UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI **INGEGNERIA**

Corso di Laurea triennale **in Ingegneria Biomedica**

Interfacciamento di sensori di flusso ai microcontrollori PIC e valutazione delle prestazioni per strumentazioni di laboratorio

Interfacing of flow sensors on microcontrollers PIC and performance evaluation for laboratory equipment

Relatore:

Prof. **Piva Francesco**

Tesi di Laurea di:

Tobia Mazza

A.A. **2021 /2022**

INDICE

1. INTRODUZIONE

1.1 Microcontrollori PIC ____ *3*

1.2 Registri e struttura interna ____ *4*

1.3 Compilatore e Programmatore ____ *5*

1.4 Connessioni ____ *6*

1.5 Display TFT ____ *9*

1.6 Sensori di flusso ____ *11*

1.7 Possibili applicazioni in ambito laboratoriale ____ *15*

2. OBIETTIVI ____ *20*

3 MATERIALI E METODI

3.1 PIC pic18f4550 ____ *21*

3.2 Programmatore pickit3 ____ *23*

3.3 Sensore di flusso YF-S201 ____ *26*

4. RISULTATI

4.1 Programma ____ *29*

4.2 Impostazioni Capture CCP ____ *32*

5. CONCLUSIONI ____ *37*

6. APPENDICE ____ *38*

7. BIBLIOGRAFIA E SITOGRAFIA ____ *47*

1. INTRODUZIONE

1.1 Microcontrollori PIC

I microcontrollori sono microprocessori specializzati nelle applicazioni di controllo elettronico e sono presenti in tutti gli elettrodomestici e le apparecchiature moderne. A differenza dei microprocessori di impiego generale, hanno al loro interno tutto quello che serve all'interfacciamento digitale ed analogico.

Un microcontrollore (MCU ovvero MicroController Unit) è un dispositivo elettronico integrato su un singolo circuito elettronico.

E' composto da:
- CPU (Central Processing Unit)
- Memoria di programma: ROM, EPROM, FLASH
- Memoria dati: RAM e EEPROM
- Oscillatore interno o esterno
- Porte di I/O configurabili
- Modulo Gestione Interrupt
- Moduli aggiuntivi (interfacce o moduli di comunicazione)

I PIC sono una famiglia di microcontrollori prodotti dalla Microchip Technology, sono progettati per interagire tramite un programma residente nella propria memoria interna e mediante l'uso di pin configurabili dal programmatore. I dispositivo programmatore ufficiale per i microcontrollori PIC è il Pickit.

La famiglia dei PIC della ditta Microchip (www.microchip.com) è la più completa e avanzata oggi disponibile; PIC è una famiglia di circuiti integrati a semiconduttore con funzioni di microcontrollore prodotto dalla Microchip Technology.
Il termine PIC non è usato come un acronimo: il suo nome aziendale è "PICmicro", anche se generalmente sta per "Programmable Interface Controller", il suo primo produttore (la "General Instrument") usava l'acronimo per "Programmable Intelligent Computer".

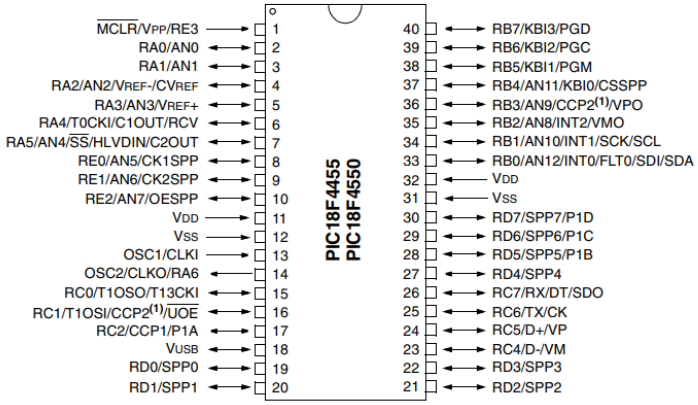
1.2 Registri e struttura interna

Tra i modelli più diffusi e recenti troviamo la famiglia dei PIC 18F che hanno una CPU a 8bit e dispongono di un moltiplicatore hardware per un calcolo più rapido degli algoritmi di controllo e interfacce di comunicazione: USART, CAN, USB, SPI e I2C.

Il modello utilizzato è il PIC18f4550 a 40pin disponibili.

Ad ogni pin è associata una porta I/O programmabile, a meno di alcune porte specifiche. Tolte, infatti, le porte destinate all'oscillatore esterno e quelle destinate all'alimentazione Vdd e Vss, sono tutte programmabili come ingresso o uscita grazie a degli appositi registri. Altri pin invece sono multiplexati con una funzione alternativa, per cui quando una periferica è abilitata quel pin non può essere usato come pin I/O generico.

40-Pin PDIP



Ogni porta ha tre registri per il suo funzionamento:

- TRISregister
- PORTregister
- LATregister

Il registro TRIS organizza la direzione dei dati, l'impostazione di un bit a 0 corrisponde a rendere il pin un'uscita, mentre l'impostazione ad 1 renderà il pin corrispondente un ingresso.

I registri PORT e LAT, in scrittura sono la stessa cosa, è infatti possibile scrivere un dato indifferentemente su uno dei due registri.

Invece, la lettura del registro PORT restituisce il dato letto sul pin del microcontrollore, è quindi influenzato da fattori esterni, la lettura del registro LAT restituisce il dato presente sul latch di scrittura.

1.3 Compilatore e programmatore

Per programmare un PIC è necessario l'utilizzo di un compilatore e di un programmatore (dispositivo fisico).

Il compilatore è un programma informatico che traduce una serie di informazioni scritte in un determinato linguaggio di programmazione, chiamato codice sorgente, in istruzioni di un altro linguaggio, chiamato codice oggetto, che generalmente sono codici eseguibili dalla macchina.

Il programmatore invece, o più correttamente programmatore USB, è un dispositivo che facendo da ponte tra il computer e il PIC fa passare il codice esadecimale in uscita dal compilatore direttamente alla memoria interna del microcontrollore, salvando tale codice nei registri di memoria permanente appositi.

1.4 Connessioni

I^2C

Il protocollo I^2C (pronuncia I-quadro-C, in Inglese I-squared-C) è stato creato dalla Philips Semiconductors nel 1982; la sigla, comunemente indicata anche con I2C, sta per Inter-Integrated Circuit. Il protocollo permette la comunicazione di dati tra due o più dispositivi I^2C utilizzando un bus2 a due fili, più uno per il riferimento comune di tensione. In tale protocollo le informazioni sono inviate serialmente usando una linea per i dati (SDA: Serial Data line) ed una per il Clock (SCL: Serial Clock line). Deve inoltre essere presente una terza linea: la massa, comune a tutti i dispositivi.

Una comunicazione avviene all'interno di due sequenze di segnali che il master invierà sul bus I^2C; queste sono il comando di START e di STOP. La regola fondamentale da tenere sempre ben presente è che nella trasmissione dei dati, il segnale SDA può cambiare stato soltanto mentre il segnale SCL è basso, ed il suo valore viene poi letto durante il fronte di salita o di discesa del segnale SCL. Se il segnale SDA cambia valore mentre SCL è alto, significa che il master non sta trasmettendo o ricevendo un dato, ma sta iniziando o terminando la comunicazione.

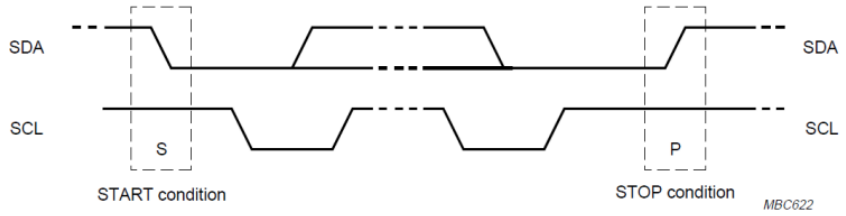


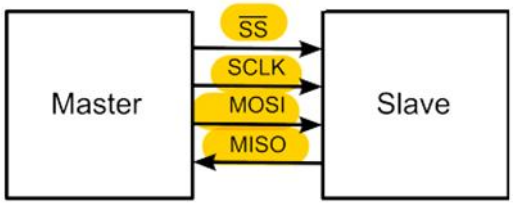
Figura 5. Sequenze di START e di STOP

Segnale di start: prima di inviare il segnale di start, lo stato del bus I²C si trova nella condizione in cui entrambi i segnali SDA ed SCL sono al livello logico alto. Per inviare il comando di start, il master pone a livello basso il segnale SDA mentre SCL è a livello logico alto, in questo modo segnala agli altri dispositivi che sta per iniziare una comunicazione.

Segnale di stop: prima di inviare il segnale di stop, sono stati trasferiti dei dati per cui lo stato del bus I²C si trova nella condizione in cui entrambi i segnali SDA ed SCL sono al livello logico basso. Tipica sequenza di trasferimento dati con il protocollo I^2C.
Comando di stop: il master pone prima a livello alto il segnale SCL e poi il segnale SDA. In questo modo segnala allo slave che la comunicazione è terminata.

SPI

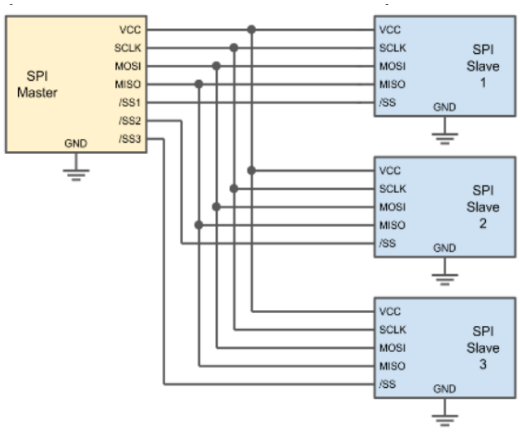
Il Serial Peripheral Interface o SPI (pronuncia: esse pi i o spi), in informatica, indica un sistema di comunicazione seriale tra un microcontrollore e altri circuiti integrati o tra più microcontrollori.
L'interfaccia SPI descrive una comunicazione singolo Master singolo Slave, ed è possibile sia trasmettere che ricevere dati in contemporanea.



La comunicazione avviene generalmente tra un Master ed uno Slave. L'interfaccia presenta inoltre 4 linee di collegamento (esclusa la massa comunque necessaria), per cui lo standard SPI è anche noto come 4 Wire Interface. È compito del Master avviare la comunicazione e fornire il clock allo Slave. La nomenclatura delle varie linee presenti nell'interfaccia SPI è normalmente la seguente:

- MISO : Master Input Slave Output
- MOSI : Master Output Slave Input
- SCLK : Serial Clock (generato dal Master)
- SS : Slave Select (Selezione dello Slave)

Alcuni modelli di PIC hanno più di una interfaccia seriale, il 18F4550 ne ha solo una e per questo non c'è necessità di specificare al compilatore a quale SPI si fa riferimento.
Nell'immagine seguente possiamo vedere uno schema riassuntivo della comunicazione SPI tra un dispositivo master e tre dispositivi slave.



I registri che ne permettono il controllo sono: SSPCON1, SSPSTAT, SSPBUF, SSPSR.

Registers associated with SPI communication

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other Resets
INTCON	GIE/ GIEH	PEIE/ GIEL	TMR0IE	INT0IE	RBIE	TMR0IF	INT0IF	RBIF	0000 000x	0000 000u
PIR1	PSPIF(1)	ADIF	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
PIE1	PSPIE(1)	ADIE	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
TRISC	PORTC Data Direction Register								1111 1111	1111 1111
SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
TRISA	—	PORTA Data Direction Register							--11 1111	--11 1111
SSPSTAT	SMP	CKE	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000

Legend: x = unknown, u = unchanged, - = unimplemented, read as '0'.
Shaded cells are not used by the MSSP in SPI mode.

Note 1: The PSPIF, PSPIE and PSPIP bits are reserved on 28-pin devices; always maintain these bits clear.

Si mostrano alcuni registri di configurazione per la comunicazione seriale.

Il primo e il secondo sono rispettivamente il registro di controllo e il registro di stato riferiti a tale interfaccia, permettono la configurazione del modulo per il trasferimento dei dati in modalità SPI.

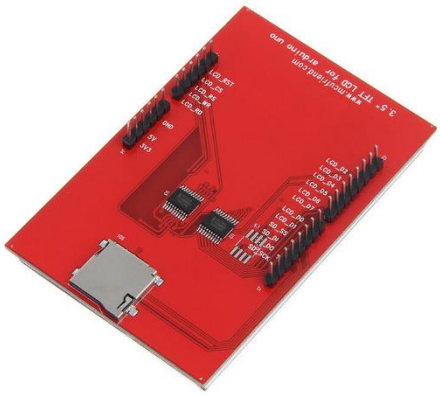
SSPSR, registro a scorrimento, è il registro utilizzato per spostare i dati.

SSPBUF è il registro del buffer, contiene il byte ricevuto o da trasmettere. Quando un byte di dati è ricevuto da uno dei due dispositivi, è copiato nel registro SSPBUF. Se dei dati devono essere inviati, allora questo registro è caricato dal programma. Dopo essere stato caricato con il byte da inviare il registro SSPBUF trasferisce il suo contenuto sul registro SSPSR. Quest'ultimo non è quindi direttamente accessibile ma può essere letto e scritto solo attraverso il registro SSPBUF.

1.5 Display TFT

Display TFT ILI9488

Questo tipo di display sono una variante dei display a cristalli liquidi (LCD) che utilizzano, appunto, la tecnologia TFT (thin film transistor, cioè transistor a pellicola sottile) per migliorare la qualità dell'immagine. Sono quindi display ad alta risoluzione, di solito a colori e alcuni anche touch screen. Fortunatamente sono a basso costo e ormai disponibili in moltissimi siti online.



I display TFT possono essere pilotati tramite interfaccia SPI, precedentemente descritta, o tramite interfaccia parallela a 8bit o a 16bit. Quest'ultima consente una maggiore velocità di trasmissione dati rispetto alla prima, per fare questo però sono necessari più collegamenti e ciò significa avere più pin occupati.
Il vantaggio, infatti, di utilizzare un'interfaccia di tipo seriale è proprio il fatto che per comunicare sono sufficienti meno pin.
Nel corso del tirocinio abbiamo lavorato con un display TFT 3,5" con risoluzione grafica 320x480 con controllori di tipo ILI9488 ad interfaccia parallela.

I pin da collegare sono: Vcc, Gnd, CS, RST, RS, WR, per quanto riguarda la porta comandi. Poi ci sono altri 8 pin da collegare per la porta dati che sono LCD_D0, LCD_D1, LCD_D2, LCD_D3, LCD_D4, LCD_D5, LCD_D6, LCD_D7.

I primi due pin sono per l'alimentazione, quello di CS (chip select) è un valore emesso dal master per indicare con quale dispositivo slave si vuole comunicare, normalmente il valore è 0, mentre il valore alto 1 indica la comunicazione.

Il terzo è il pin del display, RST, è usato all'inizio del programma per resettare il display Display TFT ILI9488 (connessione parallela).

Il display legge il comando sul fronte di discesa al contrario della connessione seriale che lo legge sul fronte di salita.

I pin del display da collegare al microcontrollore sono:

- RST (reset);
- WR (write operation);
- RS(register select);
- RD(read operation);
- CS(chip select);
- Vcc 3.3V;
- Vcc 5V;
- Gnd;

Questi display dispongono di due pin per l'alimentazione una a 5V e una a 3.3V in quest'ultimo abbiamo notato che molte volte non sono collegati a nessuna pista del circuito stampato.

1.6 Sensori di flusso

Il misuratore di portata noto anche come flussimetro o flussometro è uno strumento di misura della portata, volumica o massica, di un fluido, liquido o aeriforme.

L'interesse per la misurazione della portata è nato a seguito della necessità di conoscere le portate dei fiumi, in modo da studiare, se non controllare, le caratteristiche delle piene. In seguito, sono divenute necessarie misurazioni sulla portata di acquedotti, canali e altre condotte d'acqua.
Nell'industria moderna, le misurazioni di portata sono essenziali nel controllo dei processi e sono inerenti a diversi settori industriali, tra i quali: industria chimica, impianti idroelettrici, industria mineraria, industria alimentare e ingegneria ambientale.

Si può fare una classificazione per i misuratori di portata secondo il tipo di misura:

- Misuratore di velocità;
- Misuratore di portata vera e propria.

Ciascuno di questi può essere catalogato secondo la tecnica di misurazione:

- Misuratore dinamico, con parti in movimento provocato dal fluido misurato;

Flussimetro ad area variabile, o rotametro;
Flussimetro ad area variabile in derivazione.

- Misuratore statico, che misura l'effetto dello spostamento del corpo senza che le parti dello strumento siano in movimento.

Orifizio tarato o diaframma;
Tubo Venturi. Misuratori magnetici;
Misuratore a effetto Coriolis;
Misuratori a effetto Doppler;
Anemometro a filo caldo.

Per il calcolo della portata a mezzo di misuratori di velocità si sfrutta la relazione: Q=V*S dove Q è la portata, V la velocità del corpo, e S la sezione effettiva di passaggio, in unità coerenti. Nota la sezione di passaggio, a una velocità misurata corrisponde quindi una e una sola portata.

Misuratori diretti di portata

Misuratore volumetrico a turbina.

Sono molto usati, ne è esempio il contatore per il consumo idrico domestico, o il contatore posto nelle pompe di benzina. Possono raggiungere altissime precisioni, con scarti anche inferiori allo 0,1% del valore misurato. Si noti che, non è un misuratore di portata, ma di quantità. I modelli più recenti sono dotati di contatori a decadi digitali, meccanici o elettronici, per una lettura più agevole.

Ve ne sono essenzialmente tre tipi:

Misuratore a turbina: sono i più comuni, il flusso incanalato viene fatto passare attraverso una piccola turbina idraulica la cui velocità di rotazione è proporzionale alla portata. Un cinematismo provvede a far ruotare un indicatore su un quadrante, mostrando così la quantità di liquido passata. In alternativa la rotazione della turbina può venir misurata inserendo nella stessa un magnete che viene fatto passare in prossimità di apposito sensore generando un impulso per ogni giro.

Misuratori volumetrici: sono simili a una pompa volumetrica, in cui il fluido passante riempie delle cavità di volume noto; con sistemi vari, il numero di cavità riempite viene contato fornendo così la quantità di fluido passato.

Misuratori a disco nutante: sono sostanzialmente anch'essi dei misuratori volumetrici. In questi misuratori un disco ha un movimento di nutazione attorno a una sfera centrale, creando così una camera di volume noto; dato che la sfera centrale acquista un moto circolare, la misurazione è piuttosto semplice e alquanto precisa.

Problemi di misura

Da approfondire analiticamente, è necessario sapere che la misura di portata è una di quelle che presentano tra le più grandi difficoltà di caratterizzazione.

In un condotto la velocità del fluido non è omogenea: il fluido che lambisce le pareti è fermo, mentre quello che scorre al centro della tubazione ha la velocità massima il gradiente di velocità non è omogeneo e cambia distribuzione in funzione della velocità o di potenze della stessa.

La distribuzione di velocità nel condotto è condizionata dal percorso a monte e a valle del punto di misura del fluido. Curve, variazioni di diametro, ostacoli (valvole, filtri, specole) determinano turbolenze, tanto che esiste la raccomandazione (l'obbligo) di rispettare a monte e a valle del punto di misura percorsi rettilinei per consentire al flusso di stabilizzarsi.

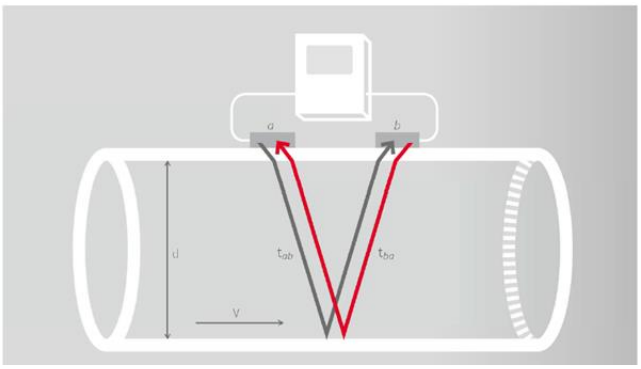
Dipendenza dalla temperatura, che influenza la viscosità del fluido che a sua volta determina transizioni nel regime di flusso, da laminare a semi turbolento a turbolento mutando drasticamente i fenomeni fisici che si manifestano nei fluidi.

Rangeability normalmente ridotta: le grandezze che vengono rilevate variano col quadrato se non col cubo della velocità, rendendole flebili sotto certe portate (limiti di sensibilità) ed enormi al di sopra (eccessiva influenza sul processo, ad esempio perdita di carico eccessiva) rendendo necessaria l'installazione di più misuratori da selezionare a seconda della portata da misurare.

Difficoltà (e talvolta impossibilità) di taratura dei sensori: ad esempio, ma non solo, la variazione del circuito necessaria all'installazione di un misuratore di riscontro altera le condizioni di funzionamento di quello da tarare.

Misuratori di Portata a Ultrasuoni

La misura di Portata deve essere rapida, semplice e diretta. Gli strumenti misurano con velocità del fluido da 0,01 a 25 m /s con una ripetibilità dello 0,15%.



Il principio di misura richiede una coppia abbinata di trasduttori sulla superficie esterna del tubo che vengono quindi utilizzati per generare impulsi ultrasonici.
Questi segnali viaggiano attraverso la parete del tubo e attraverso il fluido da un sensore all'altro, sia nella direzione del flusso che contro la direzione del flusso. Il principio chiave del metodo applicato è che le onde sonore che viaggiano nella direzione del flusso si muoveranno più velocemente di quelle che viaggiano contro di essa.
La differenza di tempo, detta "Tempo di transito" è direttamente proporzionale alla velocità di flusso del liquido e conseguentemente ai valori di flusso associati. Poiché elementi come il profilo di flusso, il tipo di liquido e il materiale del tubo possono influire sulla misura, l'elettronica del misuratore di portata compensa e si adatterà alle variazioni del mezzo al fine di fornire misure affidabili.

Benefici
Nessun Taglio e Nessuna Saldatura quindi Nessun Costo.
Nessun rischio di perdite o contaminazione del tubo e dell'area circostante
Adatto a un'ampia gamma di tipi di liquidi, semplice misura del flusso indipendentemente da pressione e conduttività, risultati ripetibili e precisi anche in condizioni non ideali
Installazione rapida, economica e semplice, prodotti robusti e di qualità progettati per essere semplici da configurare e utilizzare.

1.7 Possibili applicazioni in ambito laboratoriale

MISURATORI DI FLUSSO PER WORKSTATION ELISA AUTOMATICO

ELISA è l'acronimo di enzyme-linked immunosorbent assay (saggio immuno-assorbente legato ad un enzima). In questo tipo di saggio una sostanza da dosare (definita analita) si lega ad un'altra (generalmente rappresentata da un anticorpo) che ne rileva la presenza. Il metodo ELISA ha come fine il rilevamento e l'identificazione (sia qualitativa che quantitativa) di una sostanza specifica all'interno di un campione.

Prima del suo sviluppo, l'unica opzione per condurre un saggio immunologico era riconducibile ad una tecnica che utilizza antigeni o anticorpi marcati radioattivamente. Poiché la radioattività pone una potenziale minaccia per la salute, il test ELISA ha rappresentato un'alternativa più sicura.

Il test ELISA si basa sull'utilizzo di anticorpi marcati con un enzima (generalmente la perossidasi), in modo che i coniugati risultanti abbiano una attività sia immunologica che enzimatica. Avendo uno dei componenti (antigene o anticorpo) adeso alla piastra, la reazione antigene-anticorpo è immobilizzata e pertanto potrà facilmente essere evidenziata con l'addizione del substrato che, reagendo con l'enzima, produrrà una colorazione osservabile a vista e quantificabile con lo spettrofotometro.

Per determinare la presenza di un determinato antigene in un campione biologico, è necessario seguire delle fasi consequenziali:

Fissaggio al substrato dell'anticorpo primario specifico per l'antigene da ricercare;

Lavaggio per eliminare l'eccesso di anticorpo;

Aggiunta del campione da saggiare per la presenza o meno dell'antigene;

Lavaggio per togliere antigeni in eccesso;
Aggiunta di un anticorpo secondario specifico per l'antigene coniugato a un enzima specifico (perossidasi o fosfatasi alcalina). Se l'antigene è presente l'anticorpo si legherà selettivamente al complesso anticorpo-antigene, formando un triplo strato da cui prende il nome il test;
Lavaggio con soluzione tampone dell'eccesso;
Aggiunta del substrato per l'enzima coniugato all'anticorpo che porterà alla formazione di un prodotto colorato;

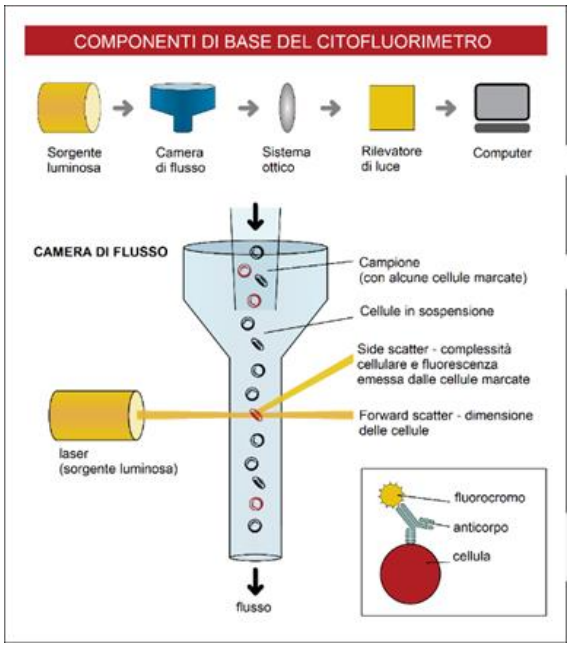
I sensori di flusso sono utilizzati per automatizzare l'intero flusso di lavoro ELISA, se necessario, dal lavaggio, la dispensazione, agitazione, incubazione alla misura dell'assorbanza finale. L'automazione può risolvere alcuni dei tipici problemi con saggi ELISA aumentando l'affidabilità e la riproducibilità, aumentando la produttività, liberando il personale da attività ripetitive e consentendo di concentrarsi su attività più importanti.

MISURATORI DI FLUSSO PER CITOFLUORIMETRIA

La citofluorimetria, o citometria, è una tecnologia di laboratorio che consente la valutazione delle caratteristiche e del numero delle cellule presenti in un campione. Il citofluorimetro è sufficientemente sensibile da analizzare cellule o particelle del diametro di un micron e in un tempo relativamente breve.
Queste informazioni possono essere raccolte grazie a dei protocolli di identificazione che prevedono la valutazione delle caratteristiche fisiche delle cellule ma anche della presenza o assenza di particolari marcatori, gli antigeni, presenti sulla superficie o all'interno delle cellule stesse. Questo metodo consente di studiare le cellule presenti nel circolo ematico, nel midollo osseo, nei fluidi corporei come il liquido cefalorachidiano, o le cellule tumorali.

Un campione di cellule viene sospeso in un fluido, prima del test e sulla base delle cellule da analizzare, il campione viene trattato con coloranti specifici in grado di discriminare i sottotipi cellulari.

Il campione contenente le cellule marcate viene introdotto nello strumento chiamato citofluorimetro.
Nello strumento il fluido contenente le cellule viene incanalato nella camera di flusso e quindi attraverso un foro molto stretto, in modo tale da creare un flusso all'interno del quale le cellule risultano organizzate in fila, una dietro l'altra. Il flusso di cellule viene posto davanti ad un rilevatore, che così analizza ciascuna cellula presente all'interno del flusso ad una velocità altissima.



Il citofluorimetro contiene uno o più laser e più rilevatori in grado di identificare alcune caratteristiche, uniche per ciascuna cellula. Ciascun laser colpisce le cellule presenti nel flusso, generando per ciascuna di esse uno scatter caratteristico e dipendente dalle caratteristiche della stessa. Le caratteristiche possono essere fisiche (dimensione e complessità cellulare) o possono dipendere dal segnale generato dal colorante (il fluorocromo) intercettato dal laser. La combinazione di queste informazioni genera un profilo caratteristico per ciascuna cellula presente all'interno del campione.
Il segnale rilevato dai rilevatori (o detector) viene amplificato (dai fotomoltiplicatori) e inviato al computer. Qui viene convertito in formato digitale e mostrato sul computer o stampato.
La stazione per la citofluorimetria per essere automatica e indipendente deve possedere sei sistemi di flusso o di portata in modo da saper esattamente quanto flusso deve essere analizzato durante il processo.

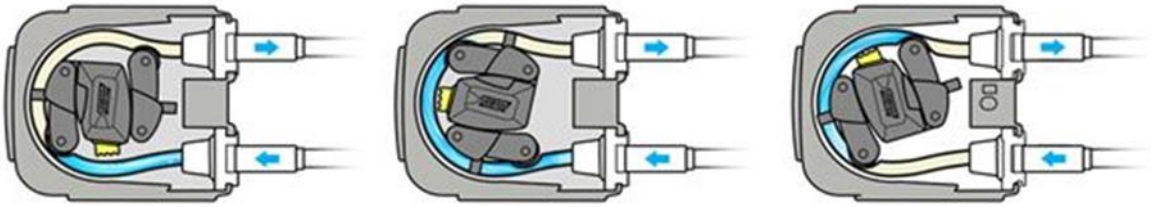
MISURATORI DI FLUSSO PER CONTROLLARE IL FLUSSO DI POMPE PERISTALTICHE

La pompa peristaltica è una pompa volumetrica usata per il pompaggio di una varietà di fluidi.

Il fluido è contenuto all'interno di un tubo dentro al corpo pompa. L'attuale principio di pompaggio, chiamato peristalsi, si basa sulla compressione alternata e sul rilassamento del tubo, che aspira il contenuto e lo pompa fuori.

Un rullo rotante passa lungo la lunghezza del tubo, crea una tenuta temporanea tra i lati di aspirazione e di scarico della pompa. Quando il rotore della pompa gira, questa pressione di tenuta si sposta lungo il tubo e costringe il prodotto a spostarsi dalla pompa alla linea di scarico. Quando la pressione è stata rilasciata, il tubo si riprende creando un vuoto che attira il prodotto nel lato di aspirazione della pompa, il meccanismo di innesco.

Combinando questi principi di aspirazione e scarico si ottiene una potente azione autoadescante.



Il perfetto isolamento tra liquido e pompa, accoppiato con la linearità di flusso, rende la pompa peristaltica ideale per il dosaggio.

Associarla ad un sensore di flusso è il metodo migliore per avere una misura più precisa della quantità di liquido.

Il liquido pompato non viene mai a contatto con nessuna parte mobile perché è completamente contenuto nel tubo. Ciò rende la pompa peristaltica una soluzione di pompaggio igienica con zero possibilità di contaminazione e riduce i tempi di manutenzione poiché il tubo è l'unica parte usurabile.

MISURATORI DI FLUSSO PER OSSIGENOTERAPIA

Con il termine ossigenoterapia si intende la somministrazione di ossigeno come parte integrante di un trattamento medico, il quale può essere attuato per una varietà di scopi sia nel trattamento di un paziente con insufficienza respiratoria cronica sia in un paziente affetto da insufficienza respiratoria acuta. Alte concentrazioni di ossigeno a livello tissutale possono essere utili o dannose, a seconda delle circostanze, e l'ossigenoterapia viene utilizzata per arrecare beneficio al paziente, aumentando l'apporto di ossigeno ai polmoni e quindi aumentando la disponibilità di ossigeno nei diversi tessuti biologici dell'organismo, specialmente quando il paziente è affetto da ipossia e/o ipossiemia.

Sono diversi i dispositivi che risultano utilizzabili per la somministrazione di ossigeno. Nella maggior parte dei casi, l'ossigeno passa prima attraverso un regolatore di pressione, utilizzato per controllare l'elevata pressione di ossigeno erogato da una bombola portandola a una pressione inferiore.

Questa pressione inferiore è poi controllata da un misuratore di portata, il quale può essere preimpostato o selezionabile. Scopo del misuratore di portata è controllare il flusso, che viene misurato in numero di litri al minuto.

I misuratori di portata tipicamente destinati all'ossigeno a scopi medici erogano valori di portata compresi tra 0 e 15 litri/minuto, ma esistono anche alcuni misuratori particolari che si spingono fino a 25 litri al minuto.

2. OBIETTIVO

L'obbiettivo principale di questo lavoro di tesi è cercare di offrire il supporto ingegneristico di cui necessita questo interfacciamento tra microcontrollore PIC e sensore di flusso, i quali abbinati insieme hanno un grandissimo potenziale.
Su tali necessità verrà sviluppato e realizzato un dispositivo di rilevazione ad hoc impiegando una componentistica con determinate caratteristiche tecniche.
Lo strumento permetterà di rilevare tramite l'onda quadra in ingresso due fronti di salita e quindi riuscire a calcolare la frequenza.
Questo segnale in frequenza è linearmente proporzionale alla portata.
Ciò verrà fatto tramite la realizzazione di un frequenzimetro (o un periodimetro) e in fine rappresentato su un display grafico.
La difficoltà iniziale sarà quella di riuscire ad interfacciare il microcontrollore PIC con sensori di flusso, creando un programma che data un onda quadra in ingresso, riesca a calcolarne la frequenza.
Verificare la linearità sarà la principale conclusione che faremo in quanto potrebbero essere presenti errori di vario tipo.
Infine si potrebbe utilizzare tutto ciò per riuscire, retro azionando questo dato, a controllare una pompa in modo da versare una quantità predeterminata e precisa di liquido.

3. MATERIALI E METODI

3.1 PIC pic18f4550

Ecco elencate le caratteristiche generali dal PIC18F4550.

Numero di parte PIC18F4550-I/P

Fabbricante Microchip Technology

Descrizione IC MCU 8BIT 32KB FLASH 40DIP

Categoria Circuiti integrati (CI) > Embedded - Microcontrollori

Stato parte 1446 pcs Stock

Tensione di alimentazione - (Vcc / Vdd) 4.2 V ~ 5.5 V

Contenitore dispositivo fornitore 40-PDIP

Velocità 48MHz

Serie PIC 18F

Dimensioni RAM 2K x 8

Tipo di memoria di programma FLASH

Dimensioni memoria programma 32KB (16K x 16)

Periferiche Brown-out Detect/Reset, HLVD, POR, PWM, WDT

Imballaggio Tube

Contenitore / involucro 40-DIP (0.600", 15.24mm)

Tipo oscillatore Internal

Temperatura di esercizio -40°C ~ 85°C (TA)

Numero di I / O 35

Dimensioni EEPROM 256 x 8

Convertitori di dati A/D 13x10b

Dimensione nucleo 8-Bit

Core Processor PIC

Connettività I²C, SPI, UART/USART, USB

Il PIC 18F4550 ha una CPU a 8bit che dispone di un moltiplicatore hardware per un calcolo più rapido degli algoritmi di controllo e interfacce di comunicazione: USART, SPI, I²C, CAN e USB.

Il modello che abbiamo utilizzato durante il tirocinio è il 18F4550 a 40pin

.

Altri moduli di rilevante importanza ai fini dell'utilizzo di questo microcontrollore sono: - SPI - I 2 C - Clock - Timer - PWM - ADC – USB

Device	Program Memory		Data Memory		I/O	10-Bit A/D (ch)	CCP/ECCP (PWM)	SPP	MSSP		EAUSART	Comparators	Timers 8/16-Bit
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)					SPI	Master I^2C™			
PIC18F2455	24K	12288	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F2550	32K	16384	2048	256	24	10	2/0	No	Y	Y	1	2	1/3
PIC18F4455	24K	12288	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3
PIC18F4550	32K	16384	2048	256	35	13	1/1	Yes	Y	Y	1	2	1/3

Uno svantaggio di questo PIC è che non può essere interfacciato con MCC (Microchip Code Configurator), un tool in grado di semplificare notevolmente la progettazione e la stesura del codice.

3.2 Programmatore pickit3

La Microchip Technology ha sviluppato a tal proposito un'applicazione software chiamata MPLAB.

MPLAB X Integrated Development Environment (IDE) è un programma software espandibile e altamente configurabile che incorpora potenti strumenti per aiutarti a scoprire, configurare, sviluppare, eseguire il debug e qualificare progetti embedded per la maggior parte dei nostri microcontrollori e controller di segnali digitali. MPLAB X IDE funziona perfettamente con l'ecosistema di sviluppo di software e strumenti MPLAB, molti dei quali sono completamente gratuiti.

Si tratta di un ambiente di sviluppo integrato (IDE) per la programmazione dei PIC a 8bit, 16bit e 32bit. Oggi disponibile per macOS e Linux oltre che per Windows.

MPLAB supporta diversi compilatori in continuo sviluppo, alcuni disponibili per la programmazione con linguaggio Assembly e altri in C, tra i più usati troviamo:

- MPLAB XC8
- MPLAB XC16
- MPLAB XC32

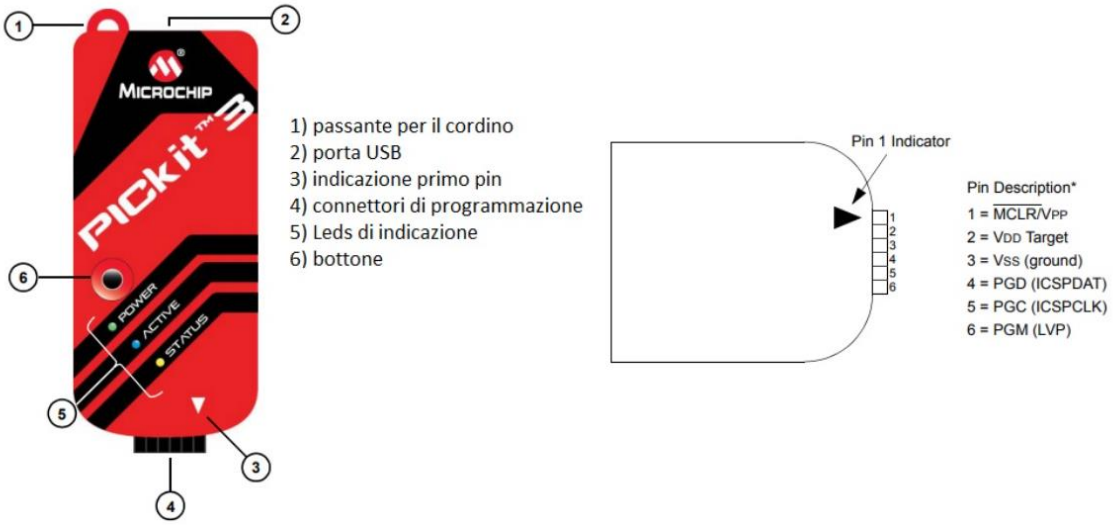
Il linguaggio di programmazione è il C per tutti i compilatori, e la versione XC32 dispone anche del C++.

In alternativa per la programmazione in C dei PIC è disponibile anche il compilatore MikroC Pro (MikroElektronika). Il vantaggio di questo compilatore è la semplicità di utilizzo e le tante librerie già pronte a disposizione, vantaggi che rendono però questo software a pagamento, mentre MPLAB è completamente gratuito. Purtroppo, MikroC Pro è disponibile solo per Windows.

Per quanto riguarda i programmatori la società produttrice di PIC offre la famiglia dei PICkit utilizzati per la programmazione di microcontrollori e memorie EEPROM.

I modelli esistenti vanno dal PICkit 1, oramai completamente inutilizzato, al PICkit 4 il quale è stato rilasciato solo da qualche anno e offre una completa compatibilità con le versioni più recenti di MPLAB.

Il PICKIT 3 è un programmatore/debugger controllato da PC tramite porta USB che permette di programmare o far un controllo di funzionalità dei microcontrollori senza rimuoverli dal circuito a cui sono connessi.
Si possono programmare una vasta gamma di PIC e dsPIC. L'aspetto e la connessione per la programmazione è la seguente:



I LEDs sono 3. Il LED POWER indica che il pickit3 è alimentato tramite USB, il LED ACTIVE indica che c'è un link attivo di comunicazione tra il PC e il programmatore mentre il LED STATUs può avere 3 colori diversi, verde indica che il programmatore è pronto a partire o la programmazione è andata a buon fine, se il LED è arancione la programmazione è in corso mentre se diventa rosso vuol dire c'è stato un errore.
Il bottone invece serve per la funzione Programmer-To-Go. La funzionalità Programmer-To-Go del PICkit 3 consente di creare un'immagine della memoria del PIC copiato all'interno del PICkit 3 per la programmazione successiva in un altro microcontrollore. Non è necessario il software o il PC per programmare i dispositivi una volta impostato il programmatore PICkit 3 per la funzione sopracitata. È sufficiente una fonte di alimentazione USB.

Prima di tutto abbiamo bisogno di alcune cose:

- Il programmatore PICKIT3 con il relativo cavo USB

- Una breadboard con i cavetti per collegare il microcontrollore con il programmatore
- Il software rilasciato da microchip MPLAB IDE

Una volta scaricato e installato il programma bisogna aprire MPLAB IPE.

3.3 Sensore di flusso YF-S201

I misuratori di flusso per liquidi ad effetto Hall sono in grado di misurare il flusso, ovvero la quantità in litri in un certo tempo, di liquidi come acqua, benzina, olio o simili. Non sono in grado di misurare il peso ma solo il volume anche se queste due grandezze in alcuni liquidi come l'acqua coincidono.

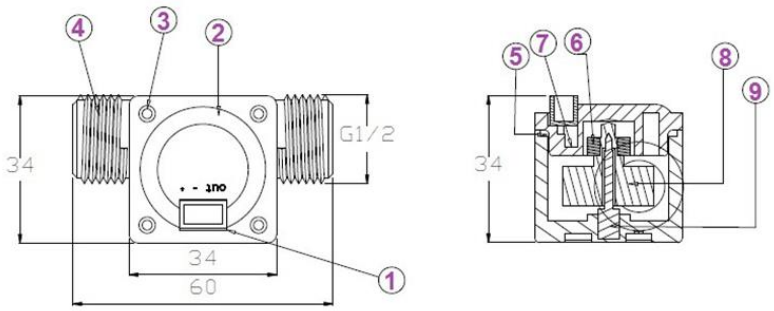
In particolare i misuratori di flusso ad effetto Hall utilizzano un girante, ovvero delle palette disposte lungo il raggio di un cerchio, per creare una rotazione con il flusso del liquido, sul girante vi è un elemento magnetico e il sensore ad effetto Hall rileva la variazione del campo magnetico fornendo un segnale di tensione.

Il sensore ad effetto Hall all'interno è digitale, quindi fornisce un livello logico alto quando la pala è vicina ad esso, basso altrimenti.

FUNZIONAMENTO DEL SENSORE YF-S201

Il misuratore sotto esame ha il seguente aspetto:

Il misuratore di flusso YF-S201 ha un solo magnete in cima al rotore e questo magnete girando "mostra" al sensore Hall o il polo nord o il polo sud, quindi il sensore fornisce in uscita un segnale alto o basso in base al polo magnetico. Quindi in questo caso si contano perfettamente i giri del girante. La geometria interna ed esterna del modulo è la seguente:

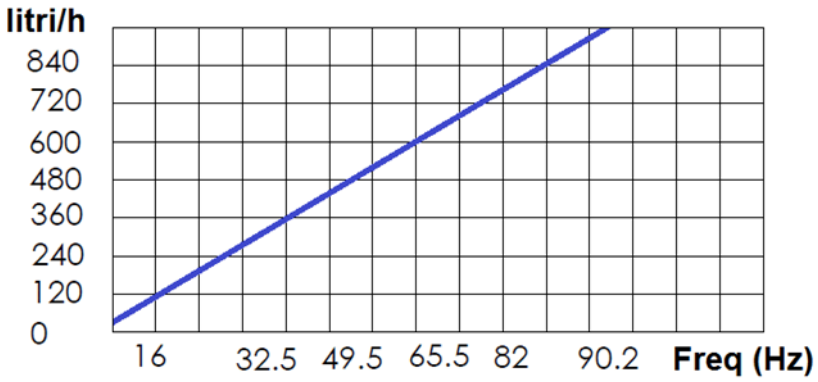


Gli attacchi per il flusso in ingresso e in uscita sono da ½ pollice e non vi è una direzione univoca di ingresso, quindi si possono collegare senza fare nessuna attenzione e inoltre si possono misurare sia il flusso entrante che uscente.
I tre fili, rosso, nero e giallo, sono rispettivamente l'alimentazione positiva e negativa e il segnale di uscita ad onda quadra. Per il fatto che il modulo misura il flusso in ambedue le direzioni si potrebbero avere errori di misura (ad esempio vi è un risucchio di acqua o aria nel sistema), quindi si consiglia di utilizzare una valvola di ritegno.

CARATTERISTICHE ELETTRICHE E IDRAULICHE
Le caratteristiche elettriche del modulo e le due caratteristiche di misura sono le seguenti:
Tensione di alimentazione da 5V fino ad un massimo di 18V
Corrente assorbita 15mA con alimentazione a 5V
Segnale di uscita a 5V con logica TTL
Flusso minimo di 1 litro al minuto e flusso massimo di 30 litri al minuto
Temperatura di funzionamento da -25°C a +80°C
Accuratezza del ±10%
Pressione massima dell'acqua di 20Mpa
Rise time segnale di uscita 40ns, Fall time di 180ns
Connessione tubi da ½ con diametro interno di ¾ di pollice
Dimensioni (lunghezza, larghezza, altezza) di 6.4×3.5×3.5

La tensione di alimentazione va da 5V a 18V, se si esce dal range o il modulo non funziona o si brucia, si deve garantire una corrente di 15mA con 5V di alimentazione mentre varia di 2mA circa se maggiore. L'uscita non dipende dalla tensione di alimentazione e i livelli logici sono 0-5V. I flussi minori di 1 litro al minuto non vengono rilevati mentre quelli maggiori di 30 litri al minuto potrebbero danneggiare il sensore o avere un elevato errore di misura. La temperatura di funzionamento arriva fino a -25°C anche se si sconsiglia di far ghiacciare l'acqua al suo interno.
L'errore del sensore è di circa il 10% in meno o in più, il che vuol dire che lui fornisce in uscita un segnale che indica ad esempio 10 litri al minuto ma in realtà possono essere 11 litri o 9 litri al minuto. Se si misura la quantità di liquido, invece, il sensore può aver misurato 1 litro, potrebbero essere 900ml o 1.1 litri. L'errore è abbastanza grande se si

vogliono applicazioni di precisione, quindi converrebbe tarare il sensore sul microcontrollore, anche se l'errore può variare nel tempo. Il segnale ad onda quadra in uscita ha un fronte di salita di 40ns mentre uno di discesa di 180ns.
Per quanto riguarda la caratteristica elettrica essa è la seguente:



La caratteristica è lineare e indica che con un flusso di 120 litri l'ora (ovvero 2 litri al minuto) il modulo ha un segnale di uscita a 16Hz, 32.5Hz con 240 litri l'ora e 90.2Hz con 720 litri l'ora. In altre parole, l'equazione che lega la frequenza con il flusso è la seguente:
Flusso (litri al minuto) = Frequenza (Hz) / 7.5 .
Quindi misurando la frequenza e dividendo per la costante 7.5 posso ottenere il valore del flusso espresso in litri al minuto. L'altra soluzione elettrica per misurare il volume del liquido è quello di contare il numero di impulsi, ovvero contare i fronti di salita visto che 450 fronti di salita corrispondono ad un litro, quindi:
Volume liquido (litri) = Fronti di salita contati / 450 .
Quindi ad esempio se il microcontrollore conta 39000 fronti di salita vuol dire che sono passati attraverso il misuratore di flusso un volume di 86.6 litri. Considerando anche l'errore del 10% la quantità effettiva è compresa tra 77.94 e 95.26 litri.

4. RISULTATI

4.1 Programma

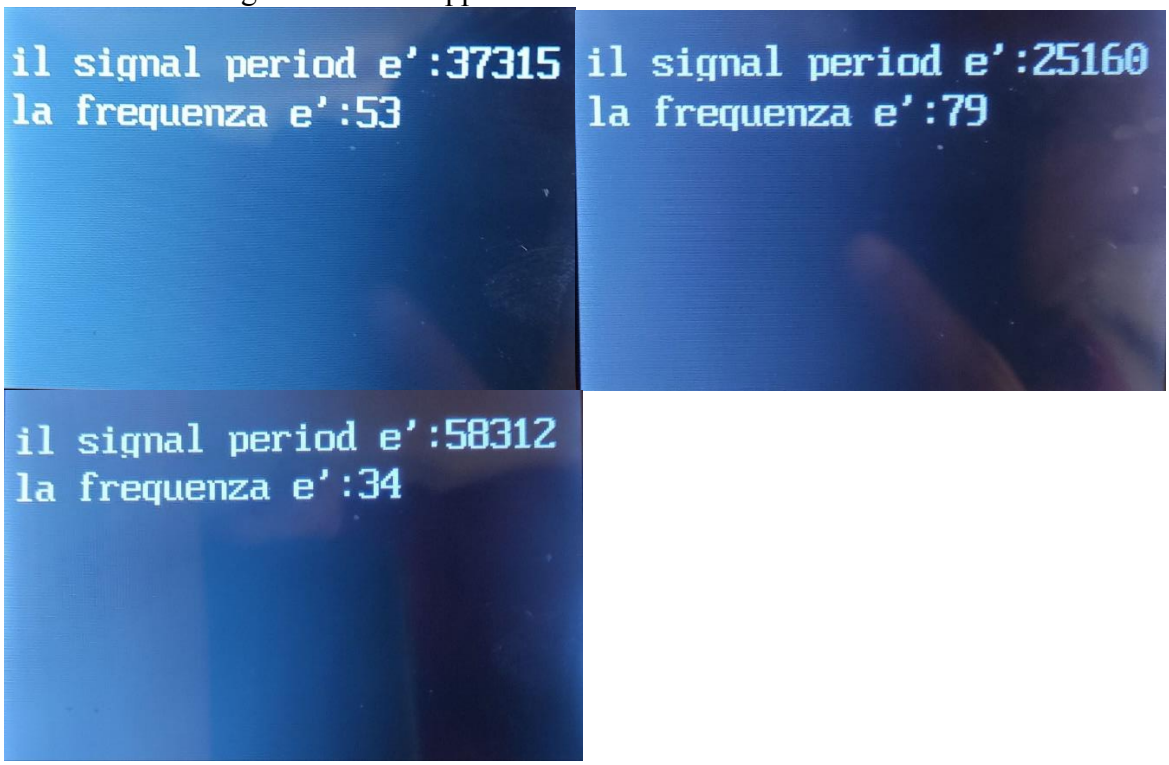
L'obiettivo era quello di progettare un'applicazione utilizzando PIC18F4550 in cui misureremo la frequenza del segnale di ingresso e la visualizzeremo su un display ili9488.

Il segnale di ingresso è collegato al pin CCP1 (RC2) per la misurazione della frequenza.

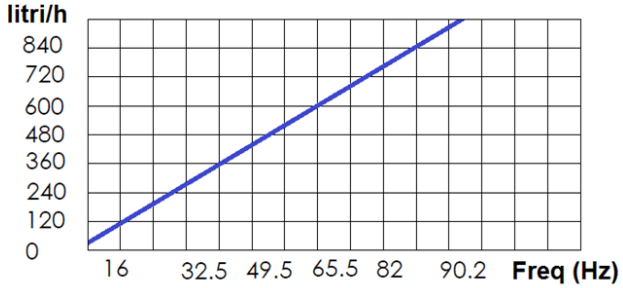
```
#include <xc.h>
#include "Conf.h"
#include "Def.h"
#include "Add_Lib.h"
#include "GFX_Lib.h"
#include "Set_Lib.h"
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define f_timer 2000000

void main()
{
   Set_Register();

   Init_ILI9488();

   Set_Background();

   display_setTextColor(200, 200, 200, 0, 0, 0);
   display_setTextSize(2);
   display_setTextFont(8, 16, 2);

   fillScreen(0, 0, 0);

   __delay_ms(1000);
   CCP1CON = 0x05;      /* La modalità di acquisizione è selezionata per rilevare il fronte di salita */
   CCPR1 = 0x00;        /* Svuota il registro CCPR1 per il conteggio*/
   T1CON = 0x80;        /* Abilita il registro TMR1 a 16 bit, nessuna pre-scale, usa l'orologio
interno, timer OFF */

   unsigned long signal_period,data1,data2;
   float Frequency;
   TRISCbits.TRISC2=1;
   OSCCON=0x72;      /* imposta il clock interno a 8 MHz */

   PIE1bits.CCP1IE=1;
   PIR1bits.CCP1IF=0;
   CCP1CON=0x05;      /* La modalità di acquisizione è selezionata per rilevare il fronte di salita */
   CCPR1=0x00;       /*CCPR1 è il registro del conteggio delle catture che viene cancellato
inizialmente*/
   TMR1IF=0;
   T1CON=0x80;  /* Abilita registro TMR1 a 16 bit, nessuna pre-scale, usa l'orologio interno, timer
OFF */
```

```
   TMR1=0;
         TMR1ON=1; /* Avvio Timer1 */
   while(1)
   {
      while(!(PIR1bits.CCP1IF));
/*Attendi il flag di interruzione che viene generato quando viene rilevato un fronte*/
      PIR1bits.CCP1IF=0;
      data1 = CCPR1;          /*Conteggio copie del primo fronte rilevato*/

      while(!(PIR1bits.CCP1IF));  /*Attendi il flag di interruzione che viene generato quando viene
rilevato il fronte*/
      PIR1bits.CCP1IF=0;
      data2 = CCPR1;          /*Conteggio copie del secondo fronte rilevato*/

      if(data1 < data2)
        {

         /*Calcolo per la misurazione della frequenza*/
         signal_period = data2 - data1;
         Frequency = (f_timer / signal_period);

         display_setCursor(10, 30);
          display_puts("il signal_period e':");
   __delay_ms(100);
         display_putn(signal_period);
         display_setCursor(10, 70);
          display_puts("la frequenza e':");
   __delay_ms(100);
         display_putn(Frequency);

         }
      TMR1=0;
   }
}
```

Ecco alcune immagini di ciò che appare a schermo.



Come si può vedere i valori di frequenza che appaiono sul display, sono dei valori accettabili.
Questi valori di frequenza sono in linea con le caratteristiche del sensore descritte sopra.



Date la seguenti formula è possibile calcolare il valore del flusso nelle tre casistiche.

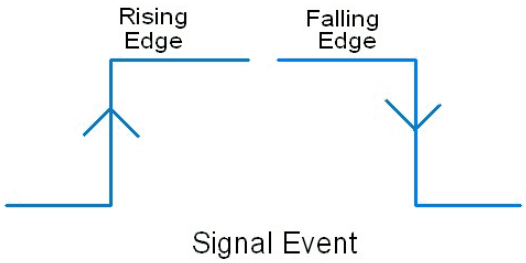
Flusso (litri al minuto) = Frequenza (Hz) / 7.5

Frequenza1= 53 Hz	flusso1=53/7,5= 7,066*60= 424 litri/h
Frequenza2= 79 Hz	flusso2=79/7,5= 10,53*60= 632 litri/h
Frequenza3= 34 Hz	flusso3=34/7,5= 4,53*60= 272 litri/h

4.2 Impostazioni Capture CCP

Input Capture è ampiamente utilizzato in molte applicazioni come: catturare l'ora di arrivo di un evento, misurazione dell'ampiezza dell'impulso, periodo di misurazione.

Un evento può essere rappresentato dai fronti di salita e di discesa di un impulso.



L'ora in cui si verifica un evento può essere registrata bloccando il conteggio quando viene rilevato un fronte (in salita o in discesa). Questo può essere fatto utilizzando la modalità di acquisizione dell'input nei microcontrollori.

PIC18F4550 ha un modulo CCP (Capture / Compare / PWM) integrato che ha la modalità di acquisizione dell'ingresso . La cattura in ingresso del segnale ha applicazioni come la ricerca della frequenza, il calcolo del duty cycle del segnale in ingresso, ecc.

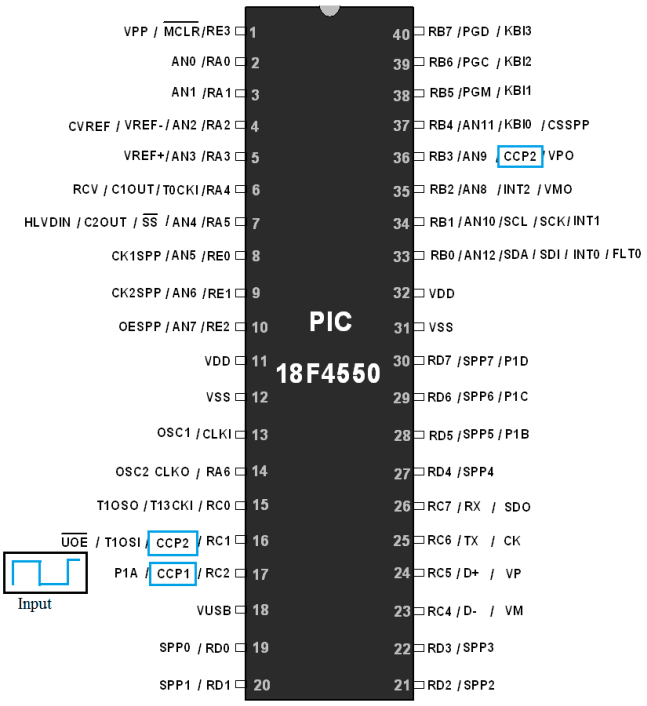
Vediamo la modalità di acquisizione nel microcontrollore PIC 18F4550 .

PIC18F4550 ha due moduli CCP

1. PCC1
2. PCC2

Qui discuteremo del modulo CCP1.

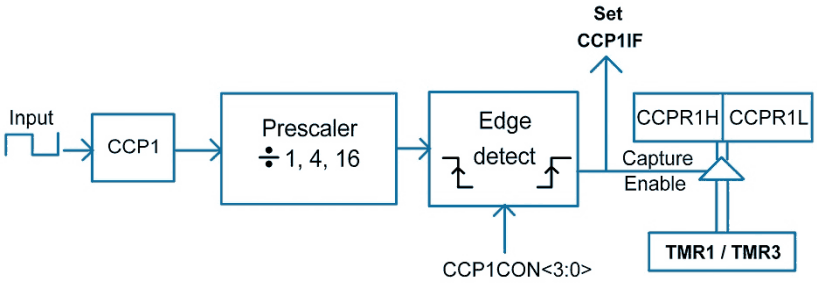
In modalità di acquisizione, l'impulso di ingresso deve essere collegato a CCP1 che è multiplexato con PORTC.2 (RC2) come mostrato di seguito e configurarlo come pin di ingresso.



Il modulo CCP1 è in grado di utilizzare solo i seguenti due timer:

1. Timer1
2. Timer3

Funzionamento della modalità di cattura



Capture Mode Block Diagram

Schema della modalità confronto PIC

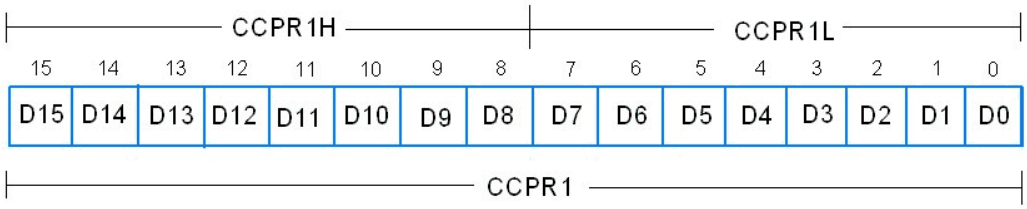
- Timer1 o Timer3 vengono inizializzati per contare il valore nell'operazione di acquisizione.
- Il registro T3CON viene utilizzato per selezionare il Timer1 / Timer3 per la cattura dell'ingresso mostrato sopra.

- Quando viene rilevato il fronte di salita o di discesa sul pin CCP1, viene impostato il bit del flag di interrupt CCP1IF. Viene cancellato tramite software.
- Quando si verifica un interrupt, il valore del timer viene copiato nel registro CCP CCPR1 (CCPR1H e CCPR1L), e guardando questo valore siamo in grado di trovare l'ora di arrivo di quell'evento.

Registri della modalità di cattura (CCP)

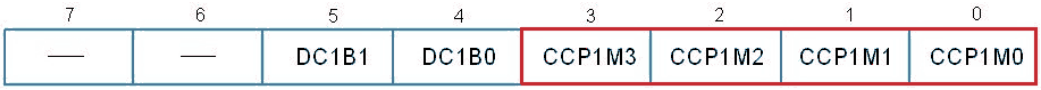
In PIC18F4550, vengono utilizzati diversi registri per la modalità di acquisizione.

1. Il registro CCPR1 è un registro di cattura di eventi a 16 bit in cui viene copiato l'evento catturato da timer1 o timer3, ovvero il registro TMR1 o TMR3. Questo viene fatto dall'hardware stesso. Il registro CCPR1L e il registro CCPR1H sono i due registri a 8 bit utilizzati rispettivamente per la memorizzazione di byte inferiori e superiori.



2. Registro CCP1CON: registro di controllo CCP

Questo è un registro a 8 bit utilizzato per selezionare la modalità di acquisizione.



CCP1M3 : CCP1M0: Bit di selezione della modalità CCP1

0100 = La modalità di cattura viene selezionata per ogni fronte di discesa rilevato.

0101 = La modalità di cattura viene selezionata per ogni fronte di salita rilevato.

0110 = La modalità di cattura viene selezionata ogni 4 fronti di salita rilevati.

0111 = La modalità di cattura viene selezionata ogni 16° fronte di salita rilevato.

Flag di interruzione CCP1IF

- Dopo che si verifica un evento di cattura, viene impostato il flag di interruzione.
- Questo flag di interrupt è CCP1IF (flag di interrupt CCP) che si trova nel registro PIR1.

CCP1IF :

1 = si è verificata la cattura del registro TMR1 o TMR3 (sul fronte di discesa o di salita dell'impulso).

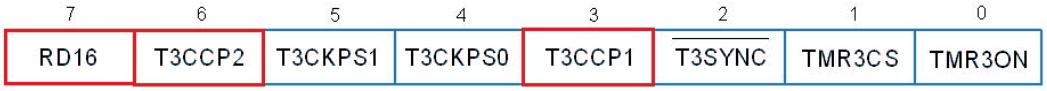
0 = non si è verificata alcuna acquisizione TMR1 o TMR3.

Dobbiamo abilitare questo flag (CCP1IF) abilitando il bit CCP1IE (bit di abilitazione interrupt CCP) nel registro PIE1 .

Registro PIE1: abilitazione interrupt periferica

Registro T3CON: registro di controllo e stato Timer3

Il registro T3CON viene utilizzato per selezionare un timer per la modalità di acquisizione. Questo è mostrato di seguito



T3CCP2: T3CCP1 (bit 6 e bit 3)

Questi due bit in combinazione vengono utilizzati per la selezione della modalità Timer e Cattura.

00 = Timer1 è la sorgente di clock di acquisizione/confronto per entrambi i moduli CCP.

01 = Timer3 è la sorgente di clock di cattura/confronto per il CCP2,

Timer1 è la sorgente di clock di cattura/confronto per il CCP1.

1x = Timer3 è la sorgente di clock di cattura/confronto per entrambi i moduli CCP.

RD16 :

Questo bit viene utilizzato per specificare che il registro Timer3 è a 16 bit (TMR3) o due a 8 bit (TMR3H e TMR3L).

RD16 = 1 , Abilita un registro a 16 bit.

RD16 = 0 , Abilita due registri a 8 bit.

L'impostazione di questo bit è richiesta solo quando selezioniamo Timer3 per la modalità di acquisizione. Altrimenti, RD16 nel registro T1CON viene utilizzato per Timer1.

Passaggi per la programmazione della modalità di acquisizione (CCP).

Inizializzazione della modalità Cattura PIC18F4550:

1. Inizializzare il registro CCPCON1 per acquisire il fronte di salita o di discesa.
2. Configurare il pin CCP1 come pin di input.
3. Se si utilizza Timer3, configurare il registro T3CON per selezionare Timer3 per la cattura dell'input; in caso contrario, Timer1 è un valore predefinito.
4. Inoltre, impostare il valore del registro T1CON o T3CON per decidere se il registro del timer deve essere due a 8 bit (TMRxL e TMRxH) o a 16 bit (TMRx).
5. Impostare i registri CCPR1 e TMR3 / TMR1 su 0.

Operazione di cattura

1. Attiva il timer.
2. Monitora fino al flag CCP1IF==1, che viene impostato quando si verifica la cattura.
3. Copia il valore del registro CCPR1 in qualsiasi variabile di dati e ottieni un orario in cui si è verificato l'evento.

5. CONCLUSIONI

Si è osservato che questi microcontrollori hanno un impiego assai maggiore e fondamentale per qualunque aspetto ingegneristico e non solo.
Le conclusioni a cui si è giunti, sono che la selezione del PIC più adatto alle proprie esigenze e caratteristiche di progetto è fondamentale.
Quindi innanzitutto si deve valutare la qualità del lavoro da svolgere, poi in secondo luogo il tempo, poiché programmare un PIC con il software gratuito Mplab, lo stesso usato durante l'arco del tirocinio, è molto dispendioso in termini di tempo, in quanto l'ingegnere dovrà programmare da solo con conseguente scrittura di tutte le librerie utili e del programma da utilizzare.
L'aspetto economico inoltre è importante, essendo la strumentazione molto fragile può capitare che alcune componentistiche che siano PIC, ram o semplicemente fili di collegamento possono rompersi e causare un malfunzionamento, ciò porta a un aumento di tempo impiegato con possibili ritardi nel progetto e una spesa maggiore.
L'alimentazione giusta è fondamentale in quanto valori di tensione non adeguate possono portare a surriscaldamento o rottura di periferiche.
Purtroppo la quantità di nozioni sulla programmazione di questi microcontrollori è piuttosto limitata quindi ciò porta automaticamente a una capacità di ricerca indipendente.

6. APPENDICE

Si può trovare in appendice parte delle librerie utilizzate.

CONF

```
#pragma config PLLDIV = 1
#pragma config CPUDIV = OSC1_PLL2
#pragma config USBDIV = 1

#pragma config FOSC = INTOSC_HS
#pragma config FCMEN = OFF
#pragma config IESO = OFF

#pragma config PWRT = OFF
#pragma config BOR = OFF
#pragma config BORV = 3
#pragma config VREGEN = OFF

#pragma config WDT = OFF
#pragma config WDTPS = 32768

#pragma config CCP2MX = OFF
#pragma config PBADEN = ON
#pragma config LPT1OSC = OFF
#pragma config MCLRE = OFF

#pragma config STVREN = OFF
#pragma config LVP = OFF
#pragma config ICPRT = OFF
#pragma config XINST = OFF

#pragma config CP0 = OFF
#pragma config CP1 = OFF
#pragma config CP2 = OFF
#pragma config CP3 = OFF

#pragma config CPB = OFF
#pragma config CPD = OFF

#pragma config WRT0 = OFF
#pragma config WRT1 = OFF
#pragma config WRT2 = OFF
#pragma config WRT3 = OFF

#pragma config WRTC = OFF
#pragma config WRTB = OFF
#pragma config WRTD = OFF

#pragma config EBTR0 = OFF
#pragma config EBTR1 = OFF
#pragma config EBTR2 = OFF
#pragma config EBTR3 = OFF

#pragma config EBTRB = OFF
```

DEF

```
#define LCD_RST            PORTBbits.RB6
#define LCD_CS             PORTBbits.RB5
#define LCD_RS             PORTBbits.RB4
#define LCD_WR             PORTCbits.RC7
#define LCD_RD             PORTCbits.RC6

#define LCD_WR_Direction  TRISCbits.TRISC7
#define LCD_RD_Direction  TRISCbits.TRISC6
#define LCD_CS_Direction  TRISBbits.TRISB5
#define LCD_RS_Direction  TRISBbits.TRISB4
#define LCD_RST_Direction TRISBbits.TRISB6

#define _XTAL_FREQ 2000000

#define TFTWIDTH   320
#define TFTHEIGHT  480
```

SET_LIB

```
#include "Def.h"

void Set_Register()
{
    OSCCON = 0b11110000; //8 Mhz, oscillatore interno
    OSCTUNE = 0b10001111;  //che dovrebbe darti la massima frequenza e
quindi il postscaler è impostato ad 1

    TRISB = 0b00000000;
     TRISD = 0b00000000;
     LATD = 0b00000000;
     PORTD = 0b00000000;
     TRISC = 0b00000000;

     INTCONbits.GIE=0;

     ADCON0 = 0b00000001;
     ADCON1 = 0b00001110;
     ADCON2 = 0b10111110;
}

void Init_ILI9488()
{
     LCD_WR_Direction = 0;
     LCD_RS_Direction = 0;
     LCD_RD_Direction = 0;
     LCD_CS_Direction = 0;
     LCD_RST_Direction = 0;

     LCD_CS = 1;
     LCD_RST = 1;
     LCD_RS = 1;
     LCD_RD = 1;
     LCD_WR = 1;

     LCD_RST = 0;
     __delay_ms(50);
```

```
LCD_RST = 1;

LCD_CS = 0;
Send_Command_8bit(0x01);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x11);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x36);
Send_Data_8bit(0xE8);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x3A);
Send_Data_8bit(0x76);
LCD_CS = 1;
__delay_ms(100);

LCD_CS = 0;
Send_Command_8bit(0x12);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x29);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x2A);
Send_Data_8bit(0x00);
Send_Data_8bit(0x00);
Send_Data_8bit(0x01);
Send_Data_8bit(0xDF);
Send_Command_8bit(0x00);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x2B);
Send_Data_8bit(0x00);
Send_Data_8bit(0x00);
Send_Data_8bit(0x01);
Send_Data_8bit(0x3F);
Send_Command_8bit(0x00);
LCD_CS = 1;
__delay_ms(50);

LCD_CS = 0;
Send_Command_8bit(0x51);
Send_Data_8bit(0xFF);
LCD_CS = 1;
```

```
     __delay_ms(50);

     LCD_CS = 0;
     Send_Command_8bit(0x53);
     Send_Data_8bit(0x2C);
     LCD_CS = 1;
     __delay_ms(50);

     LCD_CS = 0;
     Send_Command_8bit(0xB1);
     Send_Data_8bit(0xB0);
     Send_Data_8bit(0x11);
     LCD_CS = 1;
     __delay_ms(50);
}

void Set_Background()
{
    unsigned i, j;

     LCD_CS = 0;
     Send_Command_8bit(0x2C);
     for (i=0; i<480; i++)
     {
      for (j=0; j<320; j++)
      {
       Send_Data_8bit(0);
       Send_Data_8bit(0);
       Send_Data_8bit(0);
      }
     }
     Send_Command_8bit(0x00);
     LCD_CS = 1;

    setRotation(1);
}
```

ADD_LIB

```
#include <stdint.h>
#include <stdbool.h>
#include <stdarg.h>
#include <math.h>

unsigned _width = 320, _height = 480;
unsigned short rotation = 0;

void Send_Command_8bit(unsigned Com)
{
 LCD_RS = 0;
 LCD_RD = 1;
 LCD_WR = 1;
 PORTD = (unsigned char) Com;
 LCD_WR = 0;
 __delay_us(1);
 LCD_WR = 1;
}
```

```
void Send_Data_8bit(unsigned Dataa)
{
 LCD_RS = 1;
 PORTD = (unsigned char) Dataa ;
 LCD_WR = 0;
 LCD_RD = 1;
  __delay_us(1);
 LCD_WR = 1;
}

void set_address(unsigned x, unsigned y) {
 unsigned xt, xb, yt, yb;
 xt = (x >> 8);
 xb = x;
 yt = (y >> 8);
 yb = y;
     LCD_CS = 0;
     Send_Command_8bit(0x2A);
     Send_Data_8bit(xt);
     Send_Data_8bit(xb);
     Send_Data_8bit(xt);
     Send_Data_8bit(xb);
     Send_Command_8bit(0x00);
     LCD_CS = 1;
     __delay_us(1);

     LCD_CS = 0;
     Send_Command_8bit(0x2B);
     Send_Data_8bit(yt);
     Send_Data_8bit(yb);
     Send_Data_8bit(yt);
     Send_Data_8bit(yb);
     Send_Command_8bit(0x00);
     LCD_CS = 1;
     __delay_us(1);
}

void setAddrWindow(unsigned x1, unsigned y1, unsigned w, unsigned h) {

    unsigned x2 = (x1 + w - 1),
                  y2 = (y1 + h - 1);

    LCD_CS = 0;
     Send_Command_8bit(0x2A);
     Send_Data_8bit(x1 >> 8);
     Send_Data_8bit(x1);
     Send_Data_8bit(x2 >> 8);
     Send_Data_8bit(x2);
     Send_Command_8bit(0x00);
     LCD_CS = 1;
     __delay_us(1);

     LCD_CS = 0;
     Send_Command_8bit(0x2B);
     Send_Data_8bit(y1 >> 8);
     Send_Data_8bit(y1);
     Send_Data_8bit(y2 >> 8);
     Send_Data_8bit(y2);
```

```
     Send_Command_8bit(0x00);
     LCD_CS = 1;
     __delay_us(1);

     //LCD_CS = 0;
     //Send_Command_8bit(0x2C);
     //LCD_CS = 1;
     //delay_ms(1);
}

void write_pixel(unsigned x, unsigned y, unsigned r, unsigned g, unsigned
b)  {
   set_address(x, y);
   LCD_CS = 0;
   Send_Command_8bit(0x2C);
   Send_Data_8bit(r);
   Send_Data_8bit(g);
   Send_Data_8bit(b);
   LCD_CS = 1;
}

void setRotation(unsigned m) {
rotation = m % 4;
    switch (rotation) {
        case 0:
            m = (0x40 | 0x08);
            _width  = TFTWIDTH;
            _height = TFTHEIGHT;
            break;
        case 1:
            m = (0x20 | 0x08);
            _width  = TFTHEIGHT;
            _height = TFTWIDTH;
            break;
        case 2:
            m = (0x80 | 0x08);
            _width  = TFTWIDTH;
            _height = TFTHEIGHT;
            break;
        case 3:
            m = (0x40 | 0x80 | 0x20 | 0x08);
            _width  = TFTHEIGHT;
            _height = TFTWIDTH;
            break;
    }

    LCD_CS = 0;
    Send_Command_8bit(0x36);
    Send_Data_8bit(m);
    LCD_CS = 1;
}


void fillRect(unsigned x, unsigned y, unsigned w, unsigned h, unsigned r,
unsigned g, unsigned b) {
  uint32_t px;

  if( (x < _width) && (y < _height) && w && h) {
```

```
    if((x + w - 1) >= _width)
      w = _width - x;
    if((y + h - 1) >= _height)
      h = _height - y;
    setAddrWindow(x, y, w, h);
    LCD_CS = 0;
    px = (uint32_t)w * h;
    Send_Command_8bit(0x2C);
    while (px--) {
      Send_Data_8bit(r);
      Send_Data_8bit(g);
      Send_Data_8bit(b);
    }
    LCD_CS = 1;
  }

}
```

GFX_LIB

```
void fillScreen(unsigned r, unsigned g, unsigned b) {
    fillRect(0, 0, _width, _height, r, g, b);
}

void display_setCursor(unsigned x, unsigned y) {
    cursor_x = x;
    cursor_y = y;
}

void display_setTextSize(unsigned short s) {
    if (s > 0) textsize = s;
    else textsize = 1;
    //textsize = (s > 0) ? s : 1;
}

void display_setTextFont(unsigned short x, unsigned short y, unsigned
short f) {
  switch (x) {
  case 5: font_dim_X = 5;   break;
  case 8: font_dim_X = 8;   break;
  default: font_dim_X = 5;
            }

  switch (y) {
  case 7: font_dim_Y = 7;   break;
  case 16: font_dim_Y = 16;  break;
  default: font_dim_Y = 7;
            }

  switch (f) {
  case 1: N_font = 1;   break;
  case 2: N_font = 2;   break;
  default: N_font = 1;
            }
}
```

```
void display_setTextColor(unsigned short a, unsigned short b, unsigned
short c, unsigned short d, unsigned short e, unsigned short f) {
    txt_r = a;
    txt_g = b;
    txt_b = c;
    bg_r  = d;
    bg_g  = e;
    bg_b  = f;
}

void display_putn(unsigned short n)
    {
        unsigned char i = 0;

        while(n > 9999){
            n = n - 10000;
            i++;
        } if(i != 0){
            txt[0]= i + 48;
            i = 0;
            } else txt[0] = 0;
        while(n > 999){
            n = n - 1000;
            i++;
        } if(i != 0){
            txt[1]= i + 48;
            i = 0;
            } else txt[1] = 0;
        while(n > 99){
            n = n - 100;
            i++;
        } if(i != 0){
            txt[2]= i + 48;
            i = 0;
            } else txt[2] = 0;
        while(n > 9){
            n = n - 10;
            i++;
        } if(i != 0){
            txt[3]= i + 48;
            i = 0;
            } else txt[3] = 0;
        while(n > 0){
            n = n - 1;
            i++;
        } if(i != 0){
            txt[4]= i + 48;
            i = 0;
            } else txt[4] = 0;

      for(i=0; i<5; i++)
      {
          if(((txt[i] == 0) && (txt[i-1] != 0) && (i != 0)) || ((txt[i]
== 0) && (i == 4)))
          {
              txt[i]=48;
          }
          if(txt[i] != 0)
```

```
        {
            cursor_x + i * textsize;
            display_putc(txt[i]);
        }
    }
}
```

7. BIBLIOGRAFIA E SITOGRAFIA

Libreria grafica GFX. https://github.com/adafruit/Adafruit-GFX-Library

Micro controllori PIC: http://www.docenti.org/td/materiali/micro_pic.htm

Sensori di flusso: https://www.ne555.it/misuratore-flusso-liquidi-yf-s201/

Ossigenoterapia: https://www.nurse24.it/infermiere/farmaci/ossigenoterapia-cosa-come-quando.html

Pompa peristaltica: https://www.verderliquids.com/it/it/pompe-per-principio/come-funzionano-le-pompe-peristaltiche/come-funziona-una-pompa-peristaltica/

Citofluorimetria:https://labtestsonline.it/citofluorimetria#:~:text=Il%20citofluorimetro%20contiene%20uno%20o,dipendente%20dalle%20caratteristiche%20della%20stessa.

Strumentazione Elisa: https://www.microbiologiaitalia.it/test-microbiologici/elisa-principi-delle-tecniche-immunochimiche-ed-esempi-di-saggi/

Flussimetro ad ultrasuoni: https://www.pce-instruments.com/italiano/strumento-di-misura/misuratore/flussometro-a-ultrasuoni-kat_163009.htm

Capture (CCP) mode: https://www.electronicwings.com/pic/pic18f4550-timer-capture

Pickit3: https://www.ne555.it/programmare-con-pickit3/#:~:text=Il%20PICKIT%203%20%C3%A8%20un,gamma%20di%20PIC%20e%20dsPIC.