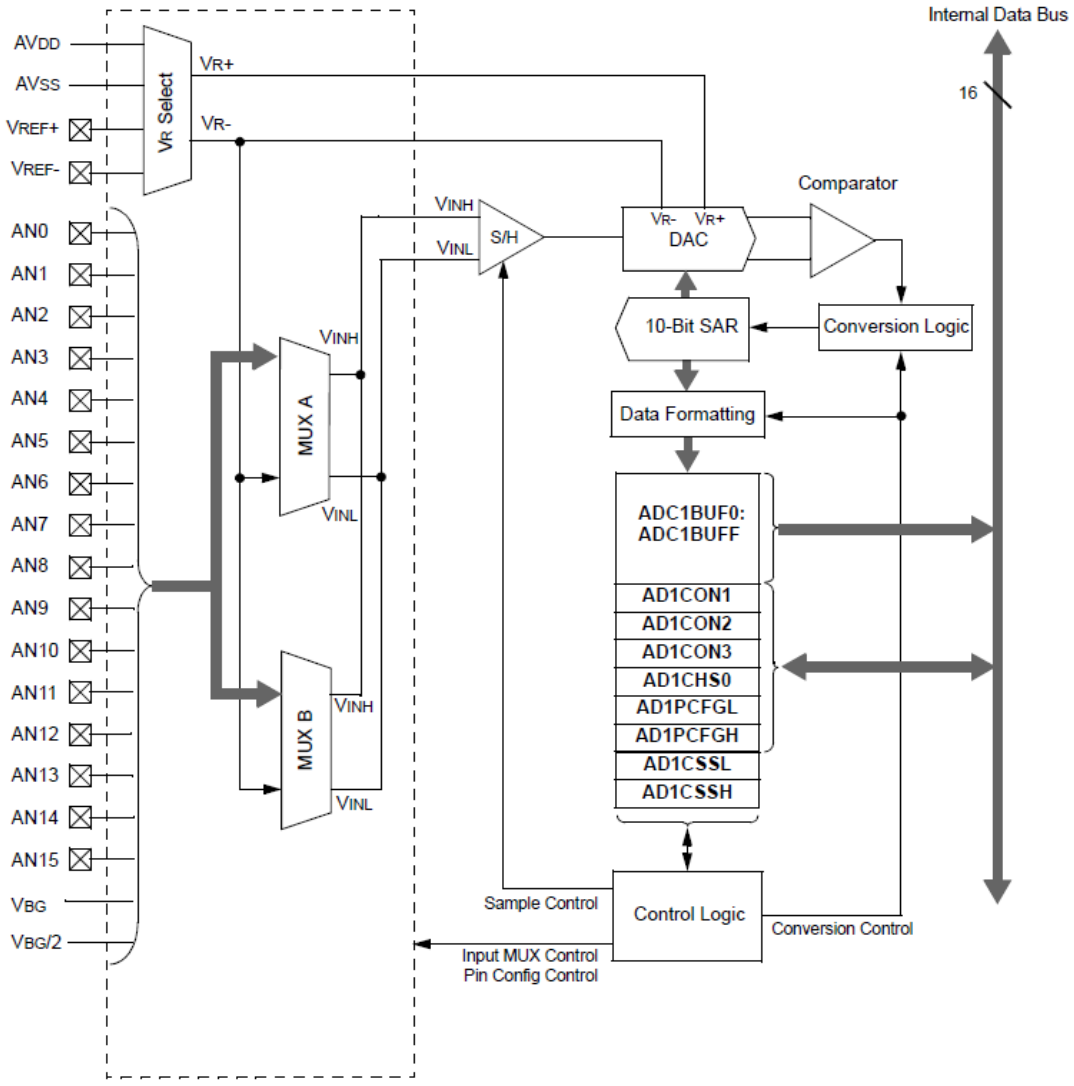

fig. 2.9: PIC16FJ256DA210 Modulo ADC interno (fonte: [10])

Nel modulo ADC del PIC24F utilizzato nel progetto è presente un primo blocco, identificato dalla sigla V_R Select, che permette di impostare le tensioni di riferimento con cui confrontare i valori analogici da leggere (V_{R+} e V_{R-}). Nonostante questa possibilità, dopo l'implementazione di una rete di condizionamento opportunamente calibrata, è possibile sfruttare direttamente i riferimenti interni di tensione V_{SS} e V_{DD}.

Si hanno quindi due multiplexer: MUX A e MUX B, che hanno il compito di eseguire la selezione del canale da inviare al modulo S/H. I due multiplexer possono lavorare in maniera alternata (si attiva prima l'uno e poi l'altro) oppure può lavorare unicamente il MUX A, che ha la possibilità di effettuare la scansione automatica di canali selezionati.

Ogni multiplexer dispone di un'altra particolare caratteristica che lo rende più affidabile; gli ingressi, uno positivo (V_{INH}) e uno negativo (V_{INL}). L'ingresso positivo è quello che accetta il segnale che verrà inviato all'ingresso non invertente dell'amplificatore del modulo S/H; l'ingresso negativo, invece, normalmente viene connesso al riferimento negativo di tensione (V_{SS}) per avere una lettura assoluta.
I due ingressi, in pratica, vengono inviati contemporaneamente nel modulo S/H, che sottrae il segnale negativo dal positivo. Viene quindi eseguito il campionamento e conversione a 10bit (*Cap. 2.2.1).*
Per poter usufruire di tutte queste funzioni il modulo ADC del PIC24F si avvale di un numero elevato di registri, i quali rendono molto complicata la sua configurazione. La Microchip ha reso disponibile una libreria di funzioni per facilitare la fase di configurazione, ma, dato che nella nostra applicazione non si fa un uso avanzato del convertitore A/D si è preferito un'impostazione di tipo manuale agendo sui singoli bit dei registri di configurazione.

L’inizializzazione del convertitore A/D avviene attraverso i registri:

- **AD1CON1**
  Registro di configurazione, consente di impostare: l’accensione e il funzionamento IDLE del modulo, nonché il formato del risultato, disponibile come intero o frazione con segno o senza, l’ impulso (trigger) per il termine del campionamento e l’avvio della conversione, la modalità e l’istante d’avvio del campionamento e il termine della conversione; quest’ultimo bit differisce dai precedenti perchè è di sola lettura, viene solitamente sfruttato per rilevare la fine conversione qualora stiamo facendo funzionare il modulo in modalità manuale;

- **AD1CON2**
  Registro di configurazione, imposta utilizzo delle tensioni di riferimento positive e negative, la selezione del multiplexer per definire l’ingresso analogico al convertitore, inoltre permette la gestione degli interrupt generati internamente al modulo ADC;

- **AD1CON3**
  Ultimo registro di configurazione, permette la selezione della sorgente di clock per il convertitore, di impostare il tempo di conversione dell’ADC ed i periodi di campionamento di S&H

- **ANCFG**
  In questo registro viene selezionato quali porte devono funzionare come analogiche e quali no;

- **AD1SSL**
  Nei pic24F gestisce la scansione automatica dei pin d’ingresso da AN_0 a AN_{15}, questa funzionalità è riservata al solo MUX A, che abiliterà la conversione sui soli canali attivati;

- **AD1SSH**,
  il registro con funzionamento analogo al precedente, gestisce la scansione dei pin d’ingresso da AN_{16} ad AN_{24} relativi al MUX B;

Capitolo 3 Periferiche di visualizzazione

3.1 Display LCD alfanumerici

Questi display sono tipicamente costruiti con case diversi, ma posseggono tutti un aspetto che li accomuna: il controller utilizzato, basato sul diffusissimo HD44780 di Hitachi.

Grazie a questa caratteristica comune, il software implementato per interfacciare un particolare LCD è quasi completamente compatibile, il controller, dialoga con il microcontrollore mediante comandi definiti secondo parametri standard indicati sulle tabelle dei datasheet [6]; ma la configurazione dei pin dell'LCD potrebbe variare; quando si cambia modello di display.

Il display alfanumerico che ho selezionato per questo progetto (fig. 3.1) agevola l'interfacciamento dell'utente con il sistema mediante un menù di selezione.

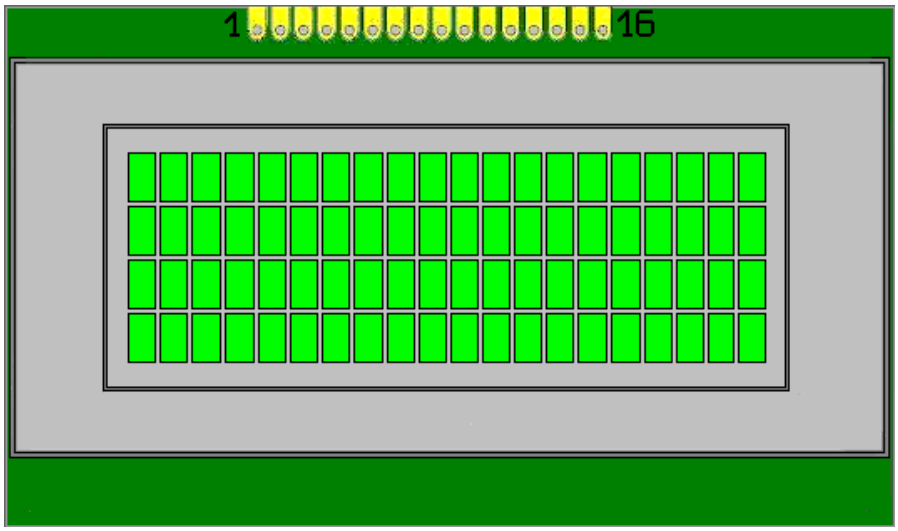

fig. 3.1: LCD 20x4 schematizzato

Di Seguito verranno descritte nel dettaglio le modalità di collegamento e comunicazione di connessione con il microcontrollore.

Modalità di collegamento

Molti display proprio come quello in esame sfruttano una alimentazione a 5V; la corrente che viene assorbita dal display, dipende in larga misura dall'estensione (in termini di caratteri) e dalla potenza necessaria per l'accensione dei singoli pixel.

Viene generalmente fornita la possibilità di modificare la regolazione del contrasto dei caratteri con lo sfondo, nel nostro caso si è ottenuto mediante l'utilizzo di un trimmer collegato come in *fig. 3.2* , che permette una regolazione ottimale del contrasto desiderato. In alternativa è possibile fissare il livello di contrasto, sfruttando un partitore resistivo di valore noto, si evita così l'uso di un trimmer.

Il controller HD44780 consente l'interfacciamento con un microcontrollore mediante due modalità di connessione:

- Con bus dati a 8 bit;
- Con bus dati a 4 bit.

Le due alternative si distinguono sia per le diverse impostazioni software sia per il numero di pin impiegati. Entrambe le modalità hanno però in comune il numero di pin di controllo.
In particolare:

- E:
  L' Enable è pin sul quale viene inviato il segnale di sincronismo: quando viene posto a livello logico alto, il controller esegue la lettura del dato presente sulla linea dati, lo decodifica e successivamente esegue sul display il carattere o comando corrispondente; in generale quando dovremo inviare un carattere da visualizzare, sarà necessario prima predisporre la linea dati, quindi porteremo il pin E a livello logico alto, per permettere che il controller esegua la lettura del dato, dopodichè riporteremo E a livello logico basso;

- R/W:
  Read or Write, il display può funzionare in modalità sia di lettura che di scrittura, anche se normalmente viene sempre utilizzata la sola funzione di scrittura. In generale se il pin viene configurato a livello logico basso, si desidera scrivere un dato, viceversa se viene posto a livello alto si vuole leggere un dato;

- RS:
  Register Select, questo pin viene usato per comunicare al display il tipo di informazione che stiamo per inviare sul databus; se RS viene posto a livello logico alto, si desidera inviare un carattere da visualizzare sul display, se invece lo poniamo a livello logico basso, stiamo inviando è un comando da eseguire; esistono vari tipi di comando, i più comuni sono: selezione della riga su cui scrivere, riposizionamento del cursore, e il clearscreen.

La modalità con bus dati a 4 bit consente un risparmio dei pin del microcontrollore, senza far aumentare significativamente la complessità del controllo. Per questa ragione, di seguito si andrà ad utilizzare un display connesso ad un microcontrollore utilizzando solo 4 bit del databus; lo schema è quello riportato in *fig. 3.2*

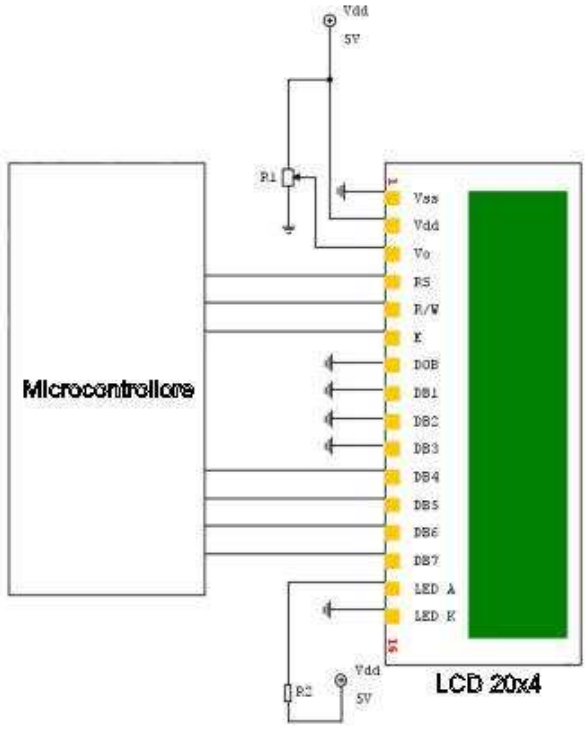

fig. 3.2: Schema collegamento LCD 4 bit

Per garantire un corretta comunicazione con il display, è importante agire sia sul databus sia sui segnali di controllo: E, RS,R/W. Il controller HD44780 infatti necessita di sequenze di comandi e dati prestabilite trasmessi categoricamente rispettando tempistiche minime .
Indicate nel flow chart del setup illustrato in *fig. 3.3*.

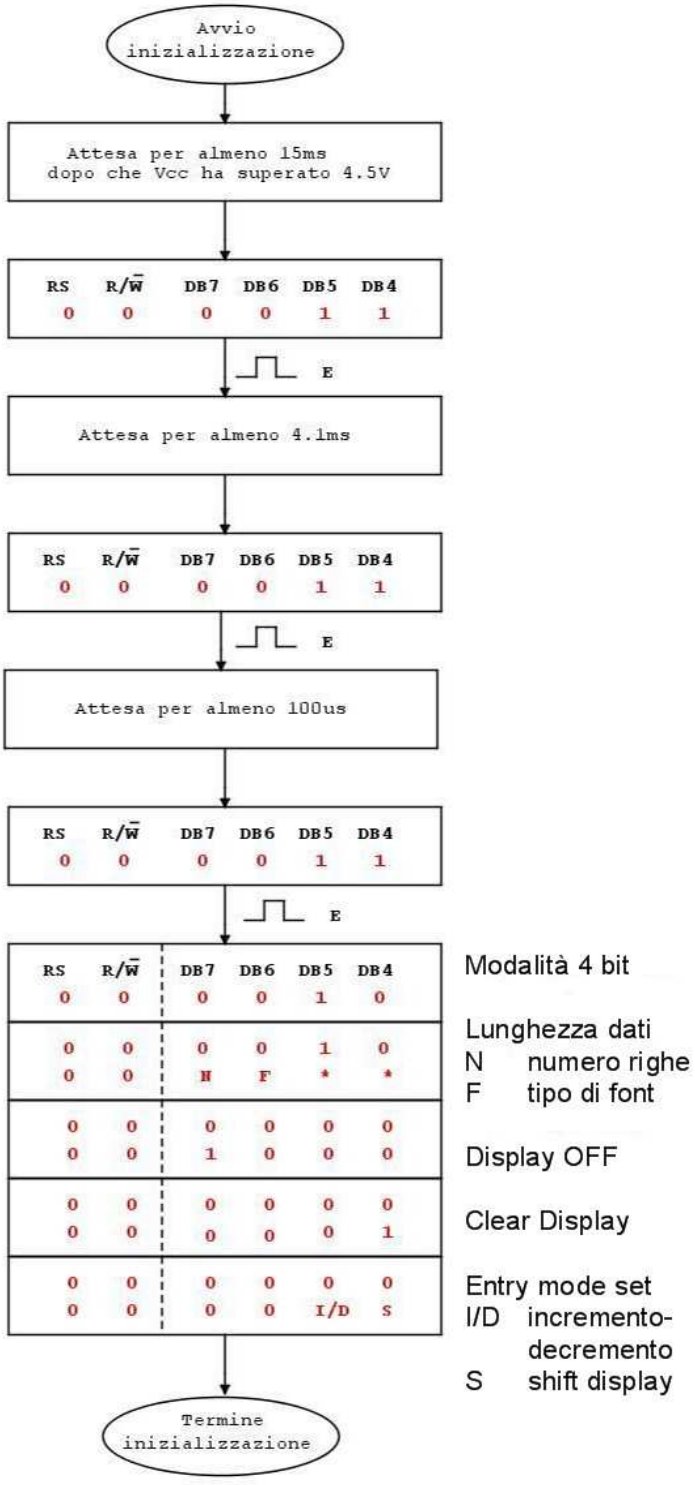

fig. 3.3: flow chart LCD setup

Per permettere l'invio di un comando al display, è essenziale configurarei pin RS ed R/W entrambi a valore 0 e iniziare la trasmissione prima con il nibble basso, poi con il nibble più alto. Ogni invio deve essere seguito da una transizione completa (low – high - low) del Enable.

Dopo l'inizializzazione il display accetta comandi di setting e trasmissione dati.

```
// Send command to LCD
void LCD_SendCommand(char Command)
{
   LCD_RS = 0;
   LCD_SendByte(Command);
}

// Send data to LCD
void LCD_SendData(char Data)
{
    LCD_RS = 1;
    LCD_SendByte(Data);
}
```

Il display LCD alfanumerico utilizzato è il modello MC42005A6W-BNMLW della marca MIDAS da 4 righe e 20 colonne. Ogni carattere viene configurato mediante una matrice di punti 5x8.

Nel display viene quindi impostata la modalità di trasmissione a 4 bit, mediante la procedura di inizializzazione descritta in *fig. 3.3*. Nonostante la comunicazione con il display abbia una natura puramente unidirezionale in scrittura, si evita di porre il pin R/W direttamente a massa ma si preferisce connetterlo al microcontrollore che genererà i segnale di controllo. Così facendo si potranno permettere sviluppi futuri per quanto riguarda l'eventuale lettura del display.

Interfaccia Utente

Il passo successivo al processo di inizializzazione è la creazione via software di un menù che permetta all'utente una efficace selezione delle possibili azioni da intraprendere. Questa configurazione si occupa della sola parte di selezione e le verrà quindi affiancato un ulteriore display di tipo grafico *(Cap. 3.2)* per permettere una visualizzazione qualitativa delle forme d'onda in elaborazione, altrimenti impossibile con i normali display alfanumerici.

Risulta molto importante definire con attenzione le tempistiche di comunicazione con il microcontrollore, queste infatti non devono interferire con il processo di misurazione quando questo è in atto.
Diviene di assoluta importanza attuare quindi modifiche alle informazioni visualizzate solo se necessario soprattutto nella fase di elaborazione (*fig.3.4*), si è quindi optato per la modifica selettiva del contenuto di una singole riga del display lasciando inalterate le altre, che permettono all'utente la sole indicazioni sullo stato del processo. Questa caratteristica viene resa evidente dal dettaglio della riga 4 dalla *fig. 3.5* dove come conferma dell'avvenuto salvataggio della caratteristica visualizzata compare un messaggio di feedback "OK SALVATO".

Questa soluzione viene affiancata da una scelta progettuale dettata dall'impossibilità di aggiungere via hardware circuiti antirimbalzo per i pulsanti di selezione implementati direttamente sulla development board [5]. Si è quindi decisa la creazione di una funzione che inibisca per un periodo di almeno 255 millisecondi pari a 255 istanti successivi di campionamento le funzioni di interfaccia utente del processo di elaborazione senza però interrompere il processo di misurazione; questo è possibile mediante l'aggiunta via software di un bit di flag che viene settato ogni qual volta si attiva la funzione di salva o interrompi l'esecuzione. L'istruzione di set del flag risulta praticamente invisibile nel flusso dell'elaborazione, questo la rende ideale.

Tale funzionalità non compromette la possibilità di salvataggi ripetuti della caratteristica reputata di interesse da parte dell'utente, infatti le forme d'onda con cui avviene la sollecitazione del sistema hanno durata superiore al tempo minimo definito. Il sistema dovrà resetare il flag la termine del processo di misura relativo alla caratteristica in esame, prima di permettere un nuovo salvataggio della caratteristica in esame.

Mentre nella fase preselezione le funzioni non devono rispettare una tempistica rigida quindi si è ricorso ad una rutine di delay che permette di raggiungere lo stesso scopo sopra descritto ma impedisce l'esecuzione di altre funzioni durante tutto il periodo temporale di 255 ms.

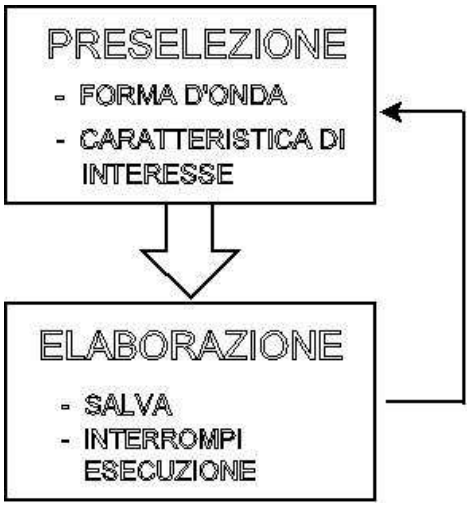

fig. 3.4: Schema a blocchi della struttura di menu.

Il menù si divide in 2 sezioni *(fig. 3.4)* distinte in base allo stato di avanzamento del processo di elaborazione. Il primo settore permette la selezione del segnale da elaborare mediante l'utilizzo di due cursori DX e SX, emulati grazie al collegamento di due pulsanti. In caso di avvenuta pressione vengono ricaricate, mediante una lettura in memoria, le sole informazione riguardanti nome e lunghezza del segnale precedente o successivo a quello attualmente visualizzato, questa scelta evita

un inutile download nel PIC della totalità dei campioni, che avviene solo mediante la pressione di un pulsante di conferma (Play). La forma d'onda così selezionata viene quindi caricata nella RAM del PIC.

Prima di proseguire alla fase di generazione/acquisizione viene richiesto di selezionare quale tra le variabili in esame (V_{in}, $Acc._{RMS}$ o V_{out}) si desidera visualizzare sul display grafico; questa decisione può essere comunque modificata successivamente nel corso della elaborazione dei campioni.

Terminata ufficialmente la fase di preselezione, si dà inizio al processo di elaborazione vero e proprio. Di seguito viene fornita la sezione di codice che definisce un singolo step del processo di visualizzazione e aggiornamento.

```
if (clear_screen == 0) {
            LCD_GOTO (2,1);
            LCD_PUTS ("                    ");
            LCD_GOTO (2,1);
            LCD_PUTS ("BUFFER 1");
            clear_screen = 1;
            buffer = BUFFER_1;
            //visualizzazione a video informazioni sul buffer
            lunghezza = read_word (0x0000, buffer);
            if (lunghezza == 0){
                  LCD_GOTO (3,1);
                  LCD_PUTS ("                    ");
                  LCD_GOTO (3,1);
                  LCD_PUTS ("Buffer vuoto");
            } else {
                  for (i = 0; i < 8 ; i++){
                  nome[i] = ((BYTE) read_byte ((i + 2), buffer));
                        }
                  LCD_GOTO (3,1);
                  LCD_PUTS ("                    ");
                  LCD_GOTO (3,1);
                  for (i = 0; i < 8 ; i++){
                        LCD_PUTCH (nome[i]);
                  }
                  LCD_PUTS(" - ");
                  LCD_PUTUN(lunghezza);
                  }
      }
      __delay_ms(255);
      if (DX == 0) select = 1;
      if (SX == 0) select = 9;
```

Durante la seconda fase, il menu oltre a indicare che è attualmente in corso una elaborazione permette una interazione tramite i due selettori precedenti, che ora assumono la funzione di "salva" e "interrompi l'esecuzione".

Nel caso si volesse interrompere l'esecuzione, si viene indirizzati ad un ulteriore schermata del menù, che chiede la conferma della decisione e in caso di risposta affermativa si provvede all'arresto di tutti i processi, riportando il sistema alla fase di preselezione. Nel caso fosse premuto il tasto salva (Cap. 4.2) la forma d'onda attualmente visualizzata sarà salvata nel primo buffer vuoto disponibile nella memoria Flash esterna, senza alterare il flusso dei processi in elaborazione.

fig. 3.5: Fase di elaborazione conferma di salvataggio

Si fornisce di seguito il codice caratteristico della seconda fase:

```
void stai_visualizzando (int onda_select){

        int onda = onda_select;
        LCD_CLEAR();
        LCD_PUTS (" Stai Visualizzando ");
        LCD_GOTO (2,1);
        switch (onda){
                  case 0:
                        LCD_PUTS ("      V_out");
                        break;
                  case 1:
                        LCD_PUTS ("      Acc RMS  ");
                        break;
                  case 2:
                        LCD_PUTS ("      V_in");
                        break;
        }//end switch
       LCD_GOTO (3,1);
       LCD_PUTS ("  <<    ZOOM    >>  "); // Cap. 2.3
       LCD_GOTO (4,1);
       LCD_PUTS ("SAVE            BACK");

     }
```

Se si reputasse necessario risulta comunque possibile implementare anche la retroilluminazione del display, mediante opportuni collegamenti.

3.2 Display LCD Grafico

Una semplice interfaccia grafica può essere realizzata con dei LED o avere dei display alfanumerici che permettono di scrivere dei messaggi. Quando questo non dovesse essere sufficiente e si dovesse avere l'esigenza di migliorare la comunicazione macchina utente si ricorre spesso a display grafici.

Sempre più frequentemente i sistemi embedded si interfacciano con l'utilizzatore per mezzo di un'interfaccia grafica, questa periferica permette l'implementazioni di una tipologia di informazioni grafiche nettamente superiore alla visualizzazione di semplici caratteri, come nei display lcd alfanumerici.

Il display grafico selezionato in questo progetto sfrutta il controllore grafico LH155BA, che sebbene non più "giovane" viene spesso utilizzato grazie alla sua reperibilità e i costi ridotti. Il controllore possiede tutta l'elettronica per potersi interfacciare con un microcontrollore e poter decodificare i comandi necessari per poter controllare un display GLCD monocromatico.

Il display GLCD che ci apprestiamo ad utilizzare nel nostro sistema ha una risoluzione 128x64 pixel, da ciò consegue che disporremo di 128 punti sull'asse X mentre sull'asse delle Y di una risoluzione di 64 punti. Quindi il nostro display grafico 128x64 possiede 8192 punti; questo risultato si raggiunge moltiplicando i coefficienti massimi associati agli assi X ed Y.

Valutando che il controllore LH155BA possiede al suo interno 1024byte di memoria RAM, che corrispondono a 8192 bit, tale caratteristica permette l'indirizzamento della totalità dei pixel di un display GLCD con risoluzione 128x64.

Volendo sfruttare il display come sola fonte di visualizzazione grafica qualitativa delle onde in elaborazione è stata scelta la configurazione in *fig. 3.6*, che permette di sfruttare il massimo spazio disponibile.

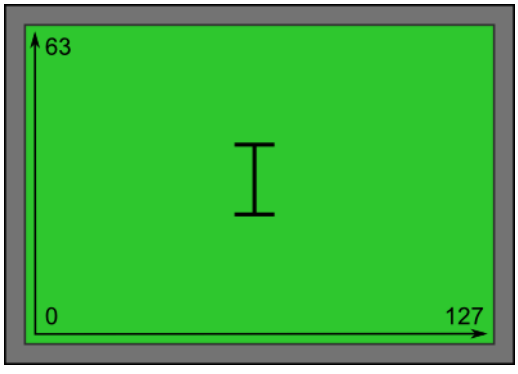

fig. 3.6: Configurazione Glcd

Sebbene l'immagine in *fig. 3.6* illustri che la risoluzione di ogni settore è di 128x64 pixel, per accedere ad ogni punto non basta semplicemente inviare dei comandi per un punto di coordinate x,y compreso tra 0 e 63. Il controllore divide infatti ogni settore in 16 pagine, come riportato in *fig. 3.7*. Ogni pagina possiede a sua volta una risoluzione 64x8 e può essere scritta solo 8 punti alla volta(verticalmente).

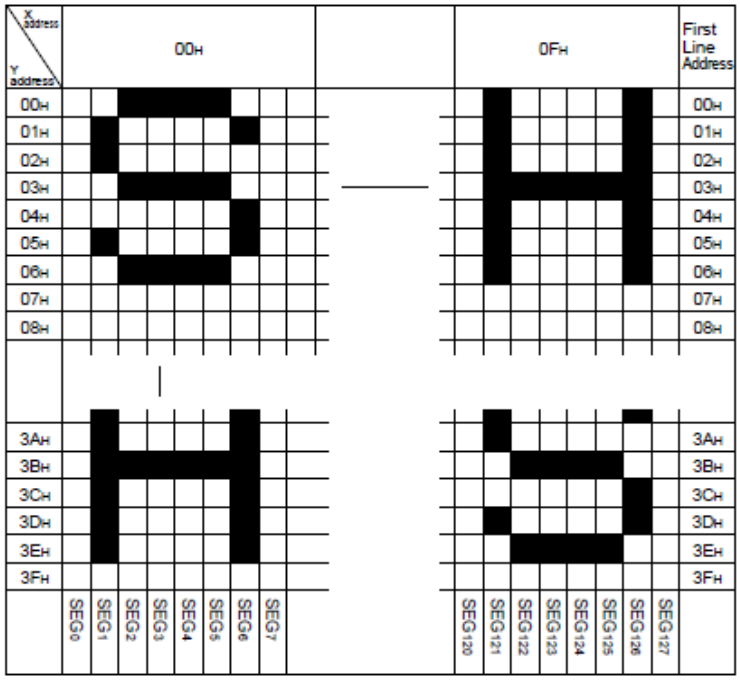

fig. 3.7: Divisione interna in settori del GLCD (fonte: datasheet LH155BA [7])

Questa divisione, che si ritrova anche in altri controllori, discende dal fatto che per ogni pixel non è associato un byte ma un solo bit, per cui un byte controlla 8 pixel. Per tale ragione 8 punti è il numero minimo di pixel che è possibile scrivere. Per poter disegnare un pixel è necessario dunque impostare, il controllore (settore), la pagina e l'indirizzo (nominato Y address). Dal momento che è possibile scrivere solo 8 punti alla volta, si intuisce che, qualora si volesse modificare un solo pixel, sarebbe necessario leggere prima il valore del byte attuale e solo successivamente scrivere il valore del nuovo pixel senza alterare il valore degli altri bit, mediante la creazione di opportune maschere software. Sfortunatamente il sistema sviluppato non può sfruttare tale soluzione in quanto non avrebbe permesso una misura in real time, dovuta alla latenza derivante dalla sequenza dei processi di lettura, elaborazione del nuovo dato e scrittura sul display.

Questa caratteristica non è idonea per permettere una caratterizzazione dei dispositivi ottimale, risulta infatti necessario rendere l'intervallo temporale di elaborazione e comunicazione dati minore

possibile, per evitare la sovrapposizione di processi successivi. Si è optato quindi per una alternativa, che sfrutta la possibilità di comunicare con il display in sola scrittura per via seriale. Questa soluzione ci garantisce la trasmissione del dato in appena 1.2 microsecondi come meglio definito nella *fig.3.8* e *tabella 3.2* garantendo un basso impatto sul processo di misuarzione ma ci impedisce però di conoscere a priori il valore del byte memorizzato nella RAM del display.

Si renderebbe quindi necessaria la creazione di una matrice 64 x 16 byte all'interno della memoria RAM del microcontrollore, questa sarebbe in grado quindi di memorizzare il valore assunto da ogni bit del display in ogni singola pagina da un byte. Questa configurazione risulta sovradimensionata in quanto, non è necessario conoscere la configurazione assunta dai pixel non relativi alla pagina da 8 bit che vogliamo modificare. Si è quindi creato un array di soli 64 byte per memorizzare i valori che vengono assegnati ad ogni singola riga dell'asse delle ordinate riferite ad una sequenza successiva di 8 bit nell'asse delle ascisse, questa soluzione permette una rapidità nell'elaborazione nettamente superiore alla precedente infatti i dati vengono salvati nella memoria RAM interna al PIC rendendoli fruibili in tempi molto brevi, viene inoltre garantito il riutilizzo della memoria allocata, la quale verrà resettata al termine dell'indirizzamento di un byte.
Il controllore si interfaccia con un microcontrollore con le seguenti linee di controllo:

<table>
<tr><th>Pin No.</th><th>Symbol</th><th>Function</th></tr>
<tr><td>1</td><td>V_{SS}</td><td>Signal ground (GND)</td></tr>
<tr><td>2</td><td>$\overline{RES}$</td><td>Controller reset (module reset)</td></tr>
<tr><td>3</td><td>CS</td><td>Chip enable</td></tr>
<tr><td>4</td><td>RS</td><td>Used to identify data sent by MPU at D0 to D7.</td></tr>
<tr><td>5</td><td>P/S</td><td>Used to switch between parallel and serial interface.
<table>
<tr><th>P/S</th><th>Chip select</th><th>Data identification</th><th>Data</th><th>Read/Write</th><th>Serial clock</th></tr>
<tr><td>"H"</td><td>CSB</td><td>RS</td><td>S0-D7</td><td>RDB,WRB</td><td>-</td></tr>
<tr><td>"L"</td><td>CSB</td><td>RS</td><td>SDA</td><td>Write only</td><td>SCL</td></tr>
</table>
P/S= "H": Fixes SDA and SCL at "H or "L".
P/S= "L" : Fixes D7 to D0 at HI-Z : RDB and WRB at "H" or "L".</td></tr>
<tr><td>6</td><td>$\overline{WR}$</td><td>Data write (write data to the module at "L")</td></tr>
<tr><td>7</td><td>RD</td><td>Data read (read data from the module at "L")</td></tr>
<tr><td>8~15</td><td>DB0~DB7</td><td>Data bus</td></tr>
<tr><td>16</td><td>V_{DD}</td><td>Power supply (+3.3V)</td></tr>
<tr><td>17</td><td>SCL</td><td>Used as data transfer clock pin when serial interface is selected.
The SDA data is shifted at rising edge of the SCL.
Internal serial/parallel conversion to 8-bit data is performed by the rising edge at 8th clock of the SCL.
Be sure to set this pin at "L" after completion of transfer or at not accessing.</td></tr>
<tr><td>18</td><td>SDA</td><td>Used as serial data input pin when serial interface is selected.</td></tr>
<tr><td>19</td><td>A</td><td>LED Backlight (+)</td></tr>
<tr><td>20</td><td>K</td><td>LED Backlight (-)</td></tr>
</table>

Tabella 3.1: Elenco segnali di controllo GLCD (fonte: datasheet LH155BA [7])

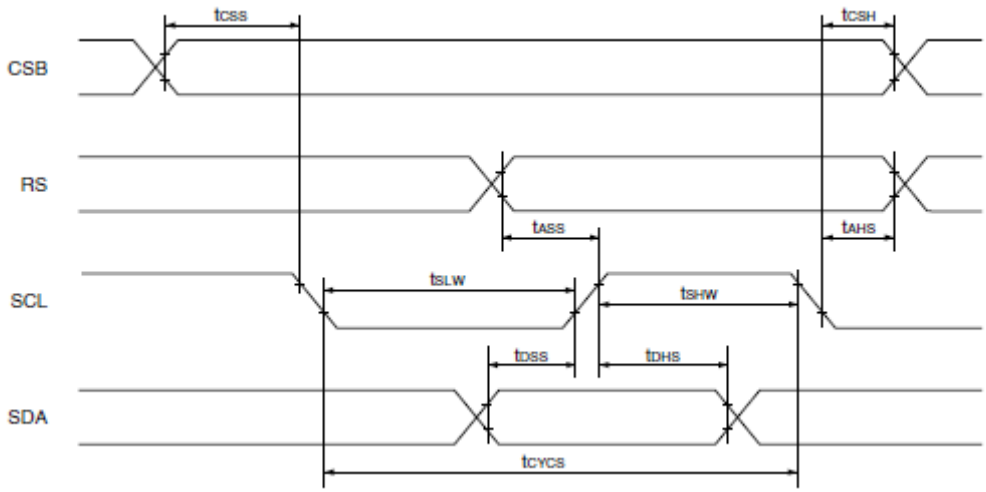

fig. 3.8: Sequenza di scrittura dati seriale controllore LH155BA (fonte: datasheet LH155BA [7])

(V_{DD} = 2.4 to 5.5 V, T_{OPR} = –30 to +85 °C)

PARAMETER	SYMBOL	CONDITIONS	APPLICABLE PINS	MIN.	MAX.	UNIT
Serial clock period	tCYCS		SCL	1 000		ns
SCL "H" pulse width	tSHW			400		ns
SCL "L" pulse width	tSLW			400		ns
Address setup time	tASS		RS	80		ns
Address hold time	tAHS			80		ns
Data set up time	tDSS		SDA	400		ns
Data hold time	tDHS			400		ns
CSB to SCL time	tCSS		CSB	80		ns
CSB hold time	tCSH			80		ns
Input signal rise and fall time	tR, tF		All of above pins		30	ns

Tabella 3.2: Tempi associati alle sequenze di scrittura seriale su un controllore LH155BA (fonte: datasheet LH155BA [7])

Qualora i tempi di scrittura rappresentati in *Tabella 3.2*, non fossero rispettati è possibile che i comandi non vengano riconosciuti per cui l'inizializzazione o la rappresentazione di oggetti sullo schermo avverrà in maniera disordinata (punti mancanti o punti in posizioni errate).

Prima di poter utilizzare una qualunque funzione grafica, risulta quindi necessario eseguire inizialmente un processo di inizializzazione del display; tale operazione, permette la definizione delle caratteristiche della comunicazione selezionate, inoltre ha il compito di porre a 0x00 il contenuto della memoria RAM del controller, un processo equivalente a un clear screen. È possibile verificare una errata inizializzazione osservando un effetto neve sullo schermo, ovvero i pixel dello schermo hanno un valore casuale che riflette il contenuto random della memoria RAM.

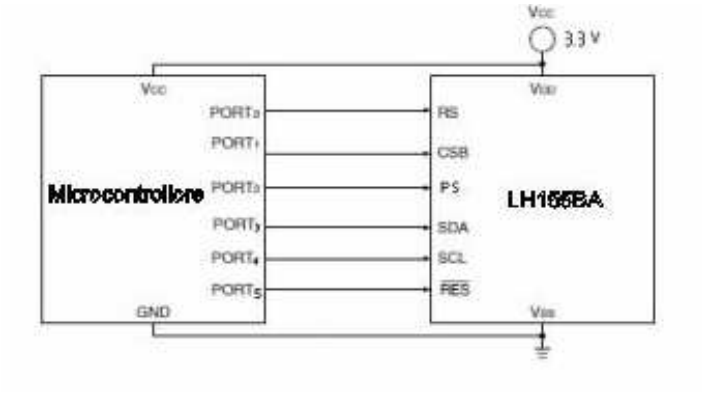

fig. 3.9: Schema elettrico collegamento seriale con microprocessore

La funzione Write della libreria grafica descritta in seguito è stata scritta per PIC24F ed è testata su PIC24FJ256DA210 Development Board.

```
void LCD_write(char dato){
   CS=0;     // Attivazione chip controllo
   SCL=0;
   delay_us(DELAY);
   RS=0;
   // Trasmissione effettuata dal MSB al LSB
   for ( i=0;  i<8;  i++ ){
         SCL=0;
         if ( (dato & 128) == 0 ) SDA=0;
         else SDA=1;
         dato = dato<<1;
         delay_us(DELAY);
         SCL=1;    // Il dato immesso è considerato valido al set di SCL
         delay_us(DELAY);
        }
   delay_us(DELAY);

SCL=0; // Predisposizione per il dato successivo.
   SDA=0;
   delay_us(DELAY);
   CS=1;        // Disattivazione chip controllo.
   delay_us(DELAY);
}
```

Funzioni aggiuntive.

Durante la fase di sviluppo del progetto si è deciso di introdurre due assi cartesiani di riferimento (X e Y) per permettere una facile definizione dell'onda e dei suoi estremi d variazione. Verrà associato all'asse delle ascisse la variabile indipendente tempo, mentre all'asse delle ordinate corrisponderà l'andamento della grandezza fisica da visualizzare, che può essere espressa in tensione (V) o come accelerazione (m/s^2) a seconda del tipo di grandezza che si è deciso di visualizzare.

L'introduzione di questa caratteristica diminuisce l'area disponibile per la visualizzazione dell'andamento caratteristico del segnale, portandolo da un 128x64 a un 120x54, ma introduce

l’indicazione dettagliata dell’intervallo minimo massimo di variazione mediante l’utilizzo di font 5x7.

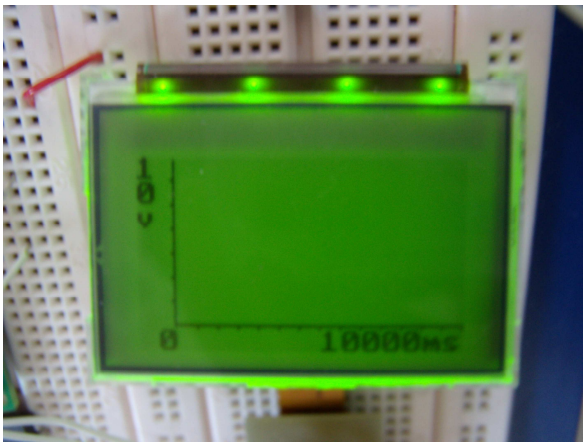

fig. 3.10: Display grafico configurazione finale

Per compensare la diminuzione di risoluzione viene aggiunta alle opzioni di visualizzazione la possibilità di variare la risoluzione temporale, mediante la pressione di due selettori viene quindi realizzato uno zoom su una finestra dell’onda fino a 1/ 256 dell’estensione temporale totale.

Ovviamente questa possibilità ha un limite inferiore dettato dal numero minimo di pixel visualizzabili sul display, quindi il coefficiente di riduzione di scala permetterà al massimo di visualizzare una porzione del segnale elaborato pari a 120 pixel, ognuno di questi equivale ad un istante preciso di campionamento del segnale.
Nel caso generale invece ogni singolo pixel visualizzato rappresenta la media aritmetica di Pix_{Value} campioni (Eq.3.2).

$$M_{Value} = \frac{\sum_{i=1}^{Pix_{Value}} V_{camp}}{Pix_{Value}} \qquad \textit{Eq. 3.1}$$

Dove V_{camp} corrisponde al livello di tensione corrispondente al i-esimo campione, mentre il numero di campioni corrispondente al singolo pixel visualizzato ed è definito da Pix_{Value} nell’*Eq. 3.2* .

$$Pix_{Value} = \frac{Lunghezza}{Zoom \cdot 120} \qquad \textit{Eq. 3.2}$$

L’ *Eq. 3.2* presenta un limite inferiore definito dall’impossibilità di visualizzare frazioni di singoli campioni, quindi se Pix_{Value} assume un valore inferiore all’unità questo sarà ricalcolato automaticamente, mediante un opportuno scaling del valore di Zoom, dato che può, a piacere dell’utente assumere tutti i valori delle potenze del 2 da 1 a 256. Sono invece mantenute costanti le altre variabili, quali lunghezza, che rappresenta la il numero di campioni totale del segnale in

elaborazione e il coefficiente numerico caratteristico del numero di pixel usati nella rappresentazione.

Ad ogni pixel della forma d'onda viene assegnata l'ampiezza in valore assoluto dell'oscillazione media corrispondente ai Pix_{Value} campioni caratteristici dell'intervallo temporale, tale valore è reso proporzionale al valore massimo V_{Max} assunto dal segnale in esame .

$$Y_{PixValue} = 54 - \frac{M_{Value} \cdot 54}{V_{Max}} \qquad \textit{Eq. 3.3}$$

Risulta necessario eseguire una procedura di sottrazione in quanto l'asse delle ordinate non fa corrispondere il suo livello zero alla comune posizione assunta dall'asse delle ascisse in un diagramma cartesiano, bensì nel punto di massimo assoluto (sessantaquattresimo bit), come si evidenzia in *fig.3.7*.

Combinando opportunamente il risultato dell'*Eq. 3.2* e *Eq. 3.3* è possibile definire univocamente le coordinate spaziali assunte da un singolo pixel nel display. Viene quindi associato l'incremento di un unità sull'asse dei tempi ogni Pix_{Value} campioni, mentre a $Y_{Pix\ Value}$ corrisponde il valore dell'ampiezza assunto nell'asse delle ordinate. Come si può constatare pur avendo implementato una visione variabile, la precisione della visualizzazione rimane di tipo puramente qualitativo; viene quindi implementata parallelamente la possibilità di salvataggio della totalità campioni in elaborazione in una memoria Flash esterna *(Cap.4)*, nonché il collegamento mediante interfaccia USB ad un PC, necessario per un download delle informazioni.

Capitolo 4 Periferiche di Memorizzazione

La categoria dei microcontrollori, di solito non dispone internamente di grandi quantità di memoria per il salvataggio permanente delle variabili. Apparentemente questo può sembrare un ostacolo, soprattutto in applicazioni, come nel caso in esame in cui è necessario caricare o salvare in runtime i segnali campione. I quali dovranno rendersi disponibili senza essere ricaricati su successive misurazioni, anche casi in cui venga tolta l'alimentazione.

L'utilizzo di una eeprom esterna è sicuramente la scelta migliore, anche se la Microchip ha approntato una libreria che permette di emulare nel microcontrollore il funzionamento di una eeprom, sfruttando la memoria programma. Sebbene entrambe le possibilità siano state valutate , si è deciso di sfruttare una memoria flash esterna, già implementata sulla demoboard del PIC , la quale comunica con il microcontrollore attraverso una interfaccia SPI *(Cap. 4.1).*

4.1 Interfaccia SPI

La comunicazione SPI (Serial Peripheral Interface) è una tra le più semplici ed efficaci. L'interfaccia SPI descrive una comunicazione Master Slave *(fig.4.1)*, di tipo sincrono e full-duplex. Il clock viene trasmesso con linea dedicata ed è possibile la ricetrasmissione dei dati.

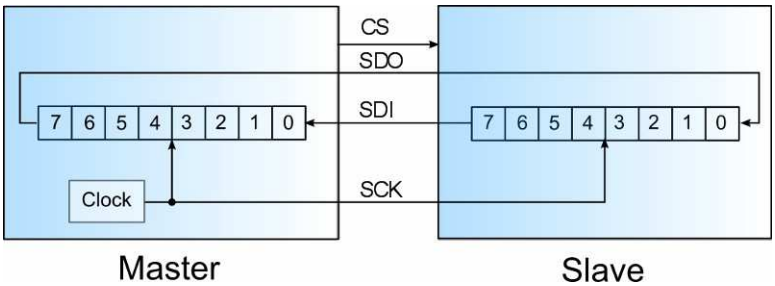

fig. 4.1: Connessione tra Master e Slave mettendo in evidenza i registri a scorrimento interni.

Il modulo di un microcontrollore che implementa l'interfaccia SPI può essere rappresentato come uno Shift Register; analizzeremo per maggiore chiarezza il metodo di trasmissione di un byte. Prima di avviare una comunicazione, il Master attiva la linea CS relativa allo Slave con cui vuole effettuare la comunicazione e successivamente il clock alla frequenza con cui avverrà la trasmissione. Dopo l'attivazione dello Slave, i bit interni al registro a scorrimento del Master vengono trasmessi all'esterno in sequenza ad ogni fronte di CK positivo, solitamente a partire dal bit più significativo. Il bit traslato entra nel registro dello Slave, il quale a sua volta inizia a svuotare

il proprio registro inviando il bit più significativo attraverso la linea SDI. La comunicazione termina quando l'ultimo bit del registro viene trasmesso.
Si nota che quanto descritto non è riferito ad un particolare tipo di formattazione. Il byte trasmesso è una semplice sequenza di bit, il cui significato dipenderà dall'applicazione.

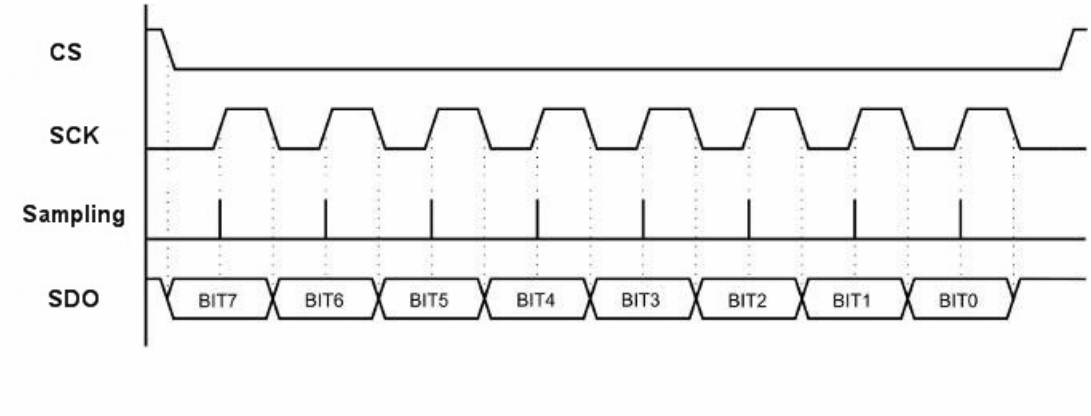

fig. 4.2: Sequenza di comunicazione SPI

Per permettere la comunicazione con la memoria flash, risulta necessario abilitare la connessione che sfrutta il protocollo SPI, come riportato in [12]. Mediante la funzione:

OpenSPI (conf1, conf2, conf3);

I valori che vengono assegnati alla funzione permettono la configurazione di tre registri necessari per il setting della porta SPI, nello specifico:

- conf1: configura il registro SPI1CON1;
- conf2: configura il registro SPI1CON2;
- conf3: configura il registro SPI1STAT.

4.2 – Comunicazione con la memoria

4.2.1 Flash memory

La flash memory, rientra nella tipologia di memorie non volatile a stato solido, per le sue caratteristiche può essere usata come memoria in lettura-scrittura. Esistono due famiglie principali di memorie flash, dette NOR e NAND flash [13], queste differiscono per l'architettura ed il processo di programmazione, entrambe permettono comunque una elevata velocità di comunicazione e una buona capacità di memorizzazione. Indipendentemente dalla tecnologia costruttiva le informazioni sono immagazzinate in un array di transistori MOS a floating gate *(fig. 4.3)*.

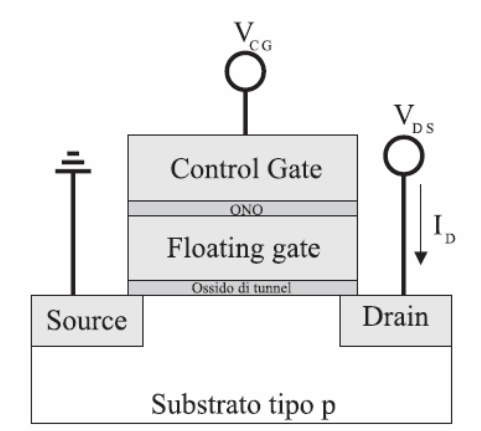

fig. 4.3: Sezione schematica di un MOSFET floating gate (fonte: [14])

4.2.2 Implementazione Hardware

Di seguito verrà definita la struttura hardware implementata nella memoria, che deve essere preventivamente organizzata per poter realizzare il programma implementato nel PIC.

Viene fissata univocamente la massima estensione temporale delle grandezze riproducibili pari a 10 secondi, inoltre da quanto affermato nel *Cap. 2.2.1* il segnale viene elaborato con una frequenza pari a 1 KHz; quindi viene univocamente stabilita la massima dimensione assunta dai campioni di una forma d'onda da riproducibile dal sistema, questa è pari a 10000 elementi e ad ogni campione (12bit) verrà associato uno spazio in memoria pari a due byte. Si è quindi deciso di creare una struttura per la gestione dei buffer in memoria flash, composta da 10 buffer di dimensione fissata *(fig. 4.5)*; divisi a loro volta internamente in 3 settori, che permettono di identificare il contenuto univocamente *(fig. 4.6)*. La prima sezione composta da 2 Byte contiene la dimensione dei dati

salvati, la seconda contiene il nome del buffer, rappresentabile al massimo con 8 caratteri (8 Byte), infine sono memorizzati tutti i campioni relativi alla forma d'onda.

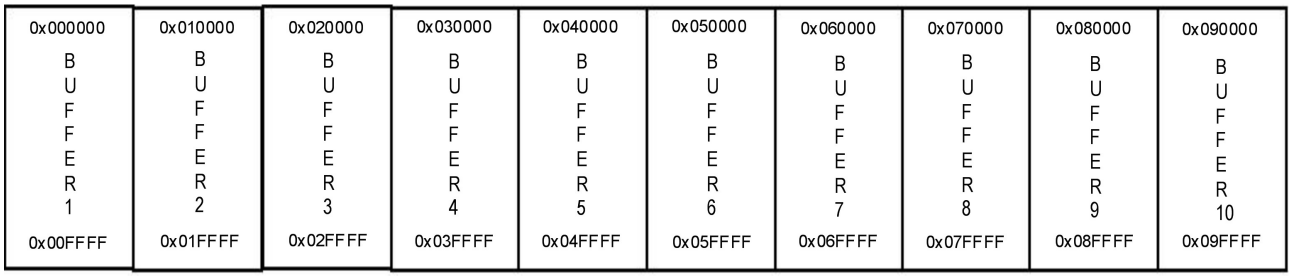

fig. 4.4: Indirizzamento dei buffer in memoria

Come si nota la dimensione dei buffer viene volutamente resa fissa nonostante il numero di campioni caratteristici per ogni dato sia variabile. Questa configurazione è ideale per evitare il ripristino totale dei dati in memoria ogni volta che l'estensione di un singolo buffer venga modificata. Inoltre per permettere alla memoria un ciclo di lettura più rapido, viene eseguita una cancellazione "virtuale" del buffer selezionato, che comporta la semplice scrittura di uno zero sul campo della lunghezza;. questo campo sarà testato ogni qual volta venga attivato un processo di scrittura sul buffer , per evitare la sovrascrittura di informazioni utili.

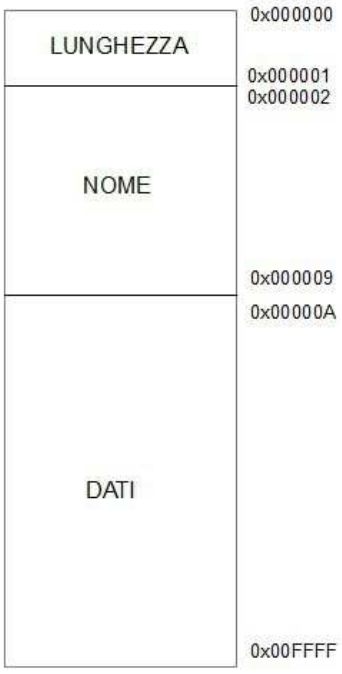

fig. 4.5: Suddivisione interna del buffer (fonte: [15])

2.4.3 – Libreria Load/Save

Il codice di seguito descritto rappresenta una rielaborazione delle funzioni sviluppate dal Dott. Fiumana Mattia durante il progetto di tesi per la gestione e comunicazione di dati mediante interfaccia usb, fra il microcontrollore e Pc [15] . Si provvederà in un successivo accorpamento di tali funzionalità al progetto in esame; per incrementare le potenzialità.
Le funzioni in esame sono mediante selezionabili dall'utente mediante un opportuno menu di selezione implementato con un display LCD alfanumerico *(Cap.3.1)*.

Load_Buffer

Lo scopo della funzione è l'esecuzione del caricamento della forma d'onda selezionata. Tale operazione avviene mediante il salvataggio del buffer selezionato in un array, creato appositamente e interno al microcontrollore; i dati così immagazzinati mantengono la formattazione originale (lunghezza, nome e valore dei campioni). La funzione di Load_Buffer viene eseguita prima dell'effettivo inizio della fase di caratterizzazione del dispositivo, non è quindi determinante a una prima analisi il tempo effettivo impiegato per eseguire la comunicazione.

```
void Load_mode(void){

      unsigned int lunghezza;
      BYTE buf_select =  choose();
      __delay_ms(250);
      lunghezza = read_word (0x0000, buf_select);
      int i;
      for (i=0; i<lunghezza; i++){
            Out_Buffer[i] = read_byte ((i+10), buf_select);
      }
      LCD_CLEAR();
      LCD_PUTS("  BUFFER CARICATO");
      __delay_ms(500);

}
```

Save_Buffer

In contrapposizione con la precedente la funzione, il Save_Buffer deve essere eseguito in run-time durante il processo di misurazione, che non può essere interrotto. Per permettere quindi un corretto salvataggio dei dati in esame, l'intero processo deve essere completato in un tempo inferiore al tempo di campionamento pari a 1 ms. Data l'elevata frequenza di lavoro del microcontrollore selezionato, la rapidità di risposta della memoria Flash, nonché del codice relativo all'apparato di conversione e adattamento del segnale, si può rilevare sperimentalmente che l'esecuzione del codice permette di rispettare tali limiti e non compromette quindi il processo di misura.

```
void Salva (void)
{
if ( CONF == 0){
    BYTE Full;
    Full =  first_free ();
    if (Full == 0xFF){
         initialize_buffer (0X05);
         write_buffer_ee (Out_Buffer, lunghezza , "ACC", 0x05);
    }
    write_buffer_ee (Out_Buffer, lunghezza , "ACC", Full);
    LCD_GOTO (4,6);
    LCD_PUTS ("OK SALVATO");
    }
}
```

Conclusioni

Realizzando il circuito descritto in questo elaborato e rappresentato dallo schema in *fig.1.4*, il processo di acquisizione e generazione descritto nel *Cap. 2* e l'interfaccia utente descritta nel *Cap. 3* si è riusciti a creare un sistema che permette una visualizzazione real-time dell'andamento del processo e mostra performance migliori rispetto al precedente progetto sviluppato mediante Labview.

Siamo quindi in grado di effettuare letture più rapide che permettono di compiere le operazioni di acquisizione senza sorpassare il millisecondo. I dati acquisiti vengono analizzati qualitativamente mediante la visualizzazione su display LCD grafico e se soddisfano le specifiche possono essere salvati in una memoria flash mediante un collegamento con interfaccia SPI, tale soluzione permette uno studio successivo più approfondito.. L'utilizzo dei buffer di memorizzazione è essenziale per il salvataggio di più misure, consentendo comunque il caricamento di una vasta sezione di sollecitazioni utili durante il processo di misurazione.

Tra i vantaggi di questo sistema possiamo evidenziare la possibilità di lavoro stand-alone, diviene quindi accessorio il collegamento con un computer.

Il computer sarà comunque utilizzato mediante un progetto integrativo allo stesso, per consentire all'utente il caricamento delle forme d'onda di interesse e lo scaricamento delle informazioni sensibili salvate.

In conclusione il progetto è riuscito a rispettare la totalità delle specifiche che erano state imposte.

Indice delle figure

Indice delle tabelle

Riferimenti bibliografici

[1] Immagine tratta dal sito www.micropelt.com

[2] Immagine tratta dal sito www.microchip.com

[3] Microchip, PIC24F Family Reference Manual, Sect. 08 Interrupts, www.microchip.com

[4] Microchip, PIC24F Family Reference Manual, Sect. 14 Timers, www.microchip.com

[5] Microchip, PIC24FJ256DA210 Development Board User's Guide,www.microchip.com

[6] Midas, Datasheet of MC42005A6W-BNMLW LCD 4x20, www.octopart.com

[7] Sharp, Datasheet of LH155BA, GLCD controller, www.sharp-world.com

[8] Linear Technology, Datasheet of LTC1450 , www.linear.com

[9] G. Iuculano, D. Mirri, "Misure Elettroniche", Cedam, 2004

[10] Microchip, PIC24F Family Reference Manual, Sect. 17 10-Bit A/D Converter, www.microchip.com

[11] J.Rabaey, A.Chandrakasan, B.Nikolic: "Digital Integrated Circuits: A design perspective"/"Circuti integrati digitali: l'ottica del progettista", 3rd Edition, Prentice Hall 2003

[12] Microchip, PIC24F Family Reference Manual, Sect. 23 Serial Peripheral Interface (SPI), www.microchip.com

[13] J. Rose, A. El-Gamal, A. Sangiovanni-Vincentelli, "Architecture of Field-Programmable Gate Arrays", Proc. IEEE, vol. 81, n. 7, Luglio 1993

[14] S. Lombardo, C. Spinella, C. Gerardi, M. Melanotte, E. Rimini: "MEMORIE MOS A «FLOATING GATE»: VERSO IL SINGOLO ELETTRONE CNR"

[15] M. Fiumana, A. Romani, M. Dini : Progetto di un sistema di gestione e comunicazione dati per un generatore di forme d'onda basato su microcontrollore (dicembre 2012)