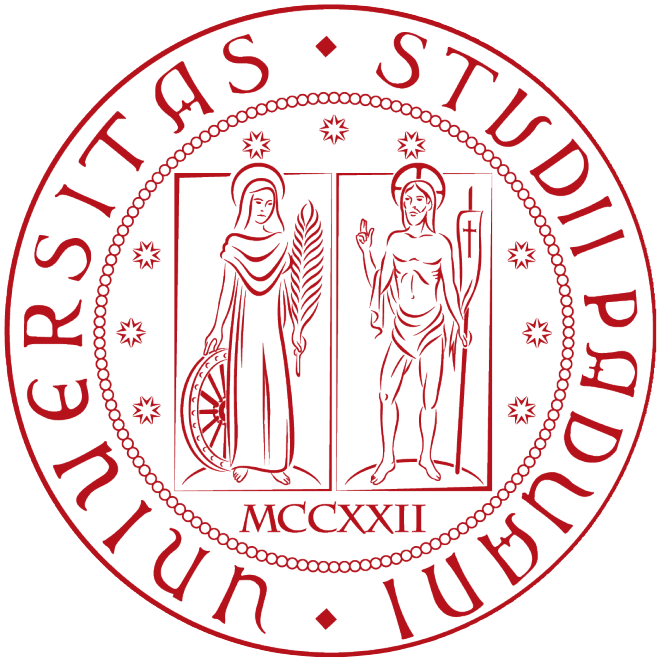
UNIVERSITÀ DEGLI STUDI DI PADOVA

Corso di laurea in Ingegneria dell'Informazione

Laboratorio di Ingegneria Informatica



Oscilloscopio basato su microcontrollore PIC18F4550 con output su LCD grafico 128x64

Autore: Lucchi Davide

15/06/2016

Introduzione

L'oscilloscopio è uno strumento in grado di graficare l'andamento del segnale collegato all'ingresso in funzione del tempo. Attraverso l'oscilloscopio si possono fare delle misure sull'ampiezza del segnale e il suo periodo. In commercio ne esistono di due tipi: analogici e digitali. Gli oscilloscopi analogici graficano il segnale su un tubo catodico ed il loro svantaggio principale è la necessità di dover eseguire manualmente le misure sul segnale, per conoscere i vari parametri elettrici. Gli oscilloscopi digitali invece, sono in grado di automatizzare questo processo, infatti grazie al fatto che il segnale viene campionato ed elaborato da un processore, questo può calcolare autonomamente i vari parametri.
Viste le capacità, l'oscilloscopio è stato fin dall'inizio uno strumento fondamentale in ciascun laboratorio elettronico. Attraverso opportuni trasduttori è infatti possibile convertire qualsiasi grandezza in un segnale elettrico, che può essere visualizzato e processato da un oscilloscopio. Ciascun modello è caratterizzato da vari parametri elettrici che ne determinano l'ambito in cui può essere utilizzato. Uno dei parametri più importanti è sicuramente la banda passante, la quale determina la massima frequenza visualizzabile dallo strumento.
Nel corso degli anni, grazie allo sviluppo tecnologico, sono stati sviluppati oscilloscopi sempre più comodi da trasportare e da utilizzare sul campo. Un esempio sono gli oscilloscopi costituiti da una semplice scheda elettrica che si interfaccia con un PC. La scheda implementa il campionamento mentre l'elaborazione del segnale viene affidata al PC.
Lo scopo di questo progetto è realizzare un oscilloscopio a bassissimo costo che, ove possibile, possa essere usato direttamente sul campo, senza alcun computer e senza dover ricorrere a strumenti estremamente costosi. Esso inoltre permette di fare misure qualitative di tensione e periodo (frequenza) sul segnale in esame. I componenti chiave del progetto sono un microcontrollore ad 8 bit ed un display grafico LCD.

Caratteristiche del dispositivo e obiettivi finali sugli errori:

Risoluzione	10 bit
Massimo range di input	0 - 5V
Banda passante	0 - 10KHz
Errore sui campioni	5/1024 = 0.00488V
Errore sulle misure di tensione con cursori	< 10%
Errore sulle misure di frequenza con cursori	<10%

Dalla tabella si nota che non sono ammesse tensioni negative in ingresso e quindi che teoricamente l'oscilloscopio non potrebbe rappresentare segnali negativi. Tale limite è stato superato imponendo lo zero della scala alla tensione di 2.5V. Questo implica però, che tutti i segnali debbano essere centrati a 2.5V per essere rappresentati in modo corretto, ma in questo modo è anche possibile rappresentare segnali naturalmente anche negativi quali i segnali sinusoidali.

Architettura dell'oscilloscopio

L'oscilloscopio in esame utilizza un microcontrollore Microchip PIC18F4550 per campionare il segnale, elaborarlo e mostrarlo su un display grafico a 128x64 pixels. Esso può funzionare in due modalità: normal mode e freeze mode. La modalità normal mode è quella di default. In questa configurazione il microcontrollore campiona all'infinito il segnale d'ingresso e continua ad aggiornarlo sul display. Il segnale campionato può essere scalato verticalmente e orizzontalmente agendo sugli opportuni potenziometri. In particolare il potenziometro che controlla la scala orizzontale va di fatto ad agire anche sulla frequenza di campionamento. La modalità freeze è attivata appena viene premuto un apposito pulsante, il quale, se premuto una seconda volta, permette di ritornale alla normal mode. In freeze mode il segnale d'ingresso non viene più campionato e sullo schermo viene stampato l'ultimo set di samples. Vengono inoltre stampati due cursori che possono essere verticali o orizzontali a seconda dello stato di uno switch, che permettono di eseguire misure qualitative di ampiezze (tensioni) e periodi (frequenze). La posizione di tali cursori può essere modificata attraverso altri due potenziometri.

Microcontrollore:

Parameter Name	Value
Program Memory Type	Flash
Program Memory (KB)	32
CPU Speed (MIPS)	12
RAM Bytes	2,048
Data EEPROM (bytes)	256
Digital Communication Peripherals	1-UART, 1-A/E/USART, 1-SPI, 1-I2C1-MSSP(SPI/I2C)
Capture/Compare/PWM Peripherals	1 CCP, 1 ECCP
Timers	1 x 8-bit, 3 x 16-bit
ADC	13 ch, 10-bit
Comparators	2
USB (ch, speed, compliance)	1, FS Device, USB 2.0
Temperature Range (C)	-40 to 85
Operating Voltage Range (V)	2 to 5.5
Pin Count	40

Figura 1

Per questo progetto e' stato usato un High-Performance, Enhanced Flash, USB Microcontroller with nanoWatt Technology Microchip PIC18F4550. Alcune delle caratteristiche principali sono indicate in Figura 1.
Il microcontrollore e' basato su una architettura ad 8 bit, ovvero il bus su cui transitano i dati ha larghezza 8 bit. Questo implica che i dati con lunghezza maggiore richiedono più di una istruzione macchina per essere eseguiti. La massima frequenza di oscillazione è $Fosc = 48\,MHz$ tuttavia essa non corrisponde alla frequenza a cui vengono eseguite le istruzioni. Internamente è infatti presente un divisore di frequenza che fa si che l'effettiva frequenza a cui vengono eseguite le istruzioni è $Fosc/4$ Ciò è dovuto alla necessità di mantere il tutto sincronizzato.
Il microcontrollore inoltre dispone di varie perifiche di comunicazione come UART, SPI, I2C e USB, di un convertitore da analogico a digitale a 10bit utilizzabile su 13 canali, comparatori, timer, moduli PWM etc. Sono infine presenti vari pin bidireazionali.

LCD Grafico 128x64

Un display LCD Grafico è una matrice di MxN pixel i quali possono essere accesi o spenti in modo indipendente l'uno dall'altro. Nonostante in commercio siano disponibili vari modelli, i loro schemi a blocchi sono sempre molto simili. Ciò che effettivamente caratterizza questi display sono il numero di pixel MxN, il controller interno che comanda i pixel e l'interfaccia esterna che puo' essere parallela o seriale. Per questo progetto è stato scelto un display 128x64 con controller interno KS0108 o equivalente ed interfaccia parallela. Il diagramma a blocchi è riportato in figura 3.

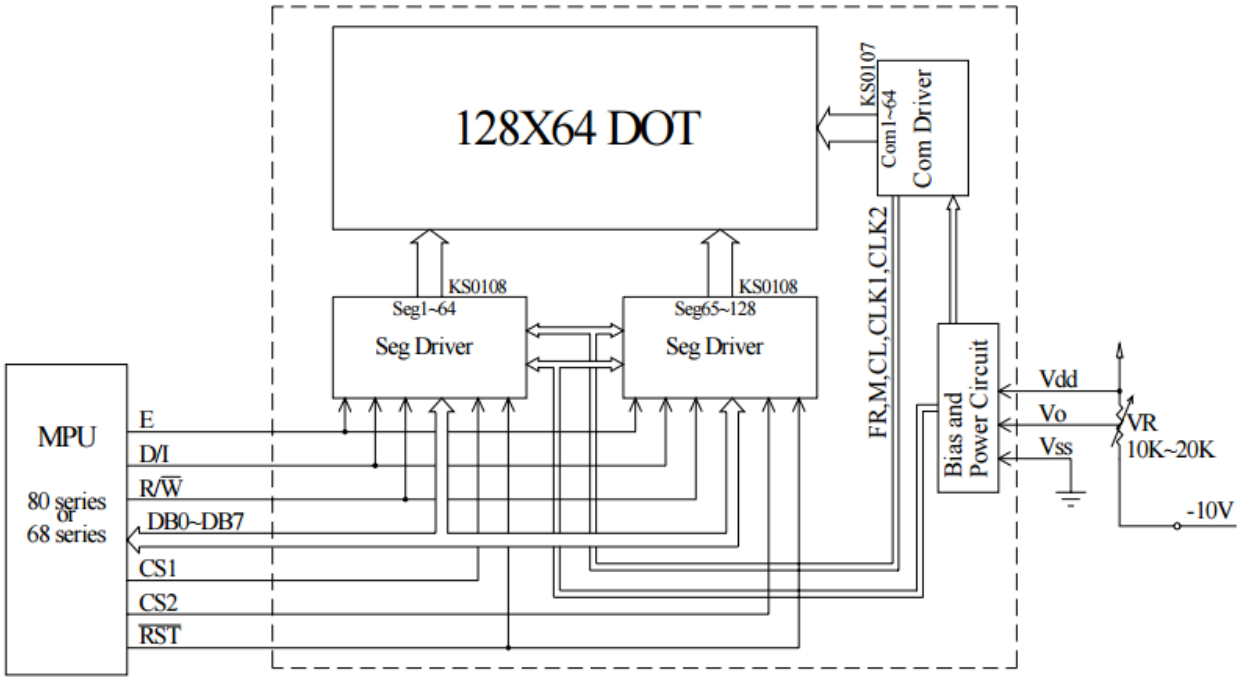


Figura 3

Dal diagramma si nota che la matrice di pixel è pilotata da due segment driver KS0108, uno per ciascuna metà. I controller possono essere selezionati con i segnali CS1/CS2 attivi alti. Il bus dati è unico ed è ad 8 bit, segnali DB0-DB7. Sono inoltre presenti i segnali E-D/I-R/W i quali controllano la direzione della comunicazione, ovvero se i dati devono essere scritti sull'LCD o se l'LCD li deve fornire. Vi è anche un segnale di reset, RST attivo basso che permette di resettare contemporaneamene entrambi i segment driver.
Ogni matrice è organizzata in pagine e colonne, struttura visibile in figura 4 per il caso 128x64. Si può notare che per ogni segment driver si hanno 8 pagine ciscuna composta da 64 colonne. Tali pagine e colonne sono selezionabili attraverso opportuni comandi e il valore presente sul bus viene scritto nella colonna selezionata. Questo fa si che vengano scritti sempre 8 pixel contemporaneamente ovvero un intera colonna di pagina. In figura 5 sono riportati i comandi del segment driver KS0108 i quali saranno implementati dalla libreria LCD.

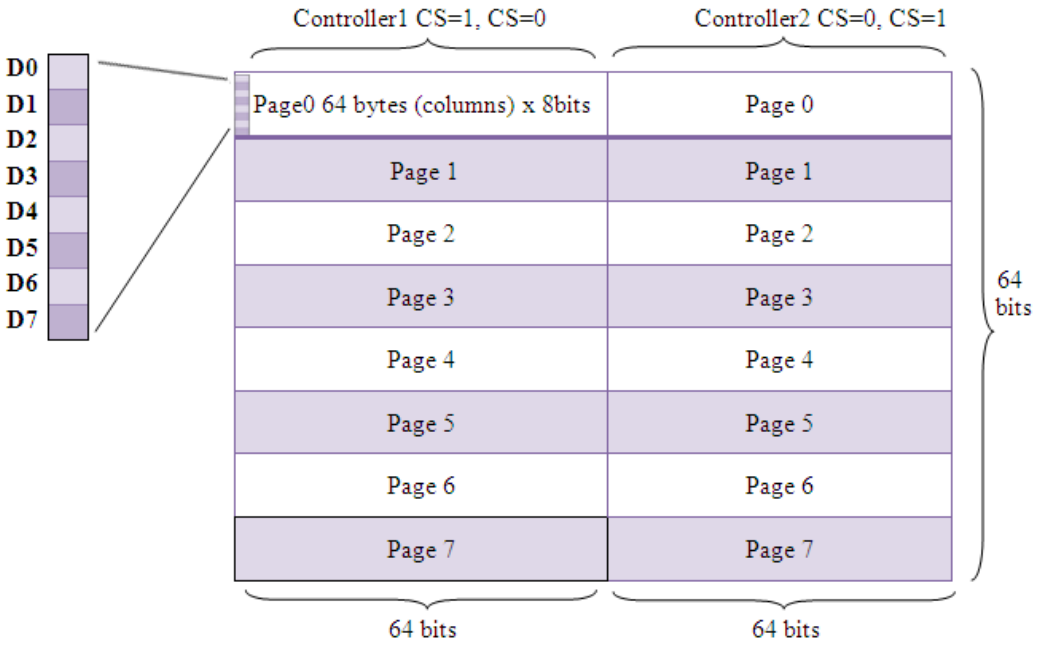


Figure 4

Instruction	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	Function
Display on/off	L	L	L	L	H	H	H	H	H	L/H	Controls the display on or off. Internal status and display RAM data is not affected. L:OFF, H:ON
Set address (Y address)	L	L	L	H	Y address (0-63)						Sets the Y address in the Y address counter.
Set page (X address)	L	L	H	L	H	H	H	Page (0-7)			Sets the X address at the X address register.
Display Start line (Z address)	L	L	H	H	Display start line (0-63)						Indicates the display data RAM displayed at the top of the screen.
Status read	L	H	Busy	L	On/ Off	Reset	L	L	L	L	Read status. BUSY L: Ready H: In operation ON/OFF L: Display ON H: Display OFF RESET L: Normal H: Reset
Write display data	H	L	Write data								Writes data (DB0: 7) into display data RAM. After writing instruction, Y address is increased by 1 automatically.
Read display data	H	H	Read data								Reads data (DB0: 7) from display data RAM to the data bus.

Figura 5

Architettura software

Il software alla base del funzionamento dell'oscilloscopio è formato da due elementi principali: la libreria per l'LCD e il programma main che viene eseguito dal microcontrollore. Questa modularizzazione permette di riutilizzare la libreria LCD in qualsiasi altro progetto purché vengano modificate le macro della libreria relative ai pin. Il programma main invece non è portabile su altri microcontrollori, a causa delle configurazioni specifiche dei registri, che possono differire da architettura ad architettura. È tuttavia portabile su altri PIC della famiglia PIC18F, purché essi contengano un convertitore ADC, con minime modifiche.

Libreria per LCD

Lo scopo di questa libreria è fornire un insieme di funzioni per poter scrivere sul display. Essa in particolare implementa tutti i comandi del controller KS0108 più alcune funzioni di utilità generale.
Perchè i comandi vengano accettati dal controller è necessario rispettare i diagrammi temporali di lettura/scrittura riportati in figura 6.

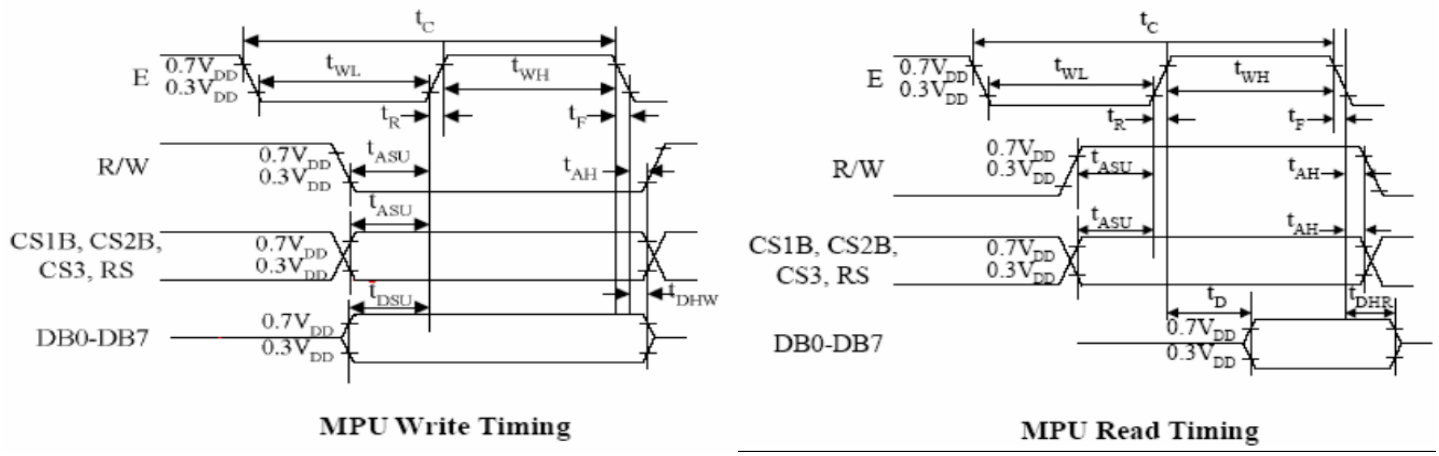


Figura 6

Dal diagramma si nota che i segnali vengono recepiti sul fronte di salita del segnale Enable (E).
La procedura di scrittura impone quindi di impostare la direzione della comunicazione R/W, D/I, selezionare il controller CS1/CS2, scrivere i dati sul bus ed infine mandare un segnale di enable. Nel caso della lettura, come prima deve essere definito il verso della comunicazione e attivato il controller corrispondente e una volta inviato il comando di Enable l'LCD risponde scrivendo il dato richiesto sul bus.

Funzioni della libreria:

I dettagli relativi all'implementazione di ciascuna funzione sono presenti nella documentazione di progetto. Queste funzioni possono essere divise sostanzialmente in tre gruppi: funzioni di supporto alla libreria stessa (il loro nome inizia con _) che non dovrebbero essere usate all'esterno, funzioni che implementano i comandi del **KS0108** e infine funzioni di utilità generale.
Perchè i dati possano essere scritti in modo corretto sul display è necessario per prima cosa chiamare **LCDinit()** che inizializza e resetta il display ed infine pone tutti i pixel a zero ovvero li spegne. Per scrivere una stringa sul display deve essere per prima cosa impostata la posizione del primo carattere ovvero il lato (sinistro, destro o entrambi), la pagina ed infine la colonna. A questo punto è possibile scrivere la stringa. Questa procedura è realizzata chiamando le funzioni **LCDpage(...)**, **LCDy(...)** e **LCDchar(...)** oppure **LCDstring(...)**. Tra le funzioni di utilità generale vi è **LCDclear()** che permette di cancellare l'intero contenuto del display, **emptycolumn()** per porre una colonna vuota tra due caratteri adiacenti, **xyaxis()** per disegnare la griglia dell'oscilloscopio, **lcdplotpixel2(...)** che permette di accendere/spegnere un singolo pixel o un intera colonna di pixel nella posizione specificata e **write_image(..)** che permette di disegnare un immagine atutto schermo.

```
void _lcd_enable(void);

unsigned char _lcd_status(void);

void _lcd_waitbusy(void);

void LCDon(unsigned char on);

char LCDy(unsigned char side, unsigned char value);

char LCDpage(unsigned char side, unsigned char page);

void LCDstartline (unsigned char side, unsigned char value);

void LCDcmd (unsigned char data);

unsigned char LCDread();

void LCDinit(void);

void LCDreset(void);

char LCDchar(unsigned char c, unsigned char position);

void LCDclear();

void emptycolumn();

void xyaxis ();

char lcdplotpixel2(char rx, char ry,int blank);

char LCDstring (char a, const char *buffer);

void write_image(const char* image, char top_to_bottom);
```

Implementazione dell'oscilloscopio

Il diagramma a blocchi dell'oscilloscopio è riportato in figura 7. In tutto vengono usati 5 canali ADC, di cui quattro collegati a potenziometri ed uno direttamente al segnale d'ingresso. Il potenziometro amplitude controlla l'ampiezza dell'onda mostrata sullo schermo, ovvero agendo su di esso si va a scalare il segnale lungo l'asse y. Il potenziometro sample time, analogamente, permette di modificare la scala lungo l'asse x. Il valore campionato, inoltre, viene utilizzato per creare un ritardo variable tra un campione e l'altro, ovvero il sample time dipende dalla posizione di questo potenziometro. Cursor 1 e 2 sono altri due potenziometri usati per regolare la posizione dei cursori orizzontali o verticali. Mode selection è un pulsante ad uno stato collegato ad un generico pin di I/O impostato come ingresso digitale, mentre horizontal/vertical switch è un interruttore a due stati collegato ad un altro pin generico di I/O.

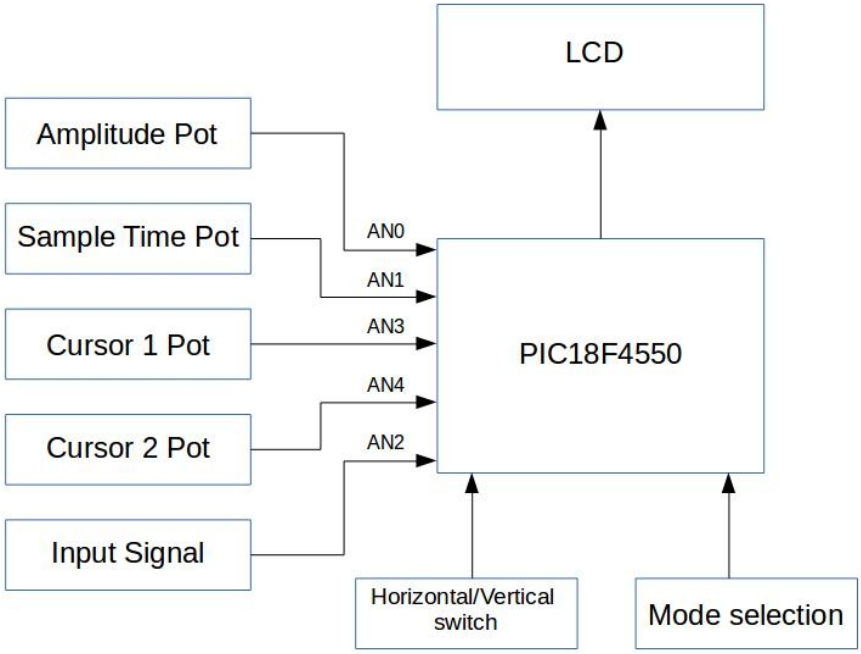


Figura 7

Modalità nomal mode:

Questa è la modalità di default in cui il dispositivo entra appena acceso. Per prima cosa il microcontrollore seleziona i canali AN0 e poi AN1 per campionare la scala verticale e la frequenza di campionamento. A questo punto si passa al campionamento vero e proprio del segnale d'ingresso, la parte di codice è riportata in figura 8. Il ciclo **for** campiona esattamente 128 volte il segnale d'ingresso e salva il dato nell'array **data**. Dopo aver salvato il dato, si entra in un ciclo **for** che realizza un ritardo dipendente dalla variablile **s_time_pot**, la quale, contiene il valore campionato dal potenziomentro Sample time. Questo ritardo, sommato al tempo richiesto dalle altre istruzioni, determina la distanza temporale tra un campione e il successivo ovvero la frequenza di campionamento. Il numero 128 è dovuto al fatto che la larghezza del display LCD è 128 pixels e che ciascun campione verra stampato nella corrispondente colonna verticale. Ciò significa che questo dispositivo non fa uso di tecniche di interpolazione, cosa comunque non realizzabile vista la

bassa capacità di calcolo del processore. Questa caratteristica risulta essere il limite più grande alla banda passante effettiva, infatti se si vuole che un periodo di un segnale occupi ad esempio 10 pixels sul display, bisognerà campionare ad una frequenza pari a 10 volte il periodo di quel segnale. La banda passante effettiva può essere calolata note le caratteristiche dell'ADC. La massima velocità di funzionamento dell'ADC è circa 100 000 samples al secondo e quindi la massima banda passante teorica circa 50kHz secondo il teorema del campionamento. In realtà dal ragionamento precedente si ha che un onda a 10KHz viene rappresentata con 10 punti, motivo per cui è stata assunta la banda passante 0-10KHz.

```
for (int i = 0; i < 128; i++)
{
        // start conversion
        ADCON0bits.GO=1;
        while(ADCON0bits.GO == 1);
        // save the sample
        data[i] =  ((unsigned int)(ADRESH << 8) | ADRESL);
        // Delay accordingly to Sample Time Pot. In this way each sample is
        // taken every s_time seconds
       for (int j = 0; j <  s_time_pot; j++)
      {
            ;
      }
 }
```

Figura 8

In figura 9 è riportato il codice che stampa i pixel sul display. Come prima cosa il punto campionato presente nell'array **data** viene convertito nell'effettiva tensione campionata con la formula

$$\frac{range}{2^{10}} * data[i]$$

ottenendo cosi' un valore compreso tra 0 e 5. Per poter rappresentare anche segnali negativi si è scelto di stampare sull'asse tutti i punti con ampiezza 2.5V, in questo modo i segnali che stanno tra 2.5-5V sono considerati positivi, mentre quelli tra 0-2.5V negativi. Questo impone delle condizioni sul segnale d'ingresso dell'ADC. Ovvero esso non solo deve avere un ampiezza compresa tra 0 e 5V ma deve anche essere centrato a 2.5V per garantire una corretta visualizzazione. Tali requisiti possono essere soddisfatti usando circuiti operazionali esterni, collegati in modo da scalare il segnale tra 0-5V e centrarlo a 2.5V. Questo è il motivo per cui al valore calcolato precedentemente viene sottratto il numero 2.5. A questo punto si moltiplica per amplitude, valore campionato dal potenziometro delle ampiezze, e per 2 piccolo fattore correttivo per adattare il segnale allo schermo. Il punto appena calcolato rappresenta la coordinata y a cui stampare il sample rispetto all'asse x. Tale punto può essere stampato con la funzione **lcdplotpixel2(...)** dove 32 rappresenta la coordinata y assoluta dell'asse x.

Modalità freeze mode:

Appena il pulsante mode selection viene premuto la modalità freeze viene attivata. Come prima

cosa vengono cancellati i vecchi cursori se presenti, dopo di chè vengono campionate le nuove posizioni dei cursori e stampate sullo schermo. A questo punto viene calcolata la distanza in pixel tra i due cursori e moltiplicata per la tensione per pixel o secondi per pixel a seconda dei cursori che si stanno utilizzando. Questo valore viene quindi stampato sullo schermo. Il programma cicla in modo infinito fino a quando il pulsante mode selection viene premuto una nuova volta determinando il passaggio all'altra modalità.

```
while (i < 128)
{
   int y = (int)(((((float)5/1024)*data[i])-2.50f)*amplitude*2);
   if (data[i] < min)
   {
      // take minimum of sampled data
      min = data[i];
      // take minimum of sampled data after processing
      min_y = abs(y);
   }
   if (data[i] > max)
   {
      // take maximum of sampled data
      max = data[i];
      // take maximum of sampled data after processing
      max_y = abs(y);
   }
   // plot pixel
   lcdplotpixel2(i,32-y,0);
   i++;
}
```

Figura 9

Dettagli sulla frequenza di campionamento

Per poter variare la scala orizzontale l'oscilloscopio fa uso di un potenziometro, il cui valore campionato, **s_time_pot**, viene usato come numero di iterazioni di un ciclo **for** tra un campionamento e il successivo. Per poter garantire una misura corretta dei periodi e quindi delle frequenze attraverso i cursori è necessario conoscere in modo preciso il tempo che intercorre tra un campione e l'altro. Questo tempo infatti rappresenta la distanza temporale tra due campioni successivi stampati sullo schermo ovvero secondi/pixel. Esso non è solo dovuto al ritardo introdotto dal ciclo **for** più interno, visibile in figura 8, ma anche a tutte le istruzioni che vengono eseguite dopo aver campionato e prima di avviare il successivo campionamento. Tali istruzioni sono dovute al salvataggio del dato e all'inizializzazione di una nuova iterazione di ciclo **for.** Questo ritardo è stato valutato in modo preciso analizzando il codice macchina ovvero contando quanti cicli di clock passano tra un campione ed il successivo. Si è giunti così alla formula

$$s\_time = (99+19*(s\_time\_pot-2))*0.1;$$

dove **s_time_pot** è il numero di iterazioni del ciclo **for** di ritardo. Il fattore 0.1 converte semplicemente il risultato in microsecondi.

Valutazione e collaudo

Il programma è stato compilato usando MPLAB IDE con il compilatore XC8. Questi strumenti sono forniti direttamente da Microchip e permettono anche il debugging completo del codice. È opportuno sottolineare che, vista la particolare architettura hardware del microcontrollore, un normale compilatore per C non può essere usato. Una volta compilato il codice, si ottiene un file .hex, il quale, può essere scritto nella memoria del microcontrollore o usato con opportuni software per fare una simulazione. Per la simulazione si è scelto di usare Proteus un simulatore di reti rilasciato da Labcenter Electronics. Esso permette di simulare il sistema finale in modo completo senza dover costruire alcuna scheda elettronica.
Il funzionamento generico del dispositivo è stato testato usando ingressi sinusoidali e onde quandre.

Sono stati ottenuti questi risultati:

Frequenza Input	Errore sulla frequenza	Errore sull'ampiezza
10Hz	~5%	~5%
100Hz	~5%	~5%
1KHz	~8%	~5%

Per frequenze superiori l'errore sulla frequenza aumenta sempre più a causa del fatto che sempre meno punti rappresentano un periodo completo del segnale. Ad esempio, con un input a 10KHz vi sono solo 8 punti per periodo. È opportuno sottolineare che l'errore sull'ampiezza dipende dalla scala dell'asse y. In particolare, per avere un errore il più piccolo possibile, è necessario impostare il potenziometro che controlla l'ampiezza sul più grande valore tale per cui l'intero segnale rimane nello schermo senza uscire. Infatti se supponiamo che il segnale abbia un ampiezza di 4V e sul display la sua altezza sia 20 pixel, allora la tensione per pixel V/p = 4/20 = 0.2 mentre se l'altezza fosse 40 pixel allora V/p = 0.1. Il valore V/p viene poi moltiplicato per la distanza in pixel tra i cursori per determinare l'ampiezza misurata. Nel caso della tensione picco picco Vpp le cose cambiano. Per calcolare tale tensione vengono usati direttamente i campioni quindi l'errore è dovuto solo alla precisione dell'ADC. Da datasheet l'ADC è a 10 bit quindi rappresenta i valori in ingresso con un numero intero compreso tra 0 e 1024. Noto che il range massimo d'ingresso è 5V allora l'errore dell'ADC è 5/1024 = 0.0048V

Conclusioni

I risultati ottenuti indicano che gli obbiettivi iniziali di progetto sono stati raggiunti, ovvero il dispositivo è in grado di rapresentare qualitativamente segnali variabili nel tempo con frequenze minori o uguali a 10KHz circa. È opportuno, comunque, sottolineare che le prestazioni risultano essere basse, a causa della limitata banda passante che condiziona anche l'errore sulle misure di frequenza. Questo dispositivo, tuttavia, rappresenta una base di partenza per quello che potrebbe essere un oscilloscopio molto più performante. Le prestazioni potrebbero essere aumentate usando un convertitore ADC esterno dedicato con più bit di risoluzione, interfacciato con un microcontrollore più potente, magari a 32bit. Il display grafico a sua volta potrebbe essere sostituito con un display tft o simile. In questo modo si potrebbero utilizzare tecniche di interpolazione per ricostruire in modo fedele il segnale, inoltre, attraverso un interfaccia USB, sarebbe possibile collegare lo strumento ad un computer ed elaborare i campioni con software come MATLAB. Queste modifiche permetterebbero di calcolare molti più parametri sul segnale d'ingresso ed il tutto in modo automatizzato ovvero usando direttamente i valori campionati. Un tale dispositivo si avvicinerebbe molto a quello che potrebbe essere un oscilloscopio commerciale.

Riferimenti esterni

Datasheet PIC18F4550

Datasheet KS0108

Datasheet Display 128x64

Proteus, Labcenter Electronics

MPLAB X

XC8 Compiler