# Ecodesign dei dispositivi elettronici

## UNITÀ 8: Sistemi a microcontrollore parte 1

Autore: Andrej Sarjaš

8.1. Introduzione ai sistemi a microcontrollore ..... 2
8.2. Struttura del microcontrollore ..... 4
8.2.1. Termini comuni ..... 7
8.3. Struttura del microcontrollore ..... 8
8.3.1. Core del processore ..... 8
8.3.2. Memoria ..... 10
8.4. Unità periferiche ..... 14
8.4.1. Ingressi e uscite digitali ..... 14
8.4.2. Ingressi e uscite analogiche ..... 16
8.4.3. Contatori e timer ..... 21
8.4.5. Interfacce di comunicazione ..... 23
8.3. Linee guida per il basso consumo energetico dei microcontrollori e gli aspetti ecologici ..... 28

Riassunto del capitolo:

- Sistemi in tempo reale
- Componenti in tempo reale
- Progettazione di programmi in sistemi in tempo reale

Funded by the Erasmus+ Programme of the European Union

Ecoinnovation Skills for European Designers, Project number: 562573-EPP-1-2015-1-SI-EPPKA2-SSA. . Il presente Progetto è stato finanziato con il contributo dell'Unione Europea. Questa pubblicazione (comunicazione) è di responsabilità esclusiva del suo autore. La Commissione non è responsabile dell'uso che può essere fatto delle informazioni qui diffuse.

## 8.1. Introduzione ai sistemi a microcontrollore

Quando Intel ha presentato il primo microcontrollore 4004, è iniziata l'era dello sviluppo dei microcontrollori. La moderna struttura del microcontrollore TMS1802 di Texas Instruments è stata sviluppata per essere utilizzata nelle calcolatrici e è stata utilizzata fino al 1971 in molti sistemi in tempo reale, come orologi, strumenti di misura, registratori di cassa, ecc. Con lo sviluppo della nuova serie TMS100, iniziata nel 1974, il microcontrollore conteneva unità periferiche che sono i componenti base anche nei moderni microcontrollori (RAM, ROM, I/O). TMS100 era allora conosciuto con il soprannome di microcomputer. Il primo controller che ha riscosso un vero successo e che è stato ampiamente utilizzato è stato il 8048 di Intel con tastiera integrata. Quell'epoca è continuata con il modello Intel 8051 e Motorola 68HCxx.

L'attuale produzione di microcontrollori raggiunge miliardi di pezzi all'anno con numerosi produttori diversi. I sistemi elettronici di oggi sono difficili da immaginare senza un microcontrollore e il loro utilizzo è in costante aumento. Ecco alcuni gruppi di dispositivi che utilizzano sistemi a microcontrollore:

- Elettrodomestici (microonde, lavatrici, macchine da caffè, ecc...).
- Telecomunicazioni (telefoni, smartphone, modem, router, ecc...).
- Elettronica di consumo (televisori, lettori musicali e video, console di gioco, ecc....)
- Industria (gestione e controllo dei dispositivi e dei processi produttivi).
- Industria automobilistica (controllo motore, sistema di guida di sicurezza, ecc....).
- Tecnologia spaziale.

I sistemi a microcontrollore sono migliorati nel tempo e sono un'ottima alternativa ai sistemi e ai circuiti analogici. Con l'inclusione dei sistemi a microcontrollore nella progettazione dei dispositivi si può ridurre notevolmente la dimensione dei dispositivi, aumentare l'efficienza, l'affidabilità e rendere più facile l'aggiornamento. Dal punto di vista ecologico, i dispositivi con sistema a microcontrollore sono molto più economici, l'uso dei materiali è minore e il riciclaggio è più facile. I moderni microcontrollori contengono unità periferiche di base (RAM, FLASH, I/O, moduli di comunicazione) e anche unità separate che possono essere utilizzate per ridurre i consumi energetici quando non si utilizza il sistema, oppure quando questo è inattivo (modalità standby, modalità sveglia, ecc.). Con una buona conoscenza dell'architettura del microcontrollore e del concetto di programma nel sistema, si può sviluppare un sistema o dispositivo efficiente e più rispettoso dell'ambiente. L'ecodesign ha un ruolo

importante nei sistemi a microcontrollore in quanto esistono molte opzioni per garantire un funzionamento affidabile, bassi consumi e un basso utilizzo di materiali.

Allora, cos'è il microcontrollore? Qual è la differenza tra microcontrollore e processore? Perché utilizzare un microcontrollore e a quali compiti serve? Ad esempio, si può esaminare più da vicino un dispositivo per il riscaldamento dell'acqua a temperatura regolabile. Il principio di funzionamento di questo dispositivo è il seguente:

- Misurazione periodica della temperatura.
- Gestione del riscaldatore in funzione della corrente e della temperatura impostata (ON/OFF).
- Interfacce per la gestione dei dispositivi (pulsanti, tastiera).
- Visualizzazione della temperatura.

In primo luogo, il processo inizia con la progettazione di circuiti stampati e viene utilizzato il processore Zilog Z80. Sono state aggiunte anche due unità di ingresso-uscita - PIO, interfaccia seriale SIO, timer, SRAM, FLASH, EEPROM. Il circuito stampato finale è mostrato nell'immagine 1.

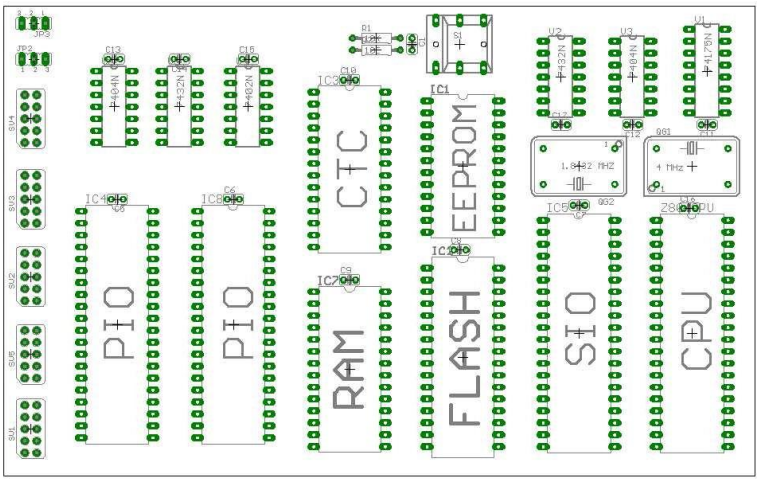



**IMMAGINE 1: CIRCUITO STAMPATO CON PROCESSORE Z80.**

Il problema può essere risolto con il microcontrollore ATmega16. La figura 2 mostra il disegno del circuito stampato per lo stesso sistema.

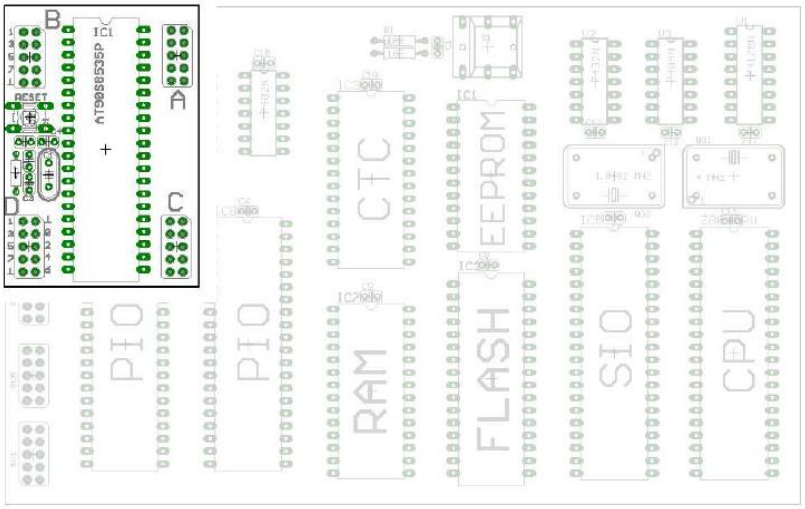



**IMMAGINE 2: CIRCUITO STAMPATO CON CONTROLLORE ATMEGA16.**

Da entrambi gli stampati è visibile che lo stampato dell'immagine 2 è più piccolo di quello dell'immagine 1. In breve, il microcontrollore è un microprocessore con unità periferiche in un unico chip. Ad esempio, si prendano in considerazione Z80 e ATmega16, perché il microcontrollore ATmega16 è stato sviluppato sulla base dello Z80. Ciò significa che include lo Z80 e dispone di unità periferiche integrate, come l'interfaccia seriale e I/O, ADC, SRAM, ROM, FLASH e EEPROM. In questo caso, è possibile anche valutare l'utilità e i vantaggi principali del microcontrollore.

Nello sviluppo dei dispositivi, sussistono molte opzioni, ad esempio nella scelta del produttore e della famiglia di microcontrollori. La famiglia dei microcontrollori è generalmente associata all'efficienza dell'unità logica aritmetica (8, 16, 32, 64, 128 bit). Per l'applicazione data è necessario determinare le dimensioni dei chip. I microcontrollori della stessa famiglia sono differenziati in base alla dimensione del chip o da pin liberi. Questo porta a 48, 100, 144 o 176 alloggiamenti a pin nella stessa famiglia. Se non esiste la necessità di molte interfacce esterne, è ragionevole scegliere chip con meno pin, perché è possibile risparmiare sul prezzo del dispositivo e sulle dimensioni del circuito stampato.

## 8.2. Struttura del microcontrollore

L'attuale sviluppo dei sistemi a microcontrollore è molto rapido. Attualmente, i più utilizzati sono sistemi a 16-32 bit con velocità di clock da 10 MHz a 300 MHz. La struttura interna del controllore è la stessa. L'immagine 3 presenta la struttura di base di un microcontrollore.

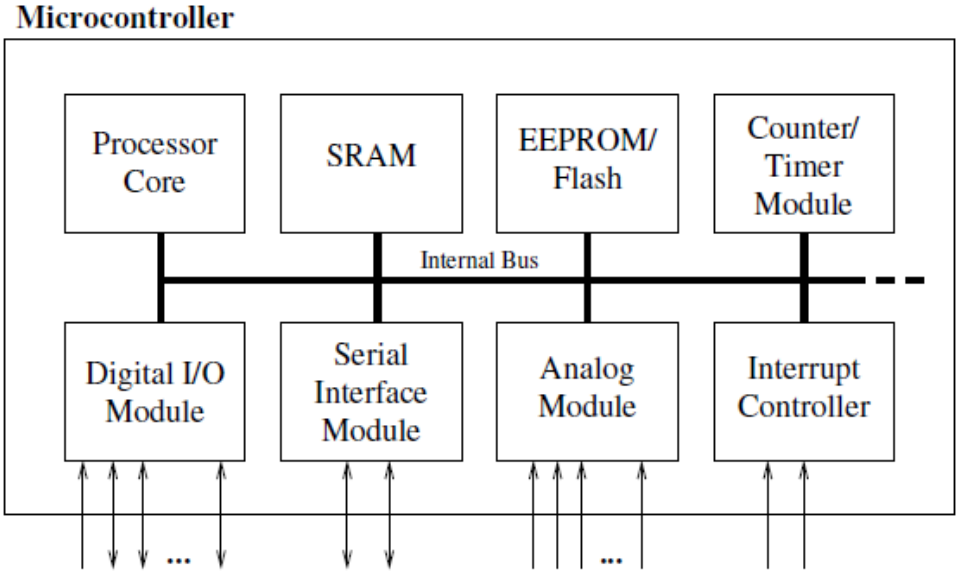



IMMAGINE 3: STRUTTURA DEL MICROCONTROLLORE

Tutte le unità periferiche del microcontrollore, come mostrato nell'immagine 3, sono interconnesse con l'" internal bus " centrale. I moduli tipici che sono comuni nella maggior parte dei microcontrollori sono:

- **CPU del processore:** Il processore o CPU (unità centrale di elaborazione) è un'unità logica aritmetica in grado di eseguire operazioni matematiche (addizione e sottrazione) e di condurre registri interni (registro della memoria, registro dei programmi, batteria, ecc.).
- **Memoria:** La memoria viene utilizzata per il salvataggio dei dati finali o intermedi e le variabili. La memoria del microcontrollore è suddivisa in programmi e dati. Tra i microcontrollori più potenti è noto il DMA (Direct Memory Access) che permette alle unità periferiche di comunicare direttamente con la memoria e non interrompere le operazioni del processore.
- **Interrupt controller:** Le interruzioni sono la chiave per un efficiente funzionamento in tempo reale di un programma. Le interruzioni interrompono le funzioni di base di un programma in caso di eventi nelle unità periferiche. Le interruzioni sono utili anche quando si vuole ridurre il consumo energetico in modalità sleep.
- **Contatori e timer:** La maggior parte dei controllori ha almeno due contatori destinati al conteggio del tempo, al conteggio degli eventi esterni

e degli intervalli. Molti timer sono dotati di modalità di codifica del conteggio e di modulazione PWM (*pulse width modulation* – modulazione di larghezza di impulso). La modulazione PWM è spesso utilizzata come DAC (*digital to analog conversion* – conversione da digitale ad anlogico). La risoluzione del timer/contatore dipende dall'orologio di sistema determinato dall'oscillatore esterno o interno e dalla lunghezza del registro di conteggio. La lunghezza del registro è determinata dal suo numero numero di bit. Pertanto, si possono avere contatori a 8, 16 o 32 bit.

- **Ingressi/uscite digitali:** Sono spesso etichettati come GPIO (general purpose input e output). Il numero di ingressi e uscite è limitato dal tipo di controllore e dalla sua versione.

- **Ingressi/uscite analogiche:** Tra i controllori più piccoli, la maggior parte ha convertitori analogici ADC (*analog to digital conversion* – conversione da analogico a digitale). Le serie più potenti di microcontrollori hanno più ADC, fino a quattro unità separate. Queste unità sono collegate ai pin esterni attraverso il multiplexer. In pratica, questo significa che è possibile utilizzare diversi ingressi analogici multiplexati. Il numero di ingressi analogici fisici multiplexati sui pin del controllore è quindi superiore al numero di unità ACD nel controllore. Le unità ACD sono differenziate in base alla risoluzione di conversione, e i convertitori da 6 a 12 bit. La risoluzione ACD è determinata dal registro di controllo dell'unità ADC impostato dal codice di programma. Nella più recente serie di controllori, la tendenza è anche quella di integrare unità DAC che sono fondamentali per l'elaborazione dei segnali di uscita.

- **Interfacce di comunicazione:** Per le periferiche in circuto stampato che non fanno parte del chip esistono tre tipologie standard di comunicazione maggiormente utilizzate. Si tratta di connessioni seriali USART, I2C e SPI. Tutte queste connessioni hanno in comune il fatto che la distanza tra i dispositivi varia da pochi centimetri alla distanza massima di un metro.

- **Watchdog timer:** Watchdog timer è un timer speciale che si attiva indipendentemente dal funzionamento del controllerore. Questo timer assicura che in caso di errore di funzionamento il controllore venga resettato.

- **Unità di debug:** Nella progettazione di un programma per il controllere è pratico utilizzare l'interfaccia di debug. Questa interfaccia può accedere al registro interno del controllore mentre il programma è in esecuzione. In questo modo, è più facile cercare errori nel codice del programma e risolvere eventuali problemi nell'implementazione dello stesso. Spesso si chiama JTAG (Joint Test Action Group).

Per riassumere: il microcontrollore è un processore limitato che ha integrato unità periferiche precedentemente elencate. Il più grande vantaggio del microcontrollore è il prezzo. È possibile risparmiare denaro, ridurre le dimensioni del dispositivo e aggiungere una certa potenza di calcolo al sistema. Lo svantaggio del microcontrollore è che non può raggiungere la velocità dei circuiti analogici. Nei sistemi con tempi di reazione molto brevi, è necessario sostituire una certa parte del sistema con un circuito analogico. Nella maggior parte dei casi, il tempo di reazione non è la chiave per il funzionamento del dispositivo. I nuovi microcontrollori raggiungono già il tempo di reazione nell'intervallo di alcuni 10 microsecondi.

### 8.2.1. Termini comuni

Spesso si possono incontrare termini diversi che sono fuorvianti o che sono usati in modo scorretto.

- **Microprocessore:** Si tratta di una normale CPU che si trova nei personal computer. La comunicazione tra le periferiche avviene attraverso il bus principale. Tutte le unità esterne (memoria, USB, timer) sono collegate al bus principale. Il microprocessore non può funzionare da solo perché ha bisogno di periferiche, come la memoria, unità di ingresso/uscita, ecc. Il microprocessore non è un microcontrollore.
- **Microcontrollore**: comprende tutte le unità periferiche e può funzionare da solo. È progettato per la gestione dei sistemi. A seconda del microprocessore, il microcontrollore è dotato di unità periferiche integrate nel chip stesso.
- **Controllore di segnale misto**: Il controllore di segnale misto può elaborare sia segnali analogici che digitali.
- **Sistema integrato**: Un'area importante del microcontrollore sono i sistemi integrati. Il termine sistemi integrati significa che il controllere è integrato nel sistema o nel dispositivo. Esempi di questo tipo sono telefoni, robot, auto, ecc. Si tratta di sistemi controllati da questi controllori.
- **Sistema in tempo reale:** Significa che il sistema è controllato e gestito nel momento definito. La reazione del sistema viene eseguita in intervalli di tempo definiti - in tempo reale. L'intervallo di tempo è spesso valutato come tempo di sistema determinato dal microcontrollore o dal processore.
- **Processore di segnale digitale DSP: Si** tratta di processori che si distinguono da un'unità logica aritmetica di addizione e sottrazione in grado di eseguire la moltiplicazione e la divisione in un ciclo. Tali processori sono destinati all'elaborazione digitale del segnale (trasformazione di

Fourier, calcolo della correlazione, autocorrelazioni e convoluzione). I principali produttori di controllori DSP sono:
Texas Instruments, STMicroelectronics, Freescale, Microchip e Nxp Semiconductors, ecc…

## 8.3. Struttura del microcontrollore

In questo capitolo daremo uno sguardo alla struttura e al funzionamento dei principali elementi del microcontrollore.

### 8.3.1. Core del processore

La CPU del processore è la parte principale di ogni microcontrollore. L'immagine 4 presenta la struttura di base del processore.

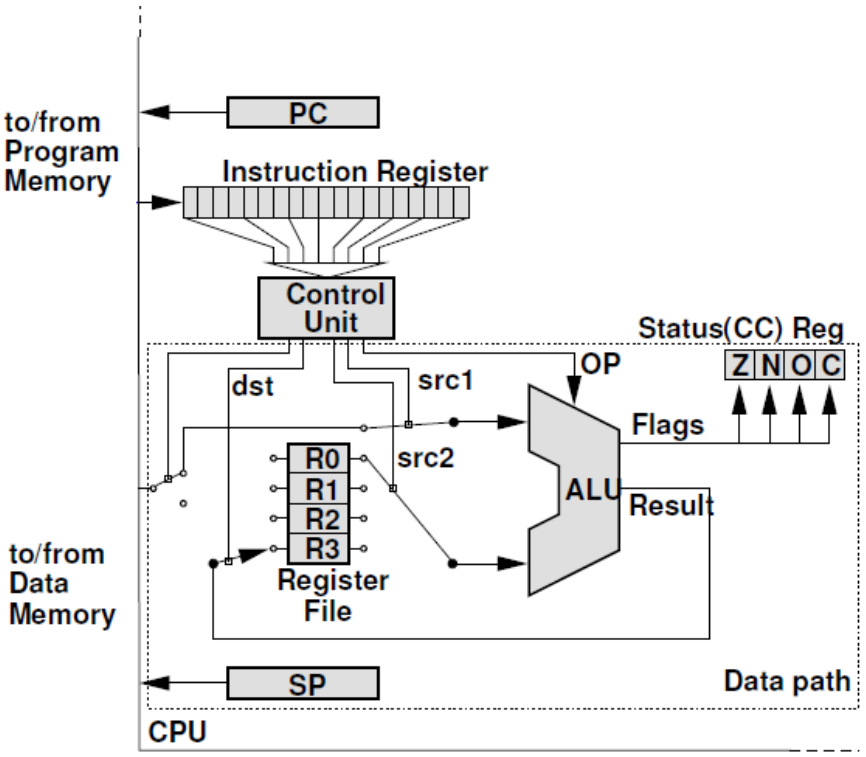



IMMAGINE 4: STRUTTURA DEL PROCESSORE

Il core della CPU è costituito da un percorso dati che esegue le istruzioni dell'unità di controllo che gestisce il percorso dati. Il core della CPU è costituito da unità logica aritmetica ALU che è in grado di eseguire solo operazioni di calcolo di base, come

addizioni, sottrazioni e complementi bitwise. Fondamentalmente, ALU prende due valori e li restituisce all'output. L'ALU salva anche nel registro di stato. Le etichette nel registro di stato indicano Z se il risultato è pari a zero, N se è negativo, O se l'operazione ha causato un overflow, C se l'operazione fornisce dati ("curry").

Il compito dell'unità di controllo è quello di eseguire le istruzioni e determinare quale sarà eseguita in un dato momento. Questa unità salva anche l'indice delle istruzioni nel contatore di programma e il contenuto di queste istruzioni nel registro delle istruzioni. L'istruzione definisce quale valore del registro sarà inviato all'ALU e dove sarà salvato. A seconda della struttura dell'unità di controllo, esistono due architetture:

- RISC: L'architettura computer con set di istruzioni ridotto è più semplice perché l'esecuzione richiede solo pochi cicli di clock. I vantaggi dei sistemi RISC sono nell'uso di una certa lunghezza di micro-codice per indirizzamento e istruzioni. Di conseguenza, le istruzioni vengono eseguite rapidamente ma sono piuttosto piccole.
- CISC: Il computer con set di istruzioni complesso è una struttura gestita da complesse istruzioni microcodificate. Tali istruzioni richiedono diversi cicli di clock per l'esecuzione. L'istruzione ha una lunghezza variabile e consente di eseguire molte istruzioni efficienti rispetto alla struttura del RISC.

La scelta della struttura dipende dall'applicazione per la quale si ha bisogno del controllore. Se si ha bisogno di istruzioni più complesse, allora è necessario usare la struttura CISC. La velocità di esecuzione delle istruzioni è collegata al clock principale della CPU. Il core del microcontrollore è spesso basato sulla struttura RISC, mentre i processori informatici si basano principalmente sulla struttura CISC.

Nell'immagine 4 si può vedere l'istruzione e la memoria dati come due soggetti separati. Non è sempre così, ma entrambe le memorie possono utilizzare la stessa memoria. A seconda che la memoria sia separata o comune, si differenziano due tipi di architettura di processore:

- **L'architettura Von Neumann:** In questa architettura, l'istruzione e la memoria dati è comune. Anche il bus utilizzato per accedere alla memoria è comune. Purtroppo, in questa architettura, può portare ad un conflitto tra le istruzioni e i dati che può causare ritardi indesiderati del sistema. Questo svantaggio è generalmente noto come collo di bottiglia Von Neumann.
- **Architettura di Harvard:** In questa architettura, la memoria, così come il bus, sono separati. Il conflitto tra dati e istruzioni non è possibile, migliorando di conseguenza l'efficienza del processore. Lo svantaggio del sistema è che l'architettura ha bisogno di più componenti, come la doppia memoria, il doppio bus e un'unità che consente l'accesso simultaneo alla memoria dati e istruzioni.

La velocità di esecuzione dei task nella CPU dipende da diversi fattori. Esso è influenzato principalmente dall'architettura della CPU, ovvero se il sistema è RISC o CISC. La velocità di esecuzione del task dipende anche dalla lunghezza dell'istruzione (8, 16, 16, 32, 64 bit). Un'istruzione a 32 bit che viene eseguita in un solo ciclo su CPU a 32 bit, è più veloce rispetto alla sua esecuzione su CPU a 8 bit. La CPU a 8 bit richiede quattro cicli per l'esecuzione delle istruzioni. Alla fine, la velocità di esecuzione del compito dipende dal clock principale (cristallo esterno o interno), il che significa che le istruzioni vengono eseguite più velocemente con un clock a 160 MHz che con un clock a 10 MHz.

### 8.3.2. Memoria

Nel pagagrafo precedente, è stata presentata la struttura dei processori che hanno bisogno di memoria per il loro funzionamento. A seconda del compito e della struttura della memoria esistono tre tipi di memoria:

- **Registro:** La memoria di registro è una memoria relativamente piccola integrata nella CPU. La CPU ha bisogno di memoria per scrivere le condizioni della CPU e le impostazioni delle unità periferiche, come ADC, DAY, I2C, I2C, ecc. Questa memoria è chiamata anche memoria a breve termine perché tutti i valori presenti nella memoria vengono persi quando la sorgente di ricarica viene scollegata.
- **Memoria dati**: Nella memoria dati si possono salvare dati a lungo termine, e di solito viene aggiunto alla CPU come unità periferica. La memoria dati è significativamente più grande del registro.
- **Memoria delle istruzioni:** È relativamente grande, analogamente alla memoria di dati e di conseguenza viene implementa esternamente alla CPU come periferica. Nell'architettura Von Neuman, l'istruzione e la memoria dati sono una periferica con un bus comune. Nei sistemi a microcontrollore, la memoria è implementata nella memoria a singolo chip, ma è divisa in settori.

I tipi di memoria sopra menzionati sono i più comuni tipi di utilizzo della memoria della CPU. Ci sono altri tipi di memoria e registri, come i registri delle pipeline, la memoria cache e diversi buffer. A seconda della struttura del microcontrollore, la dimensione della memoria è integrata nel chip ed è fissa. La memoria non può essere aggiunta o aumentata. Per questo motivo, i microcontrollori sono destinati esclusivamente all'esecuzione di compiti più semplici.

Fino ad ora la memoria è stata trattata come un'unità che svolge determinati compiti della CPU. Dal punto di vista della programmazione, la memoria può essere caratterizzata come memoria non volatile e volatile. La memoria volatile può essere classificata come dinamica o statica. Per la memoria non volatile, vengono utilizzate

abbreviazioni a seconda della struttura e del tipo di funzionamento: ROM, EPROM, EEPROM, FLASH. La classificazione della memoria è sintetizzata nell'immagine 5.

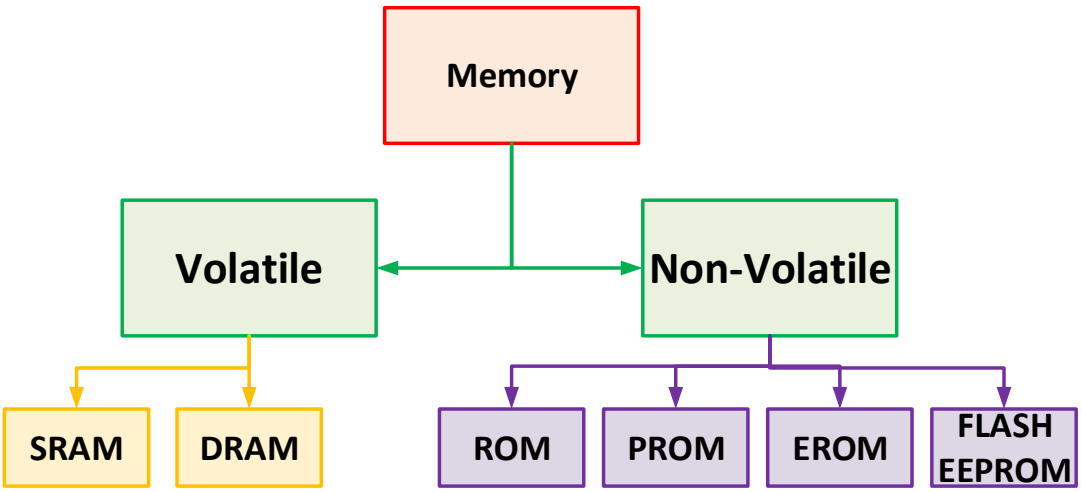



IMMAGINE 5: TIPOLOGIE DI MEMORIA

La memoria volatile è il tipo di memoria che conserva i dati fino a quando non viene spenta o il sistema viene resettato. Allora, perché non viene utilizzata solo la memoria non volatile? La risposta è semplice. La memoria non volatile è più lenta della memoria volatile e richiede più tempo operativo per il salvataggio, la cancellazione e la lettura. Facendo un confronto approssimativo: la lettura sulla memoria volatile è misurata in nanosecondi e la lettura sulla memoria non volatile è misurata in millisecondi.

- **RAM-SRAM statica:** La parola RAM è acronimo di *Random Acess Memory* (memoria ad accesso casuale), il che significa che è possibile accedere alle posizioni di memoria in modo casuale. Ogni posizione è accessibile con l'indirizzo di memoria. Contrariamente alla RAM, i registri di turno dove i dati possono essere scritti/letti solo in sequenza. Tale memoria è denominata Shift registers di tipo FIFO (First in first first out), FILO, LIFO (Last in first out), LILO, ecc. La memoria SRAM per il salvataggio utilizza celle D-flip-flop. Ogni D-flip-flop è composto da almeno sei transistor. Ogni cella D-flip-flop può salvare un bit (0 o 1), e ogni cella ha un indirizzo. L'immagine 6 mostra una memoria matrice a 16 bit con D-flip-flop. $A0_{,1}$ in $_{A2,3}$ mean row/column address.

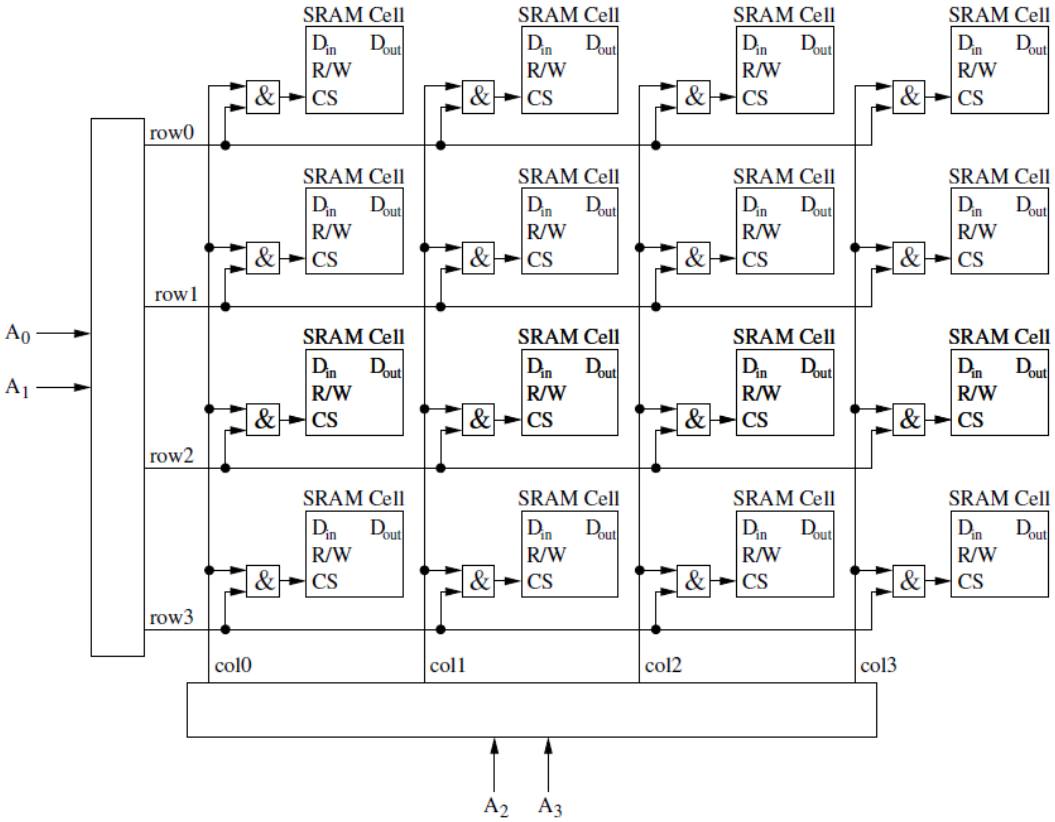



IMMAGINE 6: MEMORIA RAM A 16 BIT

- **RAM dinamica - DRAM:** rispetto alla SRAM, la RAM dinamica raggiunge capacità significativamente superiori. La SRAM ha bisogno di almeno sei transistor per risparmiare 1 bit. Per la maggiore capacità di SRAM, sono necessarie più celle che aumentano significativamente la dimensione fisica del chip e il prezzo. Se la memoria può essere ridotta, si può utilizzare un modo semplice per salvare il valore in bit. La DRAM salva i dati nella memoria di carica elettrica (condensatore). In questo modo si riduce il numero di elementi per  bit e di conseguenza si aumenta la capacità di memoria con chip di dimensioni notevolmente inferiori. Se si vuole salvare il valore logico nella memoria DRAM, bisogna caricare il condensatore in una certa posizione. Utilizza indirizzi per l'indirizzamento di ogni cella, come SRAM e DRAM. A causa della struttura più semplice e del prezzo più basso, le memorie DRAM hanno alcuni svantaggi rispetto alle memorie SRAM. La DRAM è, per esempio, più lenta. Per la manutenzione è necessario ricaricare la memoria perché il condensatore si esaurisce nel tempo.

Oltre SRAM e DRAM, vengono utilizzate anche memorie non volatili. Queste si usano quando si desidera salvare i dati per un periodo di tempo più lungo anche quando il sistema non è collegato alla rete elettrica.

- **ROM:** ROM sta per memoria di sola lettura. Sola lettura significa che non è possibile salvarvi alcun dato. La ROM di solito contiene i dati salvati dal produttore e sono fondamentali per il funzionamento del dispositivo. Nella ROM viene salvato il BIOS per i personal computer o i valori di registro per le impostazioni del microcontrollore.

- **PROM:** la ROM è utilizzata solo nella produzione di massa perché la sua produzione per piccole quantità o per ricerche pilota è troppo costosa. PROM (memoria di sola lettura programmabile) è un'alternativa alla ROM. Può essere programmato una sola volta perché contiene un fusibile che impedisce la scrittura ripetuta. PROM è anche conosciuta come OTP (*one-time programmable memory* – memoria programmabile una volta). Non è adatto alle fasi di sviluppo quando il contenuto della memoria e il programma cambiano spesso.

- **EPROM:** EPROM è una memoria di sola lettura cancellabile e programmabile. La programmazione dell'EPROM richiede una procedura complessa. Quindi di solito non può essere cancellato dal microcontrollore. Il valore dell'EPROM viene salvato in FET (field effect transistor) che viene gestito da pin-gate. L'alta tensione in pin-gate chiude il transistor. Quando questi gate sono chiusi lo rimangono finché non sono più sotto tensione. In questo modo è possibile salvare i valori digitali 1 o 0. A causa di problemi nel FET, i gate possono aprirsi nel tempo. I produttori di solito indicano per quanto tempo i dati possono essere salvati in EPROM. Questo periodo di solito è di dieci anni - ogni EPROM ha una finestra che consente la cancellazione. Questo viene eseguito attraverso la finestra con luce ultravioletta. La luce UV apre il FET, il che comporta la cancellazione della memoria. La cancellazione dell'EPROM richiede circa 30-40 minuti.

- **EEPROM:** EEPROM è una ROM cancellabile e programmabile elettricamente. Essenzialmente, la EEPROM funziona allo stesso modo della EPROM; l'unica differenza è che per la cancellazione della memoria non si utilizzano particolari tensioni esterne o luce UV. L'alta tensione per il risveglio del FET è integrata nel chip ed è nota come pompa di carica. In generale, EEPROM ha, come EPROM, un ciclo di vita che è spesso limitato a 100.000 cicli di cancellazione.

- **FLASH:** FLASH è una versione limitata della memoria EEPROM e funziona allo stesso modo della EEPROM. La differenza tra FLASH e EEPROM è che non si può cancellare ogni cella separatamente, ma solo la memoria completa di un certo settore. La ragione per l'implementazione della

memoria FLASH è il prezzo elevato della memoria EEPROM. FLASH ha anche un numero limitato di ingressi, come l'EEPROM.

- **NVRAM:** NVRAM (RAM non volatile) è una combinazione di memoria volatile e non volatile. Questa memoria ha lo stesso principio di funzionamento della SRAM, ma ha aggiunto la batteria di alimentazione. Esiste anche una versione di memoria, in cui la memoria EEPROM e SRAM sono unite nello stesso chip

Quando si opera con dati dalla memoria, è importante riconoscere il tipo di scrittura. Per scrivere i dati, esistono due approcci:

- **Big Endian**: Si tratta di una tipologia di scrittura/lettura dati in cui vengono salvati byte superiori all'inizio della posizione di memoria. Ad esempio, se si salvano i dati '0x7799' sull'indirizzo di memoria '0x00' e '0x01'. In Big Endian il valore '0x77' viene salvato all'indirizzo '0x00' e il valore '0x99' all'indirizzo '0x01'.
- •**Little Endian**: È un tipo di scrittura in cui i byte più alti sono salvati in posizioni più alte. Riprendendo l'esempio precedente: dal valore '0x7799, la parte '0x77' viene salvata nella posizione '0x01' e '0x99' viene salvata in '0x00'.

## 8.4. Unità periferiche

Nei paragrafi precedenti, è stata presentata la differenza tra microcontrollore e processore. In questo paragrafo, presenteremo le unità periferiche, che più comunemente integrate in un chip di un moderno microcontrollore.

### 8.4.1. Ingressi e uscite digitali

Ingressi e uscite digitali - I/O sono utilizzati per il controllo e la gestione di dispositivi esterni e sono le unità principali all'interno del microcontrollore. Di conseguenza, ogni microcontrollore ha almeno 1 o 2 pin I/O digitali che possono essere collegati a dispositivi esterni. Generalmente, i microcontrollori hanno più di 32 I/O che possono essere utilizzati per vari scopi. Quando si collegano gli I/O è necessario prestare attenzione ai livelli di tensione consentiti da un determinato microcontrollore. Le tensioni più comuni sono 5 e 3,3 volt.

Gli ingressi e le uscite digitali sono spesso raggruppati in porte. Ogni porta può contenere 8, 16 o 32 pin I/O. Il raggruppamento di un certo numero di pin alle singole porte dipende dal produttore e dalla lunghezza dei registri di controllo del

microcontrollore. Nella maggior parte dei casi, tutti i pin I/O sono a due vie, il che significa che possono essere utilizzati come ingressi o uscite. La maggior parte dei pin I/O hanno anche funzionalità aggiuntive, come contatori, interruzioni esterne, interfaccia di comunicazione e conversione ADC analogico-digitale o DAC digitale-analogico. Tutte le funzionalità della porta I/O possono essere impostate nel programma attraverso i registri di controllo di ogni porta.

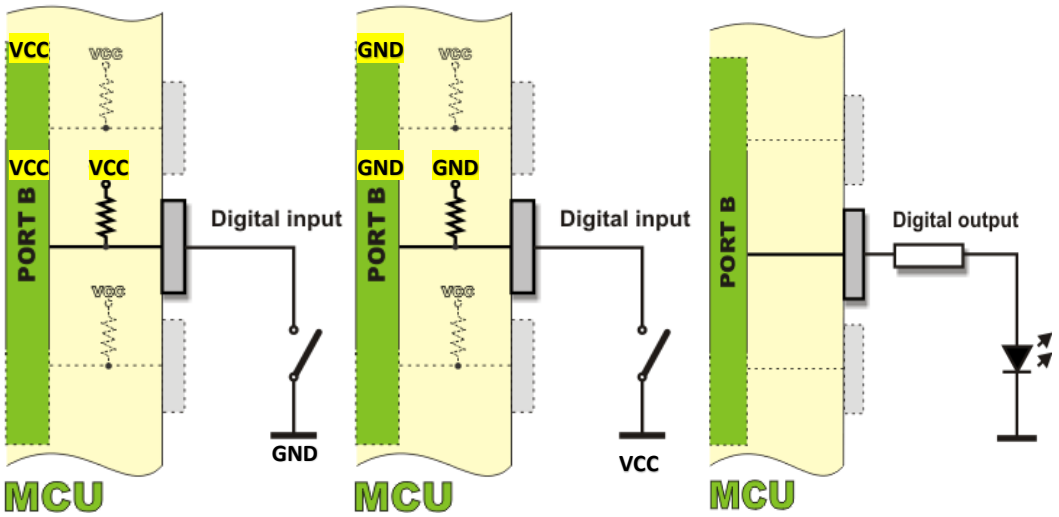
Gli ingressi/uscite digitali vengono denominati in questo modo perché il potenziale di tensione all'ingresso pin viene convertito in un valore logico "1" o "0". I livelli di tensione dei valori logici sono presentati nell'immagine 7. In generale, si può affermare che lo zero logico è il potenziale di tensione sotto 0-1V. L'"1" logico è il potenziale di tensione superiore a 2,4V-Vcc. Vcc è la tensione di alimentazione del microcontrollore. Il potenziale di tensione varia a seconda del produttore e del tipo di pin I/O e anche se viene utilizzato come ingresso o uscita [2].

| GPIO input/output pin electrical characteristics | |
|---|---|
| Output low voltage $V_{OL}$ | < 0.40 V [1]<br>< 0.66 V [2]<br>< 0.40 V [3]<br>< 0.40 V [4] |
| Output high voltage $V_{OH}$ | > 2.40 V [5]<br>> 2.64 V [6]<br>> 2.90 V [7] |
| Input low voltage $V_{IL}$ | < 0.80 V [8]<br>< 0.54 V [9]<br>< 1.15 V [10] |
| Input high voltage $V_{IH}$ | > 2.00 V [11]<br>> 2.31 V [12]<br>> 2.15 V [13] |
| Hystereses | > 0.25 V [14]<br>0.66 – 2.08 V [15] |
| Schmitt trigger input low threshold $V_{T-}$ | 1.09 - 1.16 V [16]<br>0.9 [17] |
| Schmitt trigger input high threshold $V_{T+}$ | 2.24 - 2.74 V [18]<br>0.90 V [19] |
| Pull-up/down resistance | 40 – 65 KΩ [20]<br>100 KΩ [21] |
| Pull-up/down current | < 50 uA [22]<br>< 28 uA [23] |
| Pin capacitance | 5 pF [24] |
| Bus hold resistance | 5-11 KΩ [25] |

**IMMAGINE 7: POTENZIALI DI TENSIONE PIN I/O DI DIVERSI PRODUTTORI**

I microcontrollori utilizzano spesso una logica positiva, il che significa che il "1" logico è potenziale superiore a 2.4V e lo "0" logico è inferiore a 1V. In logica negativa il logico "1" è inferiore a 1V e il logico "0" è superiore a 2.4V. I sistemi misti sono spesso utilizzati quando si desidera collegare microcontrollori con tensioni di alimentazione diverse. Si prenda, per esempio, un sistema alimentato da 5V e 3.3V. Spesso i produttori di microcontrollori producono I/O con tensione di alimentazione inferiore che sono tolleranti a 5V. Le caratteristiche dei perni e le caratteristiche elettriche dei perni sono descritte nelle schede tecniche di ogni gruppo e produttore separatamente.

Spesso s'incontrano i termini "pull-up", "pull-down" o " floating ". Il termine *pull* significa che è possibile determinare il potenziale di tensione predefinito per i pin I/O quando non è presente potenziale esterno sul pin. In modalità *pull-up*, il potenziale di default è la tensione di alimentazione, mentre in modalità *pull-down* il potenziale di default è GND. Il termine *floating* significa che il perno è lasciato galleggiante (non esiste un potenziale predefinito). In questo caso, il pin è influenzabile da fattori esterni, che possono causare interruzioni indesiderate nel sistema se il pin viene utilizzato come ingresso o uscita discreta. Il modo *floating* è spesso utilizzato nella conversione ADC o DAC. L'immagine 8 presenta le modalità pull up, pull down e floating.




*a)* *b)* *c)*

**IMMAGINE 8: MODALITÀ PIN I/O; A) PULL-UP, B) PULL-DOWN, C) FLOATING.**

### 8.4.2. Ingressi e uscite analogiche

Rispetto agli ingressi e alle uscite digitali, è possibile leggere o elaborare diversi livelli di tensione con ingressi/uscite analogiche. La lettura dei valori analogici è correlata all'unità periferica ADC nel controllore. I dati di base del convertitore ADC sono la risoluzione, la velocità di conversione e il tipo di conversione. La risoluzione ADC è data dal numero di bit. Dal numero di bit, è possibile determinare in quanti livelli di tensione sono suddivisi nel campo di tensione di ADC. L'intervallo di tensione di default di ADC è solitamente la tensione di alimentazione di ADC. La maggior parte dei microcontrollori ha un intervallo di tensione regolabile che può provenire da un circuito di riferimento esterno. I microcontrollori hanno anche una risoluzione ADC regolabile. Questo significa che è possibile scegliere tra la risoluzione del convertitore a 6, 8, 10 o 12 bit. SI considerino l'ADC a 12 bit e la tensione di alimentazione 3.3V. La tensione più bassa che può essere misurata da ADC è data dalla seguente formula:

$$V_{RES} = \frac{V_{REF}}{2^N - 1},$$

dove $V_{REF}$ è la gamma di tensione di ADC, N è il numero di bit e $V_{RES}$ è la risoluzione di tensione del convertitore. Per l'esempio dato (12 bit, 3,3V), la risoluzione della tensione del convertitore è $V_{RES}$= 0,805mV. Molti convertitori consentono di impostare la tensione di riferimento più alta e più bassa.

Ciò significa che la piena risoluzione del convertitore può essere utilizzata tra due potenziali. Per esempio, date un'occhiata alle misure dei potenziali di tensione tra 1.5V e 2.2V. con l'impostazione $V_{REF}$-=1.5V e $V_{REF}$+=2.2V si può distribuire la risoluzione a 12 bit solo tra 0.7V ($V_{REF}$+ - $V_{REF}$-). La risoluzione del convertitore tra i potenziali 1.5-2.2.2.V è ora $V_{RES} = \frac{V_{REF+} - V_{REF-}}{2^N - 1} = \frac{2.2 - 1.5}{2^{12} - 1} = 0.171mV$.

Il prossimo dato chiave è la velocità di conversione che di solito è data dal numero di campioni al secondo. La velocità di conversione nei microcontrollori di solito varia da diverse 100ksps a diversi 3Msps, a seconda del produttore e della famiglia di controllori. 3Msps significa che ADC crea 3 milioni di campioni in un secondo. La velocità dei campioni è un dato chiave nell'elaborazione digitale del segnale.

La cattura del segnale ADC è suddivisa in tre fasi. All'inizio viene utilizzato un supporto di segnale (sample & hold). Il supporto del segnale trattiene il segnale per tutto il tempo necessario alla conversione. Nell'immagine 9, il supporto del segnale è presentato come interruttore e condensatore in cui si può salvare la tensione del segnale campionato.

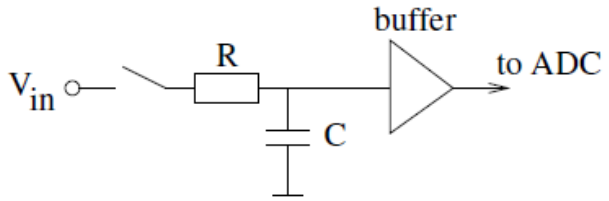



**IMMAGINE 9: SCHEMA SAMPLE & HOLD**

La seconda fase di conversione è la quantificazione. Questo è un processo in cui i livelli di tensione vengono divisi in campi di misura. Un quantum ADC presenta la risoluzione ADC che è stata presentata con la formula di cui sopra. Il processo di quantificazione è l'arrotondamento del valore analogico reale; questo significa che si inserisce l'errore di misura. Questo errore viene chiamato errore di quantificazione del convertitore ADC. L'ultimo passo nella conversione è la codifica dove ogni livello di quantificazione ha assegnato un valore binario. Nell'immagine 10 viene presentato il processo di quantificazione per DC a 3 bit con tempo di campionamento τs.

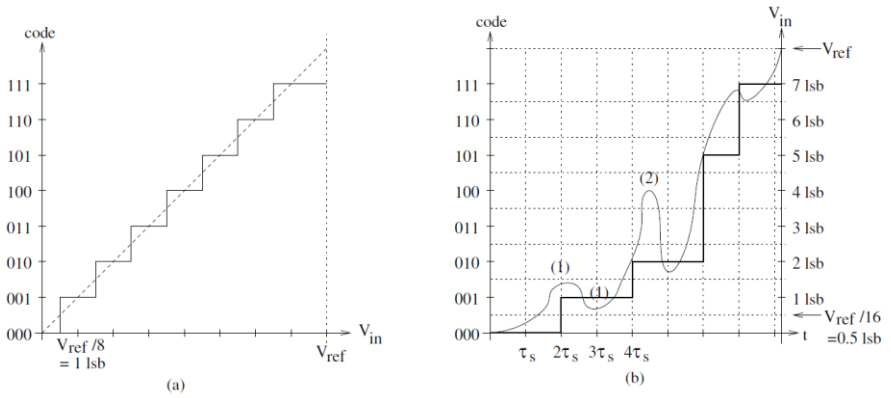



**IMMAGINE 10: PROCESSO DI QUANTIFICAZIONE: A) LIVELLI DI TENSIONE DI ADC *0-VREF*, B) QUANTIFICAZIONE DEL SEGNALE**

In un microcontrollore, l'unità ADC è collegata ai pin del controllore I/O. Questo ci permette di leggere più segnali analogici con uno o due ADC. Gli ingressi in ADC sono multiplexati. I pin multiplexati sono chiamati anche canali ADC. I microcontrollori avanzati hanno 3-4 ADC separati, mentre i più semplici hanno solo 1 ADC. L'immagine 11 presenta il multiplexing dei pin di ingresso in ADC.

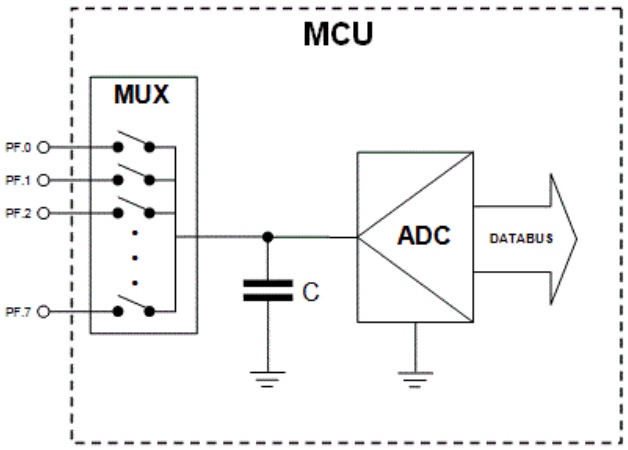



**IMMAGINE 11: PIN DI INGRESSO MULTIPLEXATI IN ADC**

La funzione del convertitore digitale-analogico DAC è quella di convertire il valore binario in segnale analogico. Il DAC è più spesso utilizzato per l'elaborazione di qualsiasi segnale di uscita dal controller. Il DAC ha caratteristiche simili a quelle di ADC. La risoluzione del DAC è definita dalla risoluzione (numero di bit) e dalla velocità di campionamento al secondo. Il livello di tensione del segnale di uscita è determinato dal riferimento di tensione impostato di default sulla tensione di alimentazione del regolatore. Le unità DAC si trovano solo nei controllori più avanzati e il numero di unità è generalmente inferiore al numero di ADC. Se il regolatore non è dotato di unità DAC, il segnale analogico può essere elaborato con la modulazione di larghezza d'impulso (Pulse Width Modulation – PWM). Il segnale PWM viene generato con frequenza fissa e duty cycle regolabile. Con il duty cycle, si genera il potenziale di tensione desiderato sul

pin di uscita. Il valore medio di un periodo di segnale PWM presenta il valore analogico di uscita.

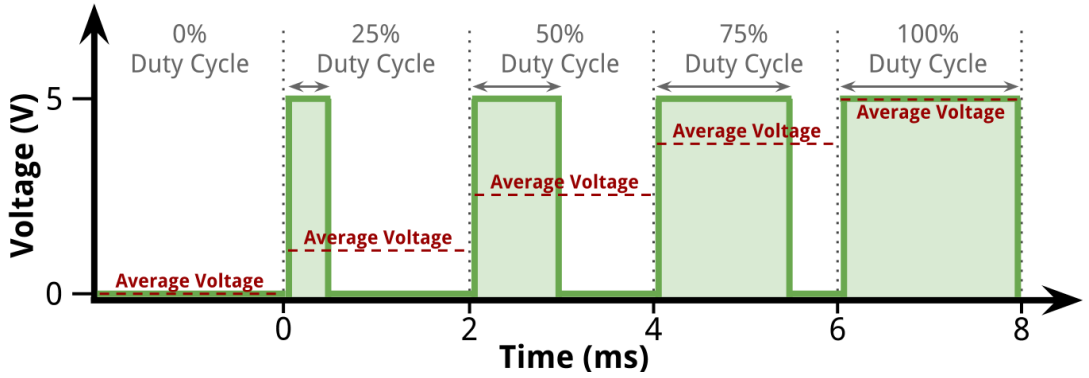



IMMAGINE 12: MODULAZIONE DELLA LARGHEZZA DI IMPULSO

### 8.4.3 Routine d'interruzione

I programmi che vengono eseguiti su sistemi a microcontrollore devono reagire a determinati eventi. Gli eventi differiscono in base alla durata, alla loro ripetizione e alla loro complessità. I controllori devono reagire ad ogni potenziale cambiamento sul pin di ingresso, così come la ricezione o la trasmissione di dati attraverso interfacce di comunicazione.

Con routine d'interruzione significa che il controllore arresta l'esecuzione del programma principale ed è dedicato alla funzione di interruzione. Se si utilizzano routine di interruzione multiple, è necessario impostare le priorità. Le priorità determinano quale routine ha il vantaggio nell'esecuzione. Quando si progetta un programma, è ragionevole classificare le routine di interruzione in base alle priorità e a come saranno eseguite. Se si utilizzano routine, che devono essere eseguite in un momento specifico, ha senso impostare un'alta priorità. Di seguito verranno presentate alcune delle più comuni routine di interruzione:

- **Interruzione a tempo**: L'interruzione a tempo è correlata al timer e viene eseguita in base al contatore del timer. L'interruzione all'orario viene eseguita all'ora esatta. L'interruzione può essere periodica o controllata attraverso il registro di stato dell'interruzione. Il controllo delle interruzioni significa che viene attivato quando necessario. Le interruzioni temporali sono spesso utilizzate per misurare il tempo e la sincronizzazione di diversi compiti periodici.
- **Interruzione a contatore:** L'interruzione a contatore è essenzialmente la stessa cosa dell'interruzione a tempo, ma non è strettamente correlata al

tempo. L'interruzione viene eseguita quando il contatore raggiunge un certo valore. Il conteggio del contatore è generalmente attivato da dispositivi esterni attraverso ingressi discreti. L'interruzione del contatore viene solitamente utilizzata per il conteggio degli impulsi.

- **Interruzione esterna:** L'interruzione esterna si attiva quando si modifica un potenziale di tensione sull'ingresso. L'interruzione esterna legge solo i valori logici 0 o 1. L'interruzione rileva il fronte positivo o negativo dell'ingresso discreto.
- **Interruzione della ricezione:** Questa interruzione viene utilizzata per la sincronizzazione con il mittente. Ciò significa che la routine rileva quando i dati sono sul bus. La routine di interruzione è spesso utilizzata nella ricezione asincrona dei dati. La ricezione asincrona dei dati significa che la trasmissione dei dati non è periodica in un determinato momento, ma casuale. L'interruzione garantisce che non si manchi di ricevere dati.
- **Interruzione della trasmissione**: L'interruzione della trasmissione dei dati consente di inviare il pacchetto di dati nel momento in questione. L'interruzione fa sì che il responsabile del trattamento non esegua altri compiti diversi dalla trasmissione dei dati.
- **Interruzione dell'ADC:** L'interruzione dell'ADC è il tempo necessario all'ADC per la lettura del valore analogico. Durante l'interruzione, il controller attende la conversione ADC e poi continua con altri compiti. L'interruzione dell'ADC è utile quando si ha la necessità di dati da ADC per i compiti successivi.
- **Interruzione del DAC:** Come nel caso dell'interruzione del DAC, l'interruzione del DAC ha un ruolo simile. Il DAC attiva per un breve periodo di tempo l'interruzione necessaria per la conversione del valore binario in uscita analogica.
- **Interruzione della modalità sleep:** L'interruzione della modalità sleep è utile quando il controllore non esegue costantemente le operazioni. Poi è possibile impostare il controllore per andare in modalità sleep. In questa modalità, ha un basso consumo energetico. L'interruzione della modalità sleep viene utilizzata in modo che il controllere utilizzi la maggior parte dell'energia per il monitoraggio di un determinato evento. Tutte le altre periferiche sono spente. All'evento impostato, l'interruzione risveglia il controllore che lavora con tutte le unità periferiche necessarie.

La routine di interruzione è un lavoro di codice di programmazione che viene eseguito all'interruzione. In questa routine, è importante cancellare il flag di interruzione. Il flag di interruzione è un bit nel registro di stato che viene impostato sul valore "1" quando si verifica l'interruzione. Se si vuole che l'interruzione venga eseguita

di nuovo, il flag deve essere resettato a "0". Quando si scrive un programma di routine di interruzione, si deve prestare particolare attenzione che il codice non funzioni più a lungo dell'interruzione. Ciò è particolarmente importante in caso di interruzioni che vengono eseguite periodicamente e rapidamente.

Ecco alcune situazioni nelle quali ha senso utilizzare le routine di interruzione:

- I compiti devono essere eseguiti in modo casuale.
- Lunghi intervalli tra due compiti.
- Il cambiamento di condizione è importante.
- Monitoraggio di brevi impulsi.
- Per il pieno funzionamento del controllore, devono essere eseguiti solo pochi compiti.
- Gli eventi sono generati da entrambi i dispositivi. Non ci sono picchi di segnale e nessun effetto di rimbalzo meccanico.

Alcune linee guida quando non c'è bisogno di routine di interruzione e quando i compiti sono eseguiti nel programma principale:

- L'operatore è umano.
- L'esecuzione dell'evento non è dipendente dal tempo.
- Gli impulsi di ingresso sono lunghi.
- Il segnale è coperto da rumore.
- Il programma principale non è lungo.

Quando si progetta un programma per il microcontrollore, è importante analizzare in modo completo l'esecuzione del programma. È necessario studiare cosa viene eseguito nella routine di interruzione e cosa verrà eseguito nel programma principale. Quando si utilizzano interruzioni multiple, bisogna essere cauti se l'esecuzione dell'interruzione richiede un ulteriore utilizzo della memoria RAM. Quando vengono attivate interruzioni multiple, il controllore può esaurire la memoria e è possibile perdere alcune informazioni dalle routine di interruzione. Il debug di problemi simili è molto difficile e richiede molto tempo.

### 8.4.3. Contatori e timer

I contatori e i timer sono unità periferiche chiave di ogni microcontrollore, spesso collegate ad altre unità. I microcontrollori hanno integrato più contatori diversi che differiscono per numero di bit (8, 16, 32 bit). I timer sono utilizzati per diversi compiti, come il ritardo, la misurazione del tempo, la misurazione della frequenza. L'uso più

semplice dei timer è l'uso del solo contatore. I timer possono generare diversi eventi, interruzioni o elaborare segnali PWM modulati.

Ogni timer è un contatore. Il contatore del timer può contare in maniera crescente o decrescente. La modalità conteggio può essere configurata all'interno del registro di controllo. La condizione del contatore può essere letta nel registro dati del contatore. Ogni contatore ha i propri registri di controllo, di stato e dati. La sua dimensione è determinata dalla lunghezza dei bit. La lunghezza del bit è il maggior numero possibile di contatori 2N-1, dove N è il numero di bit. Ad esempio, il contatore a 8 bit può contare da 0 a 255, mentre il contatore a 16 bit può contare da 0 a 65535. Se il contatore è collegato all'orologio, si ha un timer. Il timer fa scattare il conteggio a seconda dell'input tack (clock). Il conteggio del timer può essere attivato dal fronte positivo o negativo del clock. Il clock utilizzato nel timer può essere esterno o interno. In caso di clock interno, si utilizzi il clock della CPU anche se proviene da un cristallo esterno. Questa modalità di utilizzo dell'orologio è una modalità sincrona. Il contatto esterno dell'orologio significa che viene fornito un nuovo clock specifico per il timer. Il timer esterno non è correlato al clock della CPU, noto come modalità asincrona. L'orologio esterno è spesso utilizzato per il calendario (ora, giorni, mesi, ecc.). Per il calendario, il contatto dell'orologio è esattamente 32.768kHz. Il timer per il calendario è noto come contatore RTC (RTC è l'orologio in tempo reale). Il timer può essere configurato con diverse modalità di conteggio e tempi di conteggio. L'incremento di tempo del timer è il tempo di incremento di un contatore. L'incremento del tempo e la lunghezza del tempo possono essere impostati nei registri del timer.

Parametri del timer principale:

- **Clock:** Il timer clock è determinato dalla velocità del bus periferico per il timer in questione. La velocità del bus periferico è determinata dalle impostazioni di sistema del controllore. Alcuni controllori più semplici non hanno bus periferici separati per le singole unità e l'orologio è lo stesso per tutte le unità periferiche. È necessario determinare quale sorgente di clock sia in modalità asincrona o sincrona.
- **Scaling factor (prescaler):** Il fattore di scala determina la divisione dell'orologio del timer. Con questo fattore, viene determinato l'incremento del timer e la durata del conteggio quando si conta fino alla fine del registro dei dati dei contatori. L'incremento del timer è la risoluzione temporale che determina la precisione con cui si misura il tempo. Il valore del fattore di scala viene salvato nel registro PSC.
- **Period:** Il periodo di tempo del timer determina il periodo di conteggio. Ciò significa che il periodo più lungo equivale alla lunghezza del registro dei dati del contatore. La lunghezza dei dati del contatore è determinata da un numero di bit nel timer. Il periodo del timer viene salvato in ARR (auto reload register). Il periodo di conteggio determina il numero di unità conteggiate all'interno della lunghezza del registro ARR completo. Ciò

significa che il contatore non conta solo fino alla lunghezza massima dell'ARR, ma anche fino al valore del periodo impostato nel programma.

I contatori hanno altre due modalità di funzionamento. Questi sono noti come cattura in ingresso e cattura in uscita. La cattura degli ingressi viene utilizzata quando si vogliono contare certi eventi sui pin di ingresso del controllore. Ogni modifica (evento) dei valori logici "0" o "1" su uno specifico pin di ingresso viene trasferita dal contatore al registro di acquisizione degli ingressi. Questo registro può essere letto in codice di programmazione. La modalità di acquisizione degli ingressi è spesso utilizzata per la lettura di encoder diversi. La modalità di confronto delle uscite è simile alla cattura degli ingressi, ma qui vengono monitorati i pin di uscita. Le modifiche ai pin di uscita aumentano o diminuiscono le condizioni del contatore. La modalità di confronto delle uscite può anche attivare l'interruzione quando il contatore ha raggiunto il numero di incremento impostato.

Oltre ai contatori ampiamente utili, i microcontrollori contengono anche contatori speciali che non sono destinati ai suddetti usi. Si tratta di watchdog timer (WT) e sono destinati al monitoraggio del funzionamento del regolatore e del flusso del codice programma. Il timer WT ha un proprio orologio separato che non è collegato alla CPU. Il suo funzionamento è relativamente semplice. WT è impostato su un intervallo di tempo che può variare da millisecondi a secondi. All'inizio, il contatore conta fino a zero. Ogni attività, funzione, funzione, routine di interruzione azzera il contatore WT, il che significa che il timer ha il valore di partenza dopo ogni reset. Se non si è in grado di azzerare il contatore prima che raggiunga lo zero, il reset del controller hardware verrà attivato ("kick the dog"). Il controller viene riavviato. Con il timer WT si evita che il controller rimanga bloccato in cicli o non risponda.

### 8.4.5. Interfacce di comunicazione

Le interfacce di comunicazione consentono la comunicazione tra il microcontrollore e altri dispositivi esterni, come altri controllori, PC, sensori, ecc. In questo paragrafo, presenteremo solo interfacce che sono direttamente implementate nel chip del controller. Le interfacce di comunicazione si differenziano per diverse caratteristiche, come l'interfaccia parallela o seriale, la comunicazione sincrona o asincrona, la modalità punto-punto o di rete, la modalità half-duplex o full-duplex, via cavo o wireless. Qui di seguito verranno presentate solo le interfacce filo.

L'interfaccia seriale trasmette un bit in un solo clock. Ciò significa che dipende dalla velocità di clock. A seconda della frequenza di clock, si può determinare il numero di bit trasmessi per unità di tempo (bit/s), che viene chiamato velocità di comunicazione. Per questo motivo, l'interfaccia di comunicazione seriale richiede meno linee dati fisiche. L'interfaccia parallela trasmette ogni bit attraverso un bus fisicamente separato. Se si trasmettono dati a 8 bit, è necessario avere almeno 8 connessioni fisiche tra due

interfacce. Il collegamento in parallelo è più veloce di quello seriale. Il suo svantaggio è che richiede troppi bus fisici e più pin di controllore. La comunicazione parallela è spesso utilizzata per distanze molto brevi, display LCD, lettura di ADC e DAC veloci, sensori diversi, ecc. La connessione seriale è spesso utilizzata per distanze più brevi e più lunghe grazie alla sua semplicità.

In molti casi, la comunicazione tra due sistemi è bilaterale. Ciò significa che entrambi i dispositivi possono ricevere o trasmettere dati. La domanda successiva è: quando un dispositivo può trasmettere o ricevere dati. Se si utilizza la modalità full-duplex, entrambi i dispositivi possono ricevere o trasmettere dati contemporaneamente. Fisicamente questo significa che hanno due linee separate per la trasmissione (Tx) e la ricezione (Rx). In connessione seriale questo significa che sussistono due fili, uno per la trasmissione, uno per la ricezione. Nella modalità parallela a 8 bit, ciò significa 8 fili per la ricezione e 8 fili aggiuntivi per la trasmissione. Se si utilizza la modalità half-duplex, allora entrambi i dispositivi ricevono e trasmettono attraverso la stessa linea, il che significa che non è consentita la concorrenza. Questo significherebbe collisione e perdita di dati. Questa modalità richiede meno connessioni fisiche ma aumenta la complessità del protocollo di comunicazione.

Nelle interfacce di comunicazione, ci si imbatte spesso nel termine master-slave. Questa modalità è spesso utilizzata in modalità seriale. Il Device Master decide quando lo slave può accedere al bus e cosa può fare sul bus (ricevere o trasmettere dati). Se i dispositivi sono in rete, è consentito che solo un dispositivo sia il master e gli altri siano subordinati.

Le interfacce di comunicazione più comuni per i microcontrollori sono USART, SPI, I2C, ecc. Alcuni controller avanzati hanno anche integrato interfacce complesse, come CAN, USB e TCP/IP.

- ***USART*:** Si tratta di una connessione seriale (trasmettitore asincrono sincrono universale con ricevitore asincrono) che utilizza solo due linee per la comunicazione. Una linea è per la trasmissione di Tx e l'altra per la ricezione di Rx. La differenza tra USART e UART è che USART ha bisogno di una linea separata per il clock. UART ha un clock preimpostato con il quale sono collegati tutti gli altri dispositivi nel percorso di comunicazione. In USART il dispositivo ricevente utilizza il clock del mittente. L'immagine 13 presenta l'interfaccia di comunicazione UART.

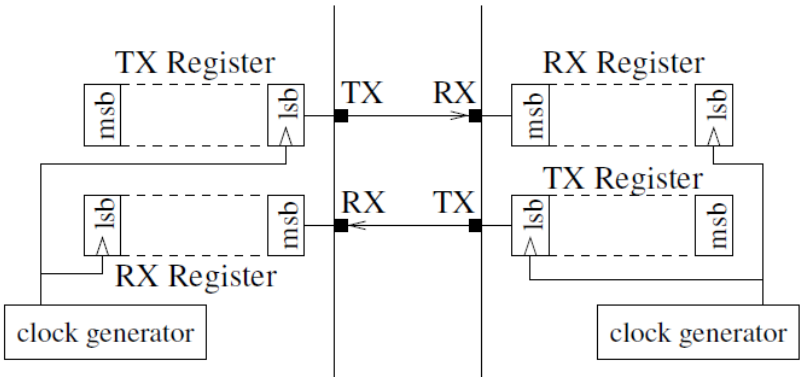



IMMAGINE 13: INTERFACCIA DI COMUNICAZIONE UART

I parametri regolabili dell'interfaccia seriale USART sono:

- ***Numero di bit di dati:*** Il numero di bit di dati determina quanti bit saranno catturati nei dati inviati. Questo può variare da 5 a 9 bit di dati. I più comunemente usati sono dati a 8 bit.
- ***Parity bit:***: L'utente può decidere se la parità sarà utilizzata o meno. La parità può essere un bit pari "1" o dispari "0". Viene utilizzato come semplice controllo sulla regolarità della comunicazione. Un esempio per l'uso di parità pari e dispari in dati a 7 bit è nell'immagine 14.

| 7 bits of data | (count of 1-bits) | 8 bits including parity | |
|---|---|---|---|
| | | even | odd |
| 0000000 | 0 | 00000000 | 00000001 |
| 1010001 | 3 | 10100011 | 10100010 |
| 1101001 | 4 | 11010010 | 11010011 |
| 1111111 | 7 | 11111111 | 11111110 |

IMMAGINE 14: PARITÀ PARI E DISPARI

- ***Stop bit:*** Lo stop bit annuncia quando il trasferimento è finito. Può essere lungo 1 o 2 bit.
- ***Velocità di trasferimento (baud rate):*** La velocità di trasferimento è determinata dal clock. Più alto è il clock, più veloce è il trasferimento. La maggior parte delle volte vengono utilizzate le seguenti velocità di connessione: 9.6kb/s, 38kp/s, 115.2kb/s, 1Mb/s, 10Mp/s.

La connessione seriale è nota con gli standard RS232, RS422, RS485, RS422, RS485, ecc. La RS232 è una connessione tra due punti in cui il livello di tensione del fronte positivo è determinato tra 3-15 V. Entrambi i dispositivi collegati alla RS232 devono essere sullo stesso potenziale e avere un GND comune. Per l'utilizzo di RS232 e UART sul

microcontrollore vengono utilizzate interfacce che aumentano o diminuiscono il livello di tensione per lo standard RS232. Spesso viene utilizzata anche l'interfaccia MX232.

RS422 è la stessa comunicazione della RS232, ma utilizza livelli di tensione differenziale. Permettono il trasferimento su lunghe distanze. I dispositivi non necessitano di GND comune.

L'RS485 è simile allo standard RS422, ma permette 32 dispositivi sullo stesso bus, mentre RS232 e RS422 ne permettono una sola coppia. In entrambi gli standard RS422 e RS485 la velocità di connessione diminuisce con la lunghezza della connessione.

- **SPI:** SPI (serial peripheral interface) è una connessione seriale destinata alla comunicazione tra dispositivi vicini. Le distanze consentite vanno da pochi centimetri al massimo di un metro. La comunicazione avviene in modalità full-duplex tra master e slave. L'interfaccia supporta anche la modalità a 3,2,1 fili.

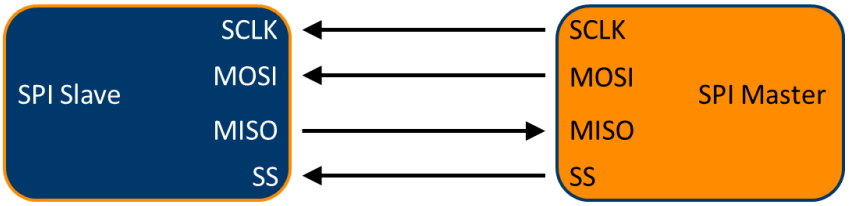



IMMAGINE 15: COMUNICAZIONE SPI

La comunicazione di SPI utilizza a pieno titolo quattro linee:

***MOSI:*** (master out slave in): Collegamento per la trasmissione dei dati dal master allo slave.
***MISO:*** (master in slave out): Connessione per la ricezione dei dati da slave a master.
***SCK:*** (system clock): Velocità di clock-comunicazione raggiunta.
***SS o CS:*** (slave select o chip select): Dispositivo di indirizzamento da parte del master.

La comunicazione SPI consente il collegamento di più dispositivi allo stesso bus SPI (MOSI, MISO, SCK). Ogni dispositivo ha solo pin SS/CS separati, come mostrato nell'immagine 16. La comunicazione SPI invia pacchetti di 8 bit. La velocità di comunicazione è correlata al clock della CPU del master e alla velocità di clock dello slave.

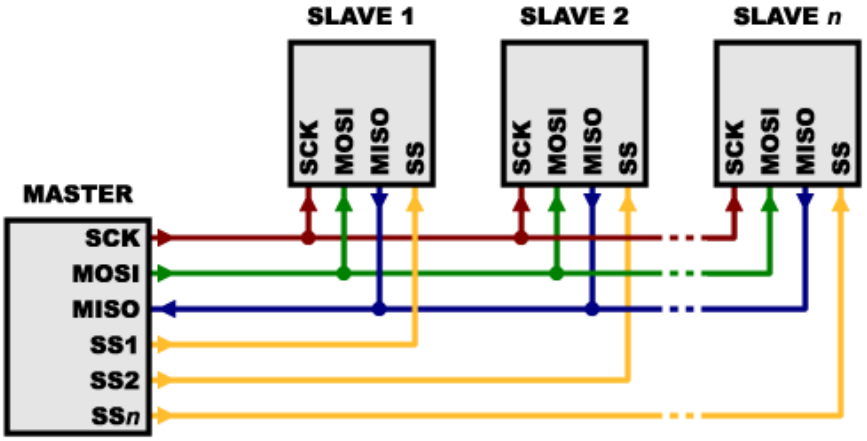



IMMAGINE 16: DIVERSI DISPOSITIVI SUL BUS SPI

- **IIC (I2C):** È una comunicazione seriale che abilita la modalità half-duplex. Utilizza anche il principio master-slave. Come SPI, l'IIC è destinato anche alla comunicazione tra dispositivi vicini. La velocità del bus IIC è determinata da tre fattori. La modalità lenta è 100 kbit/s, la modalità veloce 400 kbit/s e la velocità massima 3,4 Mbit/s. L'interfaccia di comunicazione richiede solo due linee. Una linea è dati SDA e la seconda è generata dal master. Il bus abilita il trasferimento dati a 10 o 7 bit. Il dispositivo sul bus IIC ha un indirizzo a 7 bit. All'inizio della comunicazione, il master indirizza il dispositivo con l'indirizzo, quindi inizia la ricezione e la trasmissione dei dati. L'interfaccia consente di collegare più dispositivi su un unico bus, come si vede nell'immagine 17.

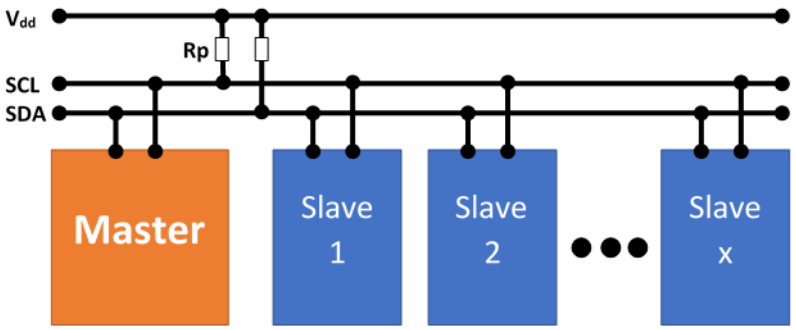



IMMAGINE 17: INTERFACCIA DI COMUNICAZIONE IIC

La comunicazione SPI è più veloce della comunicazione I2C e richiede fisicamente più pin del bus I2C. La comunicazione SPI è spesso utilizzata per la gestione di display grafici per schede SD e per catturare immagini dal chip ottico. I2C è un'interfaccia spesso utilizzata dove non ci sono limitazioni temporali rigorosamente definite.

I microcontrollori contengono spesso diverse interfacce di comunicazione. Per esempio, osservando il microcontrollore STM32F407 si possono trovare 6 interfacce USART, 3 interfacce SPI e 3 interfacce IIC.

## 8.3. Linee guida per il basso consumo energetico dei microcontrollori e gli aspetti ecologici

Ogni controllore può essere configurato in modo da avere il minor consumo energetico possibile. Il primo passo verso una progettazione ecologica richiede la scelta di un controllore in base alle sue caratteristiche e alle dimensioni del chip. La dimensione del chip è direttamente correlata ai materiali utilizzati, come la plastica dell'alloggiamento, i perni metallici o il silicio. Anche il materiale usato alla fine del ciclo di vita influisce sul riciclaggio. Quando si sceglie un controllore, non sceglieremo 144 pin chip e utilizzeremo solo pochi pin di ingresso e di uscita. Per questo, ha senso scegliere una versione più piccola della stessa famiglia. Un altro esempio sono i controllori ad alta efficienza. Quelli usano più energia dei semplici controllori. Per ogni dispositivo o applicazione, è importante valutare quale controllore utilizzare. Sul mercato sono presenti diversi controllori destinati a bassi consumi e lunga autonomia.

Quando si sceglie un controllore, è ragionevole considerare i seguenti punti:

- **Controller clock:** La regolazione dell'clock è strettamente correlata al consumo energetico e all'autonomia del sistema. La frequenza è proporzionalmente correlata all'energia. In ogni controllore è possibile impostare la CPU in determinati limiti raccomandati dal produttore. Se i compiti del controllore non richiedono molto tempo, allora il clock del controllore può essere ridotto fino a quando non soddisfa i criteri di tempo per i compiti. La maggior parte dei controllori hanno un clock gestito da un programma, il che significa che durante l'esecuzione di compiti complessi aumentando il clock si ottiene un'esecuzione veloce. Quando il controllore lavora su compiti più semplici e non impegnativi in termini di tempo, il clock viene abbassato nella misura in cui i compiti funzionano ancora normalmente.

- **Supply voltage:** La maggior parte dei controllori ha una tensione di alimentazione regolabile in una certa area. Anche la tensione di alimentazione è proporzionale al consumo energetico. Nella maggior parte dei casi, la tensione di alimentazione è anche correlata all'efficienza del controllore e al clock della CPU. Spesso s'incontra la limitazione che a una tensione più bassa non si può impostare un clock più alto. Quando si sceglie la tensione di alimentazione, è necessario scendere a compromessi tra efficienza e consumo energetico.

- **Spegnere i moduli non utilizzati:** Molte volte quando si progetta il codice del programma non si utilizzano molte periferiche. Il controllore permette che le periferiche siano spente, il che significa che non consumano energia aggiuntiva quando non sono in uso. L'utilizzo delle periferiche può essere gestito anche tramite codice programma.

- **Progettazione ottimale del codice:** Quando si progetta il codice, è importante considerare l'esecuzione. Quanto più breve è il codice, tanto minore è l'istruzione che il regolatore deve eseguire, il che significa che il regolatore può essere in modalità standby più a lungo. Quando si progetta il codice bisogna anche essere consapevoli del tipo di variabili selezionate e del loro numero. Nei programmi più grandi, può succedere che la variabile sia dichiarata e che poi venga usata raramente o addirittura non utilizzata affatto.

- **Modalità standby:** Quando il controllore abilita diverse modalità di standby è ragionevole utilizzarle, specialmente se sussistono lunghe pause tra un'esecuzione di un compito e l'altro.

**Riferimenti:**

[1] Günther Gridling, Bettina Weiss, 'Introduction to Microcontrollers', Vienn University of Technology press, 2007.

[2] http://www.mosaic-industries.com/embedded-systems/microcontroller-projects/raspberry-pi/gpio-pin-electrical-specifications