Enrico Tombelli

Docente presso
ITC "A. Volta" - Bagno a Ripoli - Firenze
(e.tombelli@libero.it)

Interfacciamento

INTERFACCIAMENTO

- SISTEMA 8086

- SEGNALI DELLA CPU 8086

La CPU 8086 ha 40 piedini (PIN) di cui:

- Sedici linee per il bus DATI/INDIRIZZI multiplexato AD_0%AD_{15} (ovvero svolgono sia la funzione di bus dati che di bus indirizzi, ma in tempi diversi: T_1⇒INDIRIZZI; T_2%T_4⇒DATI). Le linee d'indirizzo A_{16}%A_{19}, sono anch'esse multiplexate con i segnali ausiliari S_3%S_6 (segnali di stato: T_1⇒INDIRIZZI; T_2%T_4⇒STATO). Questi ultimi indicano il tipo di segmento indirizzato (S_3%S_4) NB: S_5%S_7 non utilizzati

S_4	S_3	Segmento
0	0	Extra Data
0	1	Stack
1	0	Codice
1	1	Data

- Tre linee per l'alimentazione (Vcc = +5V e GND due pin: 1 e 20) e uno per il Clock (CLK)
- Le rimanenti linee, compresi i segnali di stato S_3%S_6 rappresentano i segnali di controllo, che possono essere ingressi o uscite, ed alcuni sono bidirezionali. Dato che l'8086 ha due modi di funzionamento, i segnali di controllo variano in base al modo impostato (il modo di funzionamento è stabilito dal livello del pin MN/MX e può essere MINIMO o MASSIMO). Nel modo MINIMO (più semplice e meno efficiente) i segnali di controllo sono:

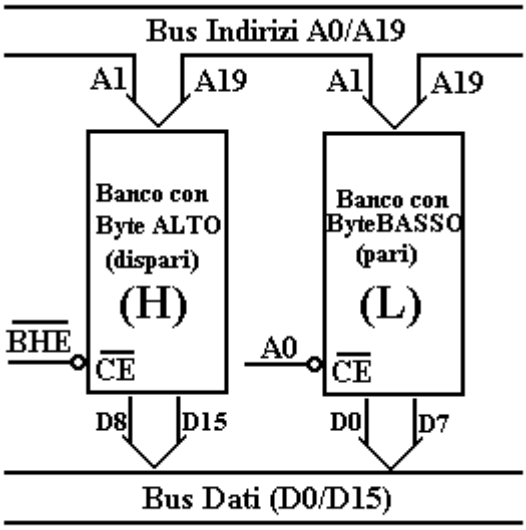
$\overline{\text{READY}}$	Ingresso	Questo segnale è un consenso alla CPU e proviene dalla memoria che richiede di inserire uno stato di attesa per poter avere il tempo di fornire o acquisire il dato in quanto essa non è ancora pronta. La CPU reagisce inserendo dopo il terzo stato interno, uno stato di attesa (T_{WAIT}) rallentando così la velocità di esecuzione.
INTR	Ingresso	Interrupt Request: è la richiesta alla CPU da parte di una periferica di in servizio di I/O. Esso è testato dalla CPU nell'ultimo stato interno del ciclo macchina e se attivo, la CPU inizia, quando abilitata, un ciclo d'Interrupt accogliendo la richiesta e inviando indietro il segnale di IntAcknowledge.
$\overline{\text{TEST}}$	Ingresso	Questo pin viene testato dalla CPU su richiesta da programma tramite l'istruzione di WAIT. Se trovato attivo la CPU si pone in uno stato "ozioso" [IDLE] sospendendo ogni attività lasciando l'operatività del sistema ad altri processori esterni (p.e. coprocessore matematico).
NMI	Ingresso	non-maskable-interrupt: Piedino d'INTERRUPT non mascherabile. è simile al segnale di INTR (interrupt), con la differenza che mentre il servizio d'interrupt ordinario può essere disabilitato, quello non mascherabile deve essere svolto in ogni caso.
RESET	Ingresso	Questo segnale serve per reinizializzare la CPU. Se rimane attivo per più di 4 stati interni la CPU si blocca per ripartire con la situazione iniziale (vengono azzerati i registri interni) nel momento in cui il segnale viene disattivato.
CLK	Ingresso	È il segnale di temporizzazione (Clock) con duty cycle al 33% generato dal circuito integrato esterno 8284.
MN/$\overline{\text{MX}}$	Ingresso	Definisce se si lavora in modo massimo (MAXIMUM⇒0; MINIMUM⇒1).
$\overline{\text{BHE}}$	Uscita	Byte High Enable (abilitazione del byte alto della cella di memoria). serve per abilitare le celle dispari della memoria, le quali, oltre ad essere indirizzate singolarmente possono esserlo anche in coppia con le celle pari in modo da

formare celle di 16 bit.

$\overline{RD}$	Uscita	(READ): Segnale d'attivazione della lettura in memoria o in una porta di I/O.
M/$\overline{IO}$	Uscita	Identifica il tipo d'accesso in lettura/scrittura se in memoria (posizione M=1) o in una porta di I/O (posizione I/O=0).
$\overline{WR}$	Uscita	(WRITE): Segnale d'attivazione della scrittura in memoria o in una porta di I/O.
$\overline{INTA}$	Uscita	Interrupt ACKNOWLEDGE: è la risposta di accettazione da parte della CPU del servizio d'Interrupt per l'I/O.
ALE	Uscita	Address Latch Enable: è generato dalla CPU per attivare il caricamento dell'indirizzo A_0/A_{19} nel latch d'interfaccia col bus indirizzi durante il primo stato interno (T1).
DT/$\overline{R}$	Uscita	Data Transmitter/Receiver. Segnala al buffer three-state d'interfaccia col bus dati se il dato sul bus dati è in SCRITTURA o in LETTURA.
$\overline{DEN}$	Uscita	Data Enable: abilitazione del buffer three-state d'interfaccia col bus dati per il transito del dato sul bus dati in SCRITTURA o in LETTURA.
HOLD	Ingresso	Il segnale di HOLD è la richiesta da parte di un dispositivo esterno, della disponibilità del Bus (completo) per operare da "Master". Tipico dispositivo che richiede tale situazione è il DMA che si occupa specificatamente delle operazioni di I/O in modo molto più veloce della CPU. Se la CPU trova attivo il segnale di hold essa pone le sue uscite in ALTA IMPEDENZA (sconnettendosi così dal bus lasciandolo a disposizione del processore che ne ha fatto richiesta). Si dice allora che la CPU è in stato di WAIT e in risposta attiva il segnale di HOLDA (HoldAcknowledge) sulla stessa linea.
HOLDA	Uscita	

- ORGANIZZAZIONE FISICA DELLA MEMORIA 8086

La capacità di indirizzamento della CPU I8086 è di 1 MByte (da 00000h a FFFFFh), corrispondente a 20 linee di indirizzo (A_0 % A_{19}) e rappresentabile con 5 cifre esadecimali.



La lunghezza di ogni singola cella di memoria è di 1 Byte, ma il normale accesso ad essa avviene per word di 16 bit. Questo è dovuto al fatto che il bus dati è a 16 bit (D_0 % D_7 e D_8 % D_{15}) e che ad ogni accesso ci si riferisce a 2 celle di memoria per volta (A_0=0 cella pari ovvero parte bassa del dato D_0 % D_7. A_0=1 parte alta del dato D_8 % D_{15}).

BHE	A_0	Tipo Trasferimento
0	0	Intera Word
0	1	Byte alto (Da indir. dispari)
1	0	Byte basso (Da indir. pari)
1	1	Nessun trasferimento

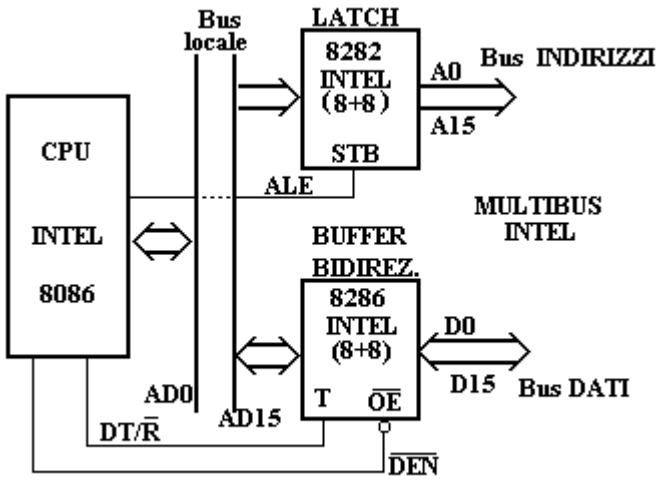
È comunque possibile trattare la memoria, cella per cella, ovvero un byte alla volta. Infatti si possono effettuare trasferimenti sul bus di una intera Word (16

bit) oppure di un singolo byte grazie alla configurazione assunta da un piedino della CPU denominato BHE nonché alla linea A_0.

In particolare il segnale BHE (Byte High Enable) abilita il banco contenente i byte alti, ovvero le parti alte delle word, mentre A_0 abilita il banco delle celle con i byte bassi delle words.
NB: è necessario precisare che quando si opera un accesso in memoria del tipo "word", ovvero a 16 bit su un indirizzo dispari, avviene un "doppio accesso". P.e. MOV AX,[101], deposita nell'accumulatore "AX" il contenuto della cella 101 e della cella 102. Per fare questo, l'8086 svolge due accessi a 8 bit in memoria utilizzando un ciclo macchina in più.

- CONFIGURAZIONE DEL SISTEMA E MULTIPLEXING DEL BUS

La necessità di limitare il numero di piedini a 40 pin, ha imposto l'utilizzo della tecnica di MULTIPLEXING delle 16 linee dei dati (D_0 % D_{15}) e delle prime 16 linee d'indirizzo (A_0 % A_{15}).



Questo significa che il bus dati e parte del bus indirizzi sono le stesse linee fisiche, e differenziano nel tempo la loro funzione a secondo della necessità.

Infatti tali linee trasportano un indirizzo nel primo stato interno (T_1) e un dato a 16 bit nel rimanente ciclo macchina (T_2, T_3, T_4 ecc.).

Data la necessità di mantenere comunque stabile l'indirizzo della memoria, mentre questa fornisce il dato, la CPU deve essere interfacciata col bus esterno (MULTIBUS Standard Intel) tramite un latch che memorizza temporaneamente l'indirizzo. L'impulso di caricamento del latch è fornito dalla CPU tramite il piedino ALE (Address Latch Enable - Attivo alto).

Il latch è stato appositamente progettato dalla Intel per questo scopo ed è il Chip "8282". Dato che presenta 8 canali, per interfacciare il bus locale con quello esterno, ne occorrono come minimo 2.

Lo stesso discorso vale per la parte dati, ma in questo caso la cosa è leggermente più complessa in quanto è bidirezionale, pertanto occorre un'interfaccia che controlli il flusso dei dati nei due sensi. Tale interfaccia è stata progettata anch'essa appositamente dalla Intel ed è il Chip "Data Tranceiver 8286" che accoppia in modo bidirezionale il bus locale col MULTIBUS intel.

Anche in questo caso, i segnali di controllo per l'accoppiatore bidirezionale, cioè l'abilitazione e la direzione, sono emessi da due piedini della CPU; In particolare:

- DEN Data Enable. Segnale di abilitazione per l'8286 (Attivo basso)
- DT/R Data Transmitter/Receiver. Segnale di direzione dati per l'8286.

Basso $\Rightarrow$ Receiver (Esterno $\rightarrow$ CPU)
Alto $\Rightarrow$ Transmitter (CPU $\rightarrow$ Esterno)

Dato che il bus dati è a 16 bit, e l'8286 supporta 8 linee, di questi chip ne occorrono 2.

S_4	**S_3**	(S_7, S_6 e S_5 non sono utilizzati)
0	0	Extra Data
0	1	Stack
1	0	Code (o nessuno)
1	1	Data

Le altre linee d'indirizzo (A_{16}%A_{19}) sono anch'esse multiplexate, ma con alcuni segnali di stato (S_3%S_6) che durante gli stati interni T_2, T_3 ecc. identificano il tipo di segmento indirizzato.

Necessita, quindi, un altro latch 8282 (parzialmente utilizzato) per il multiplexing delle rimanenti linee d'indirizzo.

- MODI DI FUNZIONAMENTO

La CPU 8086 prevede due modi di funzionamento:

1) Modo Minimo. In questo caso i segnali emessi dalla CPU sono quelli ordinari, già visti per le CPU a 8 bit (Intel 8080).

   La situazione "Modo Minimo" è stabilita dall'operatore mantenendo alto il piedino di MN/MX, che predispone un modo di funzionamento o l'altro.

   Data la necessità d'avere disponibili un numero di segnali di controllo maggiore per l'eventuale utilizzo della CPU per sistemi più potenti, è comunque possibile attuare una logica diversa (Modo Massimo) che permette di disporre maggiori prestazioni dal sistema di controllo.

2) Modo Massimo. È attivato portando basso il pin MN/MX della CPU, e questo provoca la trasformazione dei segnali relativi ai pin della CPU che vanno da 24 a 31 come in tabella:

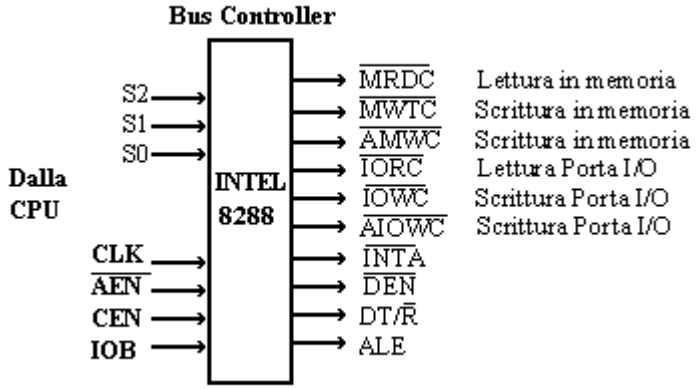
	MODO MIN		MODO MAX
31	HOLD	*	RQ/GT_0
30	HLDA	*	RQ/GT_1
29	WR	*	LOCK
28	M/IO	*	S_2
27	DT/R	*	S_1
26	DEN	*	S_0
25	ALE	*	QS_0
24	INTA	*	QS_1

Apparentemente il numero dei segnali di controllo sembra inalterato. In effetti, i segnali della vecchia configurazione (HOLD, HLDA, WR ecc.) sono comunque disponibili in quanto ricostruiti tramite decodifica.
Si può notare, infatti, che alcuni segnali della CPU (INTA, RD, WR, HALT, ecc.) non sono mai attivi contemporaneamente. La Intel ha pensato pertanto di sfruttare questo fatto generando tali segnali in modo decodificato tramite la combinazione di quelli sui piedini S_2, S_1 e S_0 secondo la tabella (ricordo che tali segnali sono multiplexati e sono attivi solo dal secondo stato interno in poi).

L'occorrenza di tali segnali, può quindi essere segnalata tramite la DECODIFICA di 3 bit (pin S_0, S_1, S_2) della CPU, secondo la tabella.

S_2 S_1 S_0	OPERAZIONE	SIGNIFICATO
0 0 0	Interrupt Acknowledge	Riconoscimento interruzione. Viene attivato il segnale INTA.
0 0 1	Read I/O	Comando di lettura di una porta di I/O. viene attivato il segnale IORC
0 1 0	Write I/O	Comando di scrittura di una porta di I/O. Vengono attivati i segnali IOWC e (in alternativa) AIOWC.
0 1 1	HALT	La CPU è nello stato di blocco ovvero di HALT
1 0 0	Instruction Fetch	La CPU sta prelevando un codice operativo, ovvero è in fase di FETCH.
1 0 1	Read Memory	Comando di lettura in memoria. Viene attivato il segnale MRDC.
1 1 0	Write Memory	Comando di scrittura in memoria. Vengono attivati i segnali MWTC e (in alternativa) AMWC.
1 1 1.	Nessun segnale	Il bus è inattivo

La decodifica è realizzata da un Chip della Intel anch'esso appositamente progettato, ovvero il "Bus Controller 8288". I pin di ingresso aggiuntivi (CLK, AEN, CEN, IOB) non provengono direttamente dal microprocessore e servono per stabilire la modalità di funzionamento e per il controllo degli altri segnali. Il segnale IOB identifica il modo con il quale viene gestito lo scambio di dati con le porte di I/O. Nel caso IOB=0 si ha il funzionamento ordinario del sistema (system bus mode). Con IOB=1 (I/O bus mode) tutte le linee di controllo di lettura/scrittura IORC, IOWC, AIOWC e INTA per una operazione di I/O sono abilitate contemporaneamente e il tipo di operazione è stabilita solo dai segnali DEN (notare che quando questo segnale è generato dal bus controller è attivo alto) e DT/R. esattamente come la memoria.



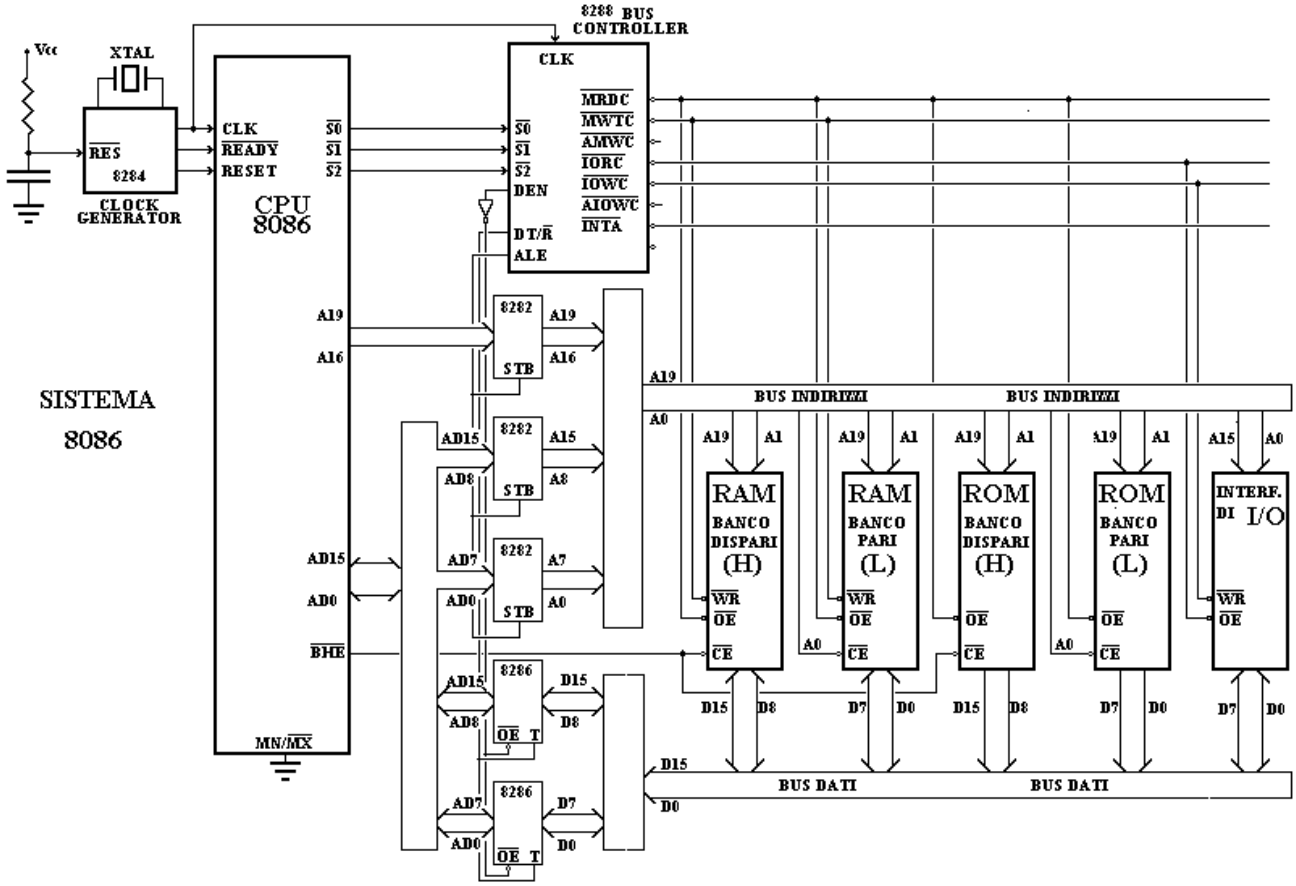
Con questo sistema è possibile il governo completo del bus controlli come specificato dallo standard “MULTIBUS” ideato dalla Intel e utilizzato nella maggior parte delle applicazioni con CPU tipo 8086/8088 (p.e. Personal computer IBM).

La scelta del modo di funzionamento della CPU (MAX/MIN) dipende dall’applicazione che deve essere attuata. In ogni caso, il modo MAX è obbligatorio là dove vengono utilizzati chip ausiliari e di supporto come il coprocessore matematico (8087) o l'interfaccia parallela "Bus Arbiter" (8289).

Nel modo Massimo, oltre ai segnali già visti (alcuni di questi sono riprodotti dal Bus Controller (8288), si hanno i segnali:

S_2,S_1,S_0	Uscita	Decodifica secondo la tabella dei segnali ricostruiti
INTA		Vedi modo MINIMO
TD/R		‘’ ‘’ ‘’
DEN		‘’ ‘’ ‘’
ALE		‘’ ‘’ ‘’
MRDC		Lettura in memoria
MWTC		Scrittura in memoria
AMWC		Scrittura in memoria (alternativa)
IORC		Lettura da una porta di I/O
IOWC		Scrittura in una porta di I/O
AIOWC		Scrittura in una porta di I/O (alternativa)
MCE/PDEN		Logica d'interrupt
$\overline{RQ}/GT_0$	I/O	Request/Grant Servono per sincronizzare la comunicazione con altri processori (come il coprocessore matematico 8287)
$\overline{RQ}/GT_1$		
$\overline{LOCK}$	Uscita	(Blocco) Serve per far conoscere ai processori esterni alla CPU che essa non è disponibile a cedere il controllo del bus e quindi deve inibire la loro richiesta. Tale consenso è necessario per impedire l’inibizione della CPU durante l’esecuzione di parti di programma "Ininterrompibili".
QS_1, QS_0	Uscita	Queue status: Identifica lo stato della coda di prefetching secondo i seguenti codici

QS_1	QS_0	Operazione
0	0	Nessuna
0	1	Prelievo primo Byte del codice operativo dalla coda
1	0	Svuotamento coda (per salto)
1	1	Prelievo Byte successivo dalla coda



SCHEMA LOGICO DEI COLLEGAMENTI IN MODO MAX

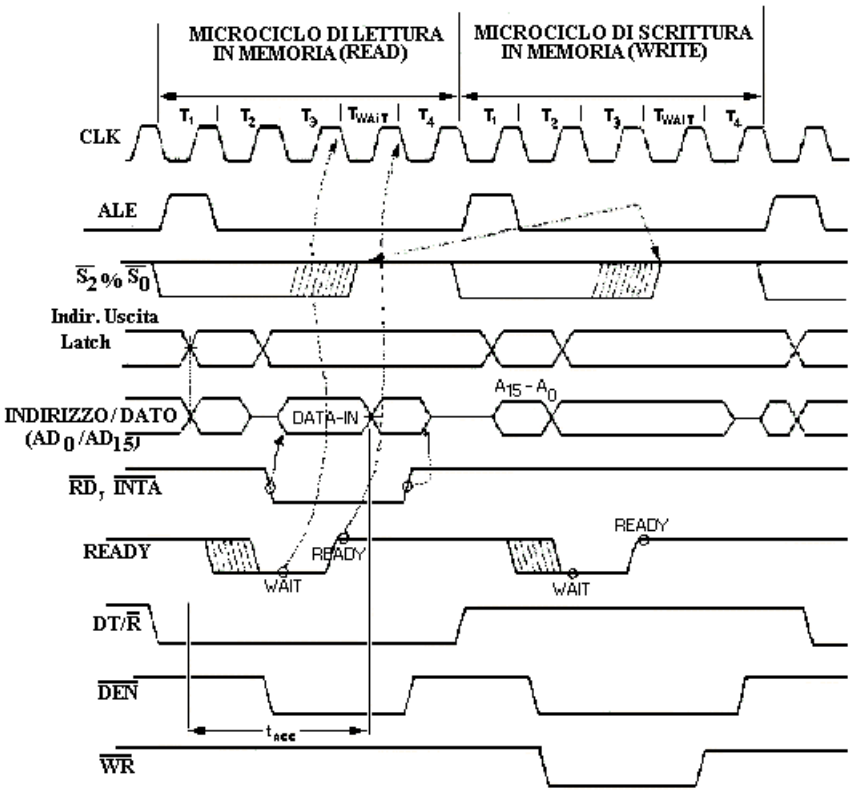
- TEMPIFICAZIONE DEI SEGNALI CPU I8086

I processi descritti precedentemente (gestione fisica della memoria) sono tempificati secondo il diagramma seguente.

Alcune osservazioni:

- Nel primo stato interno (al centro di T_1 sia nel ciclo di scrittura che di lettura) viene attivato il segnale "ALE" per il caricamento dell'indirizzo nel latch d'interfaccia con il bus indirizzi (AD_1/AD_{15} contiene l'indirizzo). Successivamente "ALE" viene disattivato e il microprocessore è in attesa dei dati in ingresso (DATA-IN nel ciclo di lettura) o dispone i dati in uscita (DATA-OUT nel ciclo di scrittura).

- Per completezza del grafico viene attivato, nel secondo stato interno (T_2 sia nel ciclo di scrittura che di lettura) il segnale di READY (attivato da una eventuale memoria esterna lenta rispetto alla CPU). Tale segnale provoca la generazione di un ulteriore stato interno (T_{WAIT}) di attesa. Questa possibilità è necessaria ogni qualvolta si abbia memorie lente rispetto alla CPU. Infatti tale attesa adatta la velocità della CPU a quella della memoria. Durante lo stato di attesa la CPU testa nuovamente il segnale di READY e se lo trova ancora attivo introduce ulteriori stati di attesa, finché la memoria non ritira l'attivazione del segnale di READY. Per questo, quando le memorie sono lente rispetto alla CPU si dice che quest'ultima lavora con 1, 2, 3 ecc. WAIT-STATE. Ciò significa che la velocità della CPU non è sfruttata al massimo, in quanto frenata dalla lentezza delle memorie.



TEMPIFICAZIONE DEI SEGNALI IN UN CICLO DI LETTURA/SCRITTURA IN MEMORIA

- Il segnale di DT/R è alto nel ciclo di scrittura (dove è attivo anche il segnale di WR) e basso in quello di lettura. Sia in lettura che in scrittura viene attivato il buffer three-state d'interfaccia col bus dati tramite il segnale DEN.

- CIRCUITO DI RESET

L'operazione di RESET consiste nell'inizializzazione del sistema e in particolare della CPU. In altre parole svolge l'azzeramento dei registri interni della CPU e inizializza la logica sequenziale che governa il sistema. Tale operazione può essere svolta automaticamente all'accensione od occasionalmente in modo manuale dall'operatore qualora ci sia l'esigenza. La CPU effettua il reset nel momento in cui è attivato il comando di RESET. Tale segnale è normalmente pilotato da due circuiti che ne controllano l'attivazione. I due circuiti servono per l'attivazione MANUALE e AUTOMATICA del reset.

- ATTIVAZIONE AUTOMATICA:

E' avviata senza l'intervento dell'operatore in quanto necessaria dopo l'accensione del sistema. Una volta azionato l'interruttore dell'alimentazione, la tensione da essa prodotta segue un transitorio durante il quale il sistema non può funzionare. Per questo, il circuito di reset automatico attiva immediatamente il comando di reset e lo mantiene per un tempo determinato sufficiente per stabilizzare la tensione di alimentazione al valore corretto, dopodiché ritira l'azione di reset e il sistema si avvia allo start.

La temporizzazione può essere realizzata tramite un circuito R-C (vedi figura). Il condensatore si presuppone inizialmente scarico, ma all'accensione inizia a caricarsi secondo la curva esponenziale:

$$Vc_{(t)} = Vcc(1-e^{-t/RC})$$

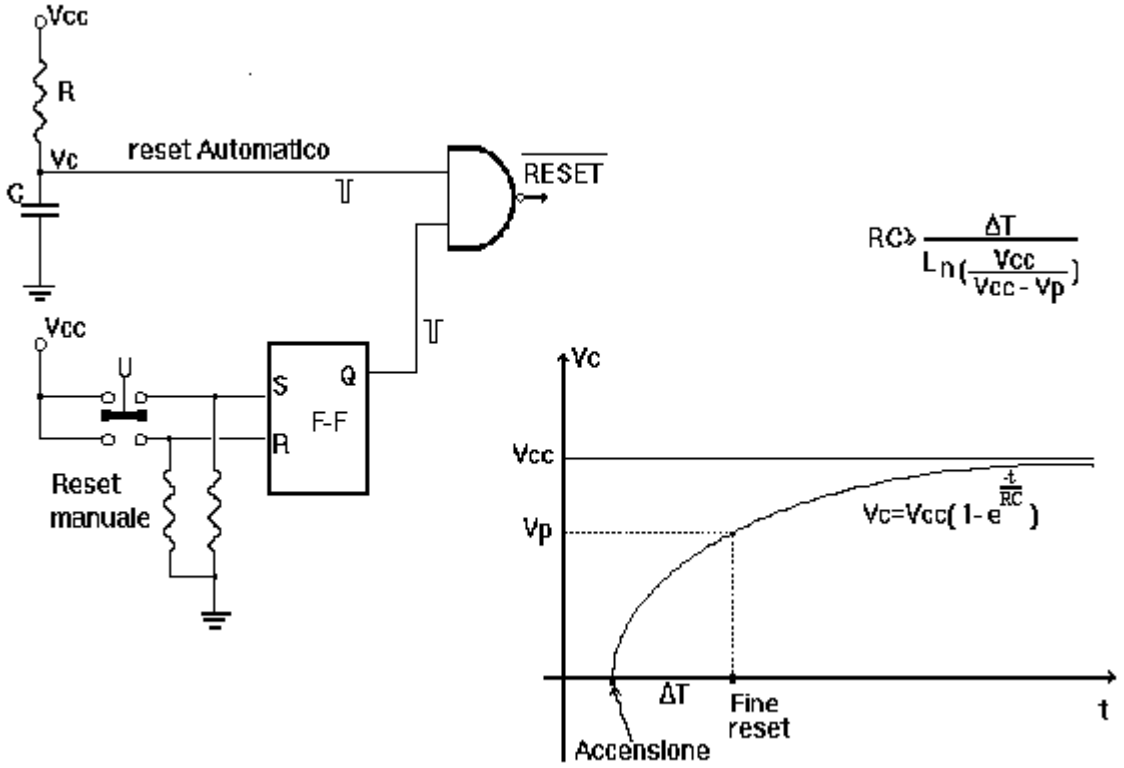
Finche la tensione Vc si mantiene al di sotto della soglia di scatto della porta AND, il reset rimane attivo. Quando la tensione ai capi del condensatore (Vc) raggiunge il livello di scatto della porta AND il comando di reset viene ritirato. è possibile calcolare la relazione che lega la costante di tempo RC al tempo di attivazione del RESET. Per questo basta imporre nella espressione della carica del condensatore che quando è trascorso un tempo sufficientemente lungo (normalmente 10 impulsi di clock, dato che p.e. nella CPU 8086 il reset deve restare attivo almeno per 4 impulsi di Clock) la tensione Vc abbia raggiunto la tensione di scatto della porta AND (in logica TTL è 2,5V), ovvero

$Vc_{(10T)} = 2{,}5V$ NB: $10T=10/f$

Pertanto

$$Vc_{(10T)} = Vcc(1-e^{-10T/RC}) > 2{,}5V$$

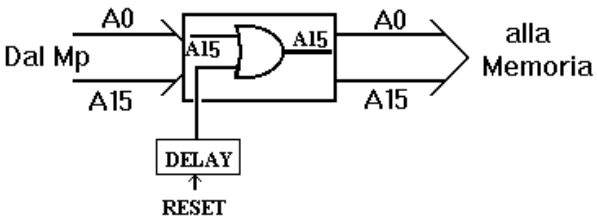
Esplicitando RC si ottiene l'espressione minima della costante di tempo.



- ATTIVAZIONE MANUALE

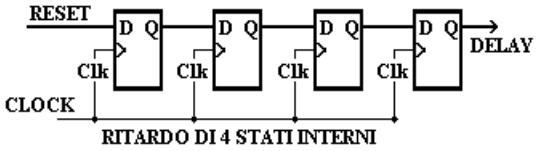
Può essere utile per l'operatore poter reinizializzare il sistema in qualunque momento. Questo è lo scopo del circuito per i reset manuale. Esso è composto da un interruttore che mantiene alto il livello del segnale in condizioni normali, ma che lo porta a zero quando viene premuto il pulsante dall'operatore. Il F-F- set/reset ha lo scopo di evitare ALEE DINAMICHE dovute al rimbalzo dei contatto dell'interruttore (circuito antirimbalzo) che potrebbero creare dei malfunzionamenti dovuti alla natura impulsiva del segnale che si verrebbe a creare altrimenti.

In alcuni casi il circuito di reset ha anche il compito di mantenere l'indirizzo preliminare della prima istruzione da eseguire che deve trovarsi necessariamente in memoria ROM.



Forzatura di A16 a 1 da parte del comando di RESET

Dato che al reset vengono azzerati i registri, compreso il PC, la prima istruzione da eseguire è nella cella 0000h. Le prime celle di memoria però sono normalmente zona RAM e quindi vuote. Per questo l'azione del reset deve forzare l'indirizzo sul bus indirizzi in modo da farlo puntare nella zona in ROM a indirizzi più alti e quindi scavalcando inizialmente l'azione del PC (p.e. se la memoria ROM parte dall'indirizzo 8000h il reset agirà sulla linea A15 forzandola a livello alto per tutta la durata della fase di feth della prima istruzione. Ovviamente tale istruzione deve essere un salto all'inizio del programma di funzionamento del sistema che provvederà a caricare a sua volta il programma di monitoraggio o un sistema operativo). Il ritardo (funzione DELAY) necessario per mantenere alta la linea A15 può essere realizzato tramite una catena di FF tipo "D". Ogni FF realizza un ritardo di uno stato interno.

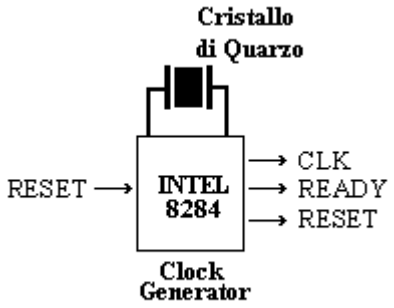


RITARDO DI 4 STATI INTERNI

L'azione non è necessaria per la CPU 8086 in quanto il reset non azzera il registro CS, anzi setta tutti i suoi bit a "1". Pertanto, coma già visto la prima istruzione si trova all'indirizzo FFFF:0000h dove è possibile implementarla come memoria ROM

- CLOCK DI SISTEMA

Il clock del sistema è prodotto da un apposito circuito che genera il segnale ad una frequenza ben precisa e stabilita in base alle caratteristiche di un cristallo di QUARZO. Esso vibra con un cadenza costante e determinata (ogni tipo di cristallo ha una frequenza di vibrazione propria del taglio col quale è sezionato). Per questo viene sfruttato il fenomeno della PIEZOELETTRICITÀ, per la quale se un cristallo viene sottoposto a deformazione elastica esso reagisce producendo una tensione fra due delle facce di cui è composto. Tale fenomeno è reversibile, in quanto se al cristallo viene applicata una differenza di potenziale esso si deforma di conseguenza producendo un fenomeno di risonanza elettromeccanica.



La CPU 8086, non è dotata internamente di circuiti generatori della forma d'onda del clock, ed è necessario effettuare esternamente questa operazione a partire dal cristallo di quarzo che genera la frequenza campione di riferimento.

Può essere utilizzato allo scopo il chip Intel "Clock Generator 8284" che, a partire dalla frequenza del cristallo ad esso collegato, genera il segnale di clock con un "DUTY CICLE" del 33% (alto per il 33% del periodo).

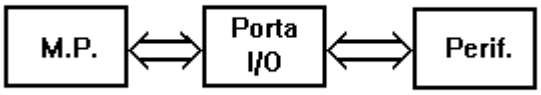
Si può notare che il circuito per la generazione del clock integra anche la parte relativa ai circuiti di reset.

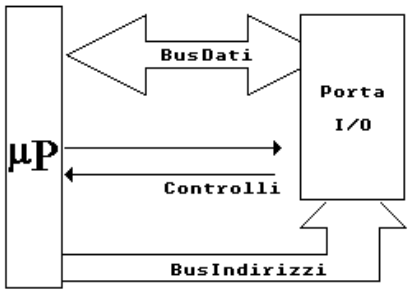
- I/O DEL MICROPROCESSORE

Consiste nella gestione delle interfacce con le periferiche (Stampanti, Tastiere, Monitor, Modem ecc..).

A causa della incompatibilità che esiste nella maggior parte dei casi, fra la CPU e la periferica, (velocità diversa, potenze non sufficienti, ecc..) e della eventuale lontananza che può sussistere fra questi, (che porterebbe al degradamento del segnale lungo il percorso), il collegamento deve essere effettuato interponendo fra i due dispositivi, un sistema di interfacce.

- INTERFACCIA





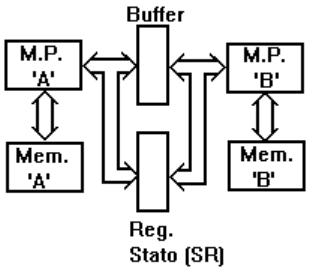
L'interfaccia ha il compito di:

1. Selezionare la periferica interessata alla operazione di I/O.
2. Rigenerare la potenza necessaria per pilotare la trasmissione dei segnali.
3. Adattare (tradurre) i vari tipi di sistemi di comunicazione intercorrenti fra i due sistemi colloquianti (p.e. conversione parallelo/seriale e viceversa).
4. Adattare le velocità fra i dispositivi comunicanti (sistema della Mail-box) e sincronizzare il trasferimento.

La più semplice interfaccia di questo tipo è detta PORTA I/O. Ad ogni periferica viene quindi associata una porta I/O.

Nel caso in cui il MP sia collegato a più periferiche, e quindi saranno collegate al circuito, più porte I/O, la selezione della periferica avverrà tramite la selezione della relativa porta.

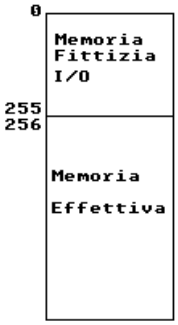
- PORTA I/O



Ogni porta I/O è costituita al minimo da:

- Registro di stato (o Controllo); provvede a memorizzare lo STATO della periferica in relazione alla comunicazione.
- Registro Buffer; ospita i vari dati che devono essere scambiati fra le varie unità.

P.e. due CPU che comunicano fra loro.



La CPU "A" preleva dei caratteri dalla memoria "A" e li invia alla CPU "B" che li ripone nella memoria "B".

Il registro di stato (SR) identifica il fatto che il registro buffer è pieno (e quindi la CPU "B" deve leggere) oppure che il buffer è vuoto (e che quindi la CPU "A" deve scriverci il dato da trasferire). In particolare si possono utilizzare due caratteri per il riconoscimento della situazione:

 # ⇒ Buffer pieno.
*** ⇒ " vuoto.**

- COLLEGAMENTO AL BUS

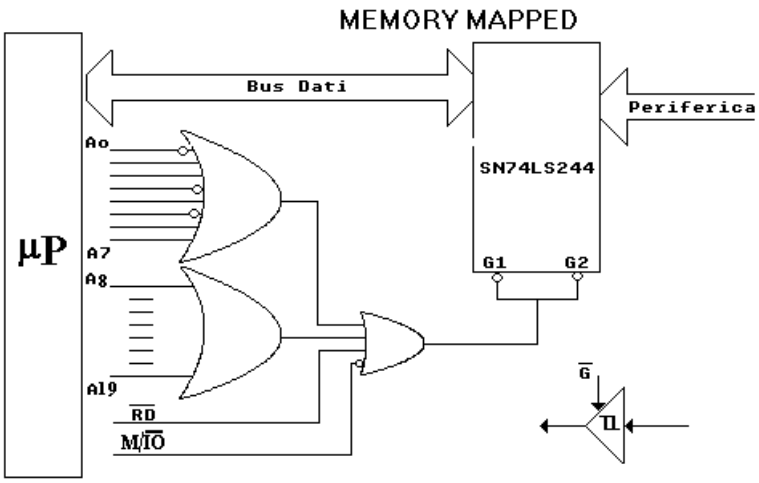
Le tecniche di collegamento al bus permettono di assegnare un indirizzo ad ogni porta di I/O, in modo da poterla identificare correttamente. Per questo tutte le porte di I/O sono collegate al BUS indirizzi nello stesso modo col quale sono collegate le memorie. La loro selezione avviene, come per la memoria, tramite i segnali di selezione, lettura e scrittura (CS, WR, RD, CE, ecc.). I dati vengono scambiati attraverso il bus dati avvalendosi di porte bidirezionali.

Un esempio di collegamento di porte I/O al bus può essere fatto associando alle porte di I/O indirizzi compresi fra 0 e 255. Ogni registro della porta I/O, può essere quindi selezionato, utilizzando solo le prime 8 linee del bus indirizzi (A_0 % A_7).

- TECNICHE DI COLLEGAMENTO AL BUS

Le principali tecniche di collegamento al bus delle porte di I/O, sono due e provvedono alla assegnazione di indirizzi ai vari registri buffer della interfaccia.

- MEMORY MAPPED
- I/O MAPPED (I/O ISOLATO)



- TECNICA MEMORY MAPPED

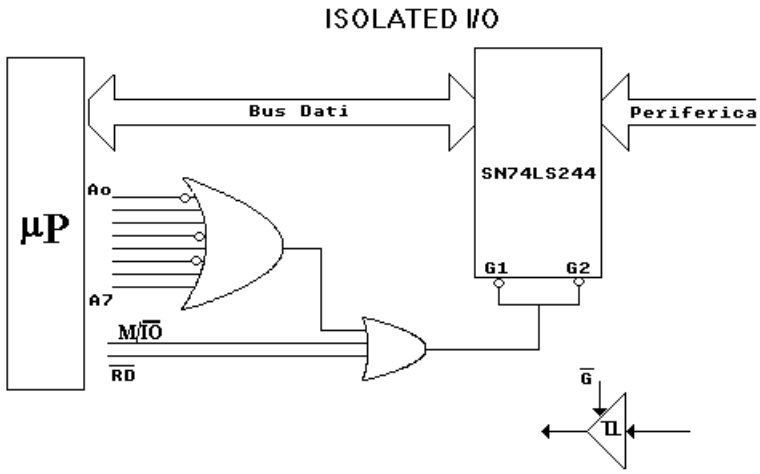
Il registro che rappresenta una porta di I/O è visto come una locazione di memoria che ha un suo indirizzo specifico.
Tutte le combinazioni d'indirizzamento, sono quindi spartite fra celle di memoria e registri di I/O. Esempio: affinché i segnali di lettura (RD) e selezione (M/IO) arrivino agli ingressi di selezione della porta I/O SN74LS224, è necessario che le linee d'indirizzo assumano i valori seguenti:

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Indirizzo
0	0	1	0	1	0	0	1	00029h

Le linee da A_8 a A_{19} sono a livello basso. La porta di I/O in figura è abilitata quando il segnale M/IO è a livello alto (ovvero indirizzo la memoria) e impongo sul bus indirizzi il valore 00029h. Per prelevare un dato dalla porta in questione posso utilizzare l'istruzione per la lettura in memoria, p.e.

MOV AL,[0000:0029].

- TECNICA I/O ISOLATO



Nella tecnica di I/O Isolato, l'indirizzamento avviene nello stesso modo che per l'altra tecnica, ma il segnale di abilitazione (M/IO) non ha la stessa funzione che nel caso precedente. Esso è generato da opportune istruzioni di I/O dedicate le quali portano M/IO a livello basso (0).

P.e.
IN AH, [Indirizzo di I/O] ovvero
IN AH,[29h] per le operazioni di ingresso.

OUT [indirizzo di I/O],AL ovvero
OUT [29h],AL per le operazioni di uscita.

- PREGI / DIFETTI

La tecnica Memory Mapped permette di utilizzare istruzioni generiche, le stesse utilizzate per la gestione della memoria. Ha però lo svantaggio di non rendere disponibile la capacità di indirizzamento in memoria al 100 %, in quanto parte di essa deve essere utilizzata per la selezione delle porte di I/O.

- INTERFACCIAMENTO I/O NEL SISTEMA 8086

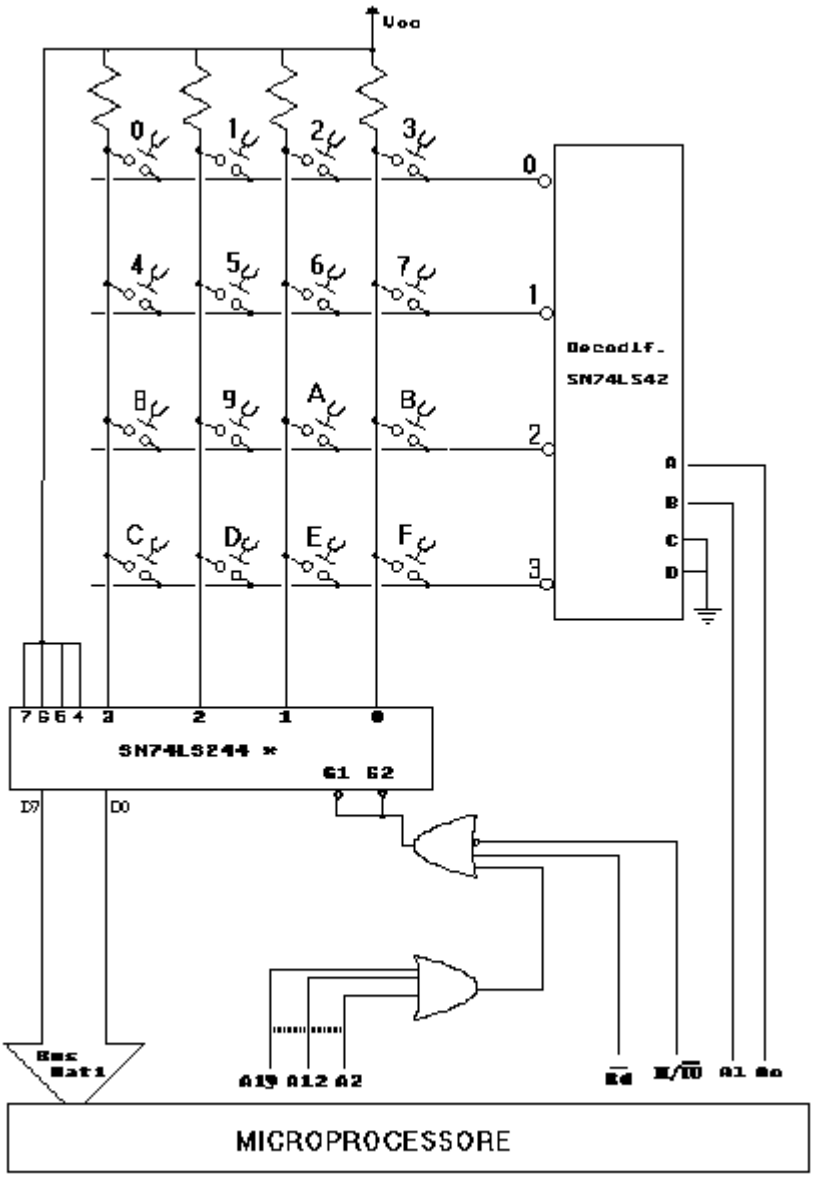
Nel caso che il collegamento al bus non sia di tipo memory mapped, la CPU dispone d'istruzioni per la gestione diretta delle porte di I/O (IN, INW, OUT, OUTW).

Con questa tecnica sono disponibili 64K indirizzi, corrispondenti a 64 Kbyte o 32 Kword di 16 bit ciascuna. Le linee del bus controlli che definiscono il tipo di operazione (lettura o scrittura), e attivano la memoria o le porte di I/O, dipendono dal modo operativo:

- Modo MINIMO: M/IO Abilita la memoria quando è a livello alto, mentre a livello basso abilita le porte di ingresso-uscita. La lettura e la scrittura sono, invece, abilitate indipendentemente dai soliti segnali di RD e WR.
- Modo MASSIMO: In questo caso la gestione è più complessa, dato che concorrono 4 segnali al funzionamento, come specificato in tabella.

Modo MAX	**LETTURA**	**SCRITTURA**
MEMORIA	MRDC	MWTC oppure AMWC
PORTE di I/O	IORC	IOWC oppure AIOWC

Le linee di indirizzo utilizzate per le operazioni di I/O sono quelle che vanno da A_0 ad A_{15}. Le rimanenti sono forzate a zero. È possibile, per rendere il sistema più veloce, disporre di indirizzamenti diretti (che permettono di sfruttare i primi 256 indirizzi utilizzando solo le prime 8 linee del bus indirizzi) oppure sfruttare l'intera capacità di indirizzamento utilizzando l'indirizzamento di tipo indicizzato tramite il registro DX. È possibile, inoltre, utilizzare delle porte di I/O a 8 bit, ma in questo caso è necessario assicurarsi che se collegate alla parte bassa del bus dati (D_0-D_7) a queste corrispondono indirizzi pari (A_0 = 0).



SISTEMA DI GESTIONE DI UNA TASTIERINA ESADECIMALE

- ESEMPI DI COLLEGAMENTO AL BUS

- INTERFACCIA PER UNA TASTIERINA ESADECIMALE

La tecnica di collegamento al bus utilizzata è la Memory Mapped. Le linee di indirizzo A_2, ..., A_{19} identificano la porta di I/O relativa alla tastierina, ovvero il buffer three-state SN74LS244. Quando le linee d'indirizzo assumono i valori nulli, ovvero quando sul bus indirizzi è posto uno dei valori:

00000h, 00001h, 00002h, 00003h

A_1 e A_0, collegati al decoder, portano a livello basso una delle righe secondo il seguente schema:

A_1	A_0	Riga
0	0	0
0	1	1
1	0	2
1	1	3

Pertanto, la tastiera si configura come una memoria di 4 celle, i cui indirizzi sono:

Riga	**indirizzo**
0	00000h
1	00001h
2	00002h
3	00003h

Ogni riga è trattata, a tutti gli effetti come una cella di memoria (a sola lettura). La riga è selezionata ciclicamente. E' quindi portata a "massa" la relativa linea.

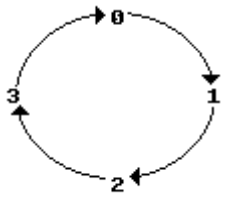
Se un tasto è premuto, p.e. all'incrocio fra la riga "1" e la colonna "3" (ovvero il tasto identificato dal numero "4"), è portata a zero la relativa colonna, quando viene selezionata la riga ove si è premuto il tasto.

Esempio:

↓ **COLONNA SELEZIONATA (3)**

riga selezionata	Bus dati (HEX)	D_7, ... , D_4,	**D_3** .... ,	D_0	
0	F F	1 1 1 1	**1** 1 1 1		
1	F 7	**1 1 1 1**	**0** **1 1 1**		←**RIGA SELEZIONATA (1)**
2	F F	1 1 1 1	**1** 1 1 1		
3	F F	1 1 1 1	**1** 1 1 1		

(⇒PREMUTO TASTO "4")



NOTA: Ogni scansione di riga deve essere fatta ad intervalli regolari di tempo, più piccoli della persistenza della pressione del dito sul tasto. Dato però che tale persistenza (dovuta ai tempi umani), è dell'ordine del 1/10 sec, mentre il periodo di scansione è dell'ordine del msec, tale problema non sussiste.

- PROBLEMA DEI RIMBALZI

Può essere risolto, in modo SW, andando a rileggere più volte (p.e. 10 volte) il tasto stesso, e verificando poi la coerenza del fenomeno.

- PROGRAMMA PER LA GESTIONE DELLA TASTIERA

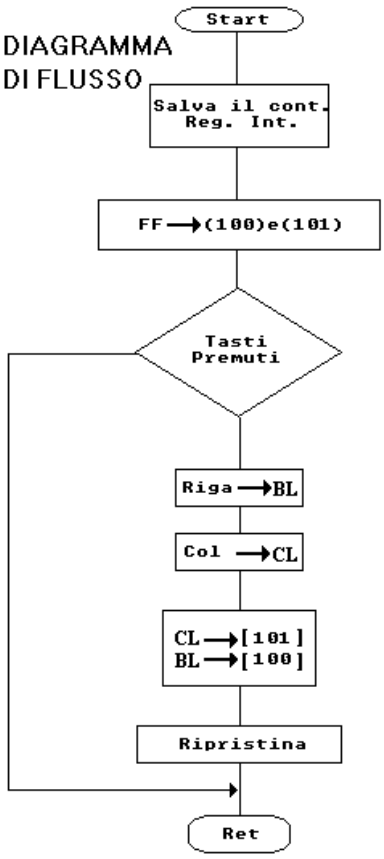
La routine appresso descritta, rileva se è stato premuto un tasto oppure no. Se è stato premuto un tasto, ne rileva le coordinate deponendole nelle celle di memoria "100h" e "101h". Si presuppone che il segmento di lavoro sia 0000h (tutti i numeri sono esadecimali)

Reg. "BL" contatore di RIGA.
Reg. "CL" contatore di COLONNA

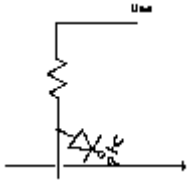
Successivamente deposita tali coordinate nelle locazioni di memoria:

[100h] RIGA e [101h] COLONNA

```
        PUSH    AX          Salva il contenuto
        PUSH    BX          di tutti i  registri
        PUSH    CX          interni della CPU
        PUSH    DX

        MOV     AL, FFh     Inizializzazione delle
        MOV     [100h], AL  celle di memoria 100h
        MOV     [101h], AL  e 101h.

        MOV     BL,00h      Controlla la pressione
        MOV     AL,[0000h   di un tasto sulla
                ]
        CMP     FFh         riga N. 0.
        JNZ     P1

        INC     BL          Controlla la pressione
        MOV     AL,[0001h   di un tasto sulla
                ]
        CMP     FF          riga N. 1
        JNZ     P1

        INC     BL          Controlla la pressione
        MOV     AL,[0002h   di un tasto sulla
                ]
        CMP     FFh         riga N. 2.
        JNZ     P1

        INC     BL          Controlla la pressione
        MOV     AL,[0003h   di un tasto sulla
                ]
        CMP     FFh         riga N. 3.
        JZ      P3

P1:     MOV     CL,FFh      Ricerca la colonna
P2:     INC     CL          relativa al tasto
        RCR     AL          premuto.
        JC      P2
        MOV     [100h],BL
        MOV     [101h],CL

P3:     POP     DX          Ripristina il
        POP     CX          Contenuto dei
        POP     BX          Registri Interni.
        POP     AX

        RET                 ritorno al programma
                            chiamante
```



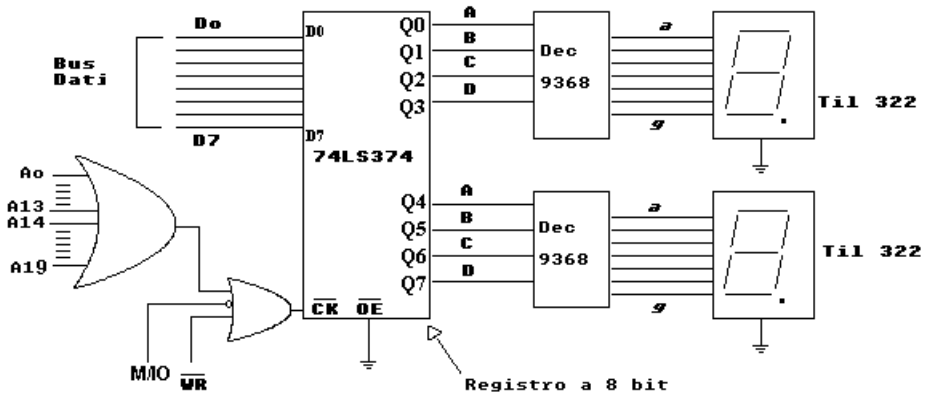
NOTA:Il programma non prevede la pressione contemporanea di più tasti. Nel caso che ciò succeda viene rilevata la pressione del tasto che si trova nella riga e nella colonna con indice più basso. Si deve tener conto che la pressione contemporanea di due tasti cortocircuita le relative righe del decoder e dato che una delle due è selezionata (0V) mentre l'altra non lo è (5V) può avvenire la rottura del decoder.



Per la risoluzione del problema si possono utilizzare dei diodi al Germanio (tensione di soglia 0.2V) interposti ad ogni incrocio della matrice formante il circuito della tastierina, come in figura.

- INTERFACCIAMENTO CON UN DISPLAY A SETTE SEGMENTI

A_{19}	A_{18}	A_{17}	A_{16}	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



INTERFACCIA PER DISPLAY A 7 SEGMENTI

La tecnica utilizzata è di tipo Memory Mapped, e il registro a 8 bit di figura è indirizzato da tutte le linee del bus indirizzi. Il registro ha indirizzo 0000:0000h e il caricamento del dato da visualizzare può essere effettuato tramite le due istruzioni:

MOV AL,53h
MOV [0000h],AL

Tali istruzioni fanno comparire sui display la coppia di cifre "5" (digit più significativo) e "3" (digit meno significativo).

NOTA sugli integrati: SN74LS374 registro a 8 bit (Buffer unidirezionale); 9368 decoder BCD/7 segmenti.

- SISTEMA COMPOSTO

Esistono dei dispositivi che gestiscono automaticamente tastiera e display, p.e. l'INTEL 8279 gestisce una tastierina esadecimale e un gruppo di 8 display

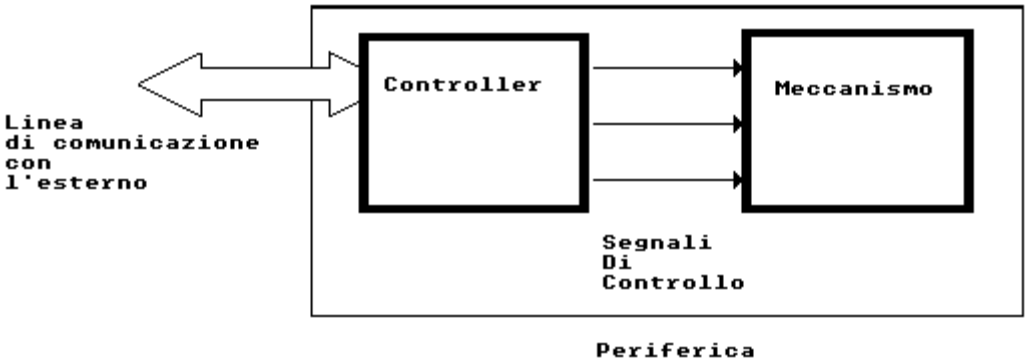
- CHIP AUSILIARI DELL'8086

Fra i chip ausiliari dell'8086 sono disponibili vari chip dedicati:

- ♦ PORTA PARALLELA Intel 8255, capace di pilotare fino a tre periferiche contemporaneamente.
- ♦ PORTA SERIALE Intel 8251 e 8250
- ♦ DMAC Intel 8237 e 8257
- ♦ PIC Intel 8259 (Per la gestione dell'interrupt)
- ♦ CONTROLLORE PROGRAMMABILE DI CRT Intel 8275 e 6845
- ♦ CONTROLLORE DI FLOPPY DISK Intel 8272 (Per disk drivers a singola e doppia densità)
- ♦ BUS ARBITER Intel 8289 per l'interfacciamento con altri processori.
- ♦ TIMER programmabile (PIT) Intel 8253.
- ♦ COPROCESSORE MATEMATICO Intel 8087.

- PROBLEMATICHE RELATIVE ALL'INTERFACCIAMENTO

Il funzionamento di ogni periferica è supervisionato da un dispositivo detto "CONTROLLER" che si trova all'interno della periferica stessa; p.e. in una stampante, il controller genera i segnali per l'avanzamento dei motori passo-passo che trascinano il carrello e fanno ruotare il rullo e che controllano il movimento degli aghi della testina di stampa.



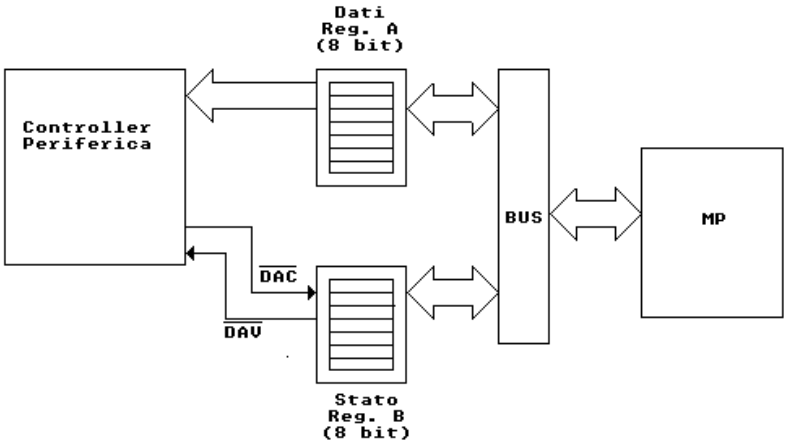
- COMUNICAZIONE FRA LA PERIFERICA E IL MICROPROCESSORE

Come già spiegato, il controller non può essere collegato direttamente al M.P., ma, deve essere interfacciato tramite una porta di I/O.

Devono essere inoltre stabilite le modalità con le quali deve avvenire lo scambio di dati fra la periferica (Controller) e la CPU. Tali modalità sono dette "TECNICHE DI HANDSHACKING".

Nei casi più banali (p.e. la tastierina o il display dell'esempio precedente), il colloquio è molto semplice in quanto non ci sono problemi di velocità o rigenerazione di potenza.

In genere, invece, per rendere più affidabile e veloce la trasmissione, vengono scambiati attraverso l'interfaccia, anche alcuni segnali di controllo; In particolare, vengono scambiati dei segnali di "STATO" che servono a sincronizzare il colloquio.



- REG. "A" (Buffer dati 8 bit). In esso vengono messi dalla CPU i dati (un carattere per volta)
- REG "B" (Registro di Stato). Di esso, in questo caso, vengono utilizzati solo due bit, ognuno corrispondente ad un singolo segnale di stato.

I segnali tipici utilizzati sono:

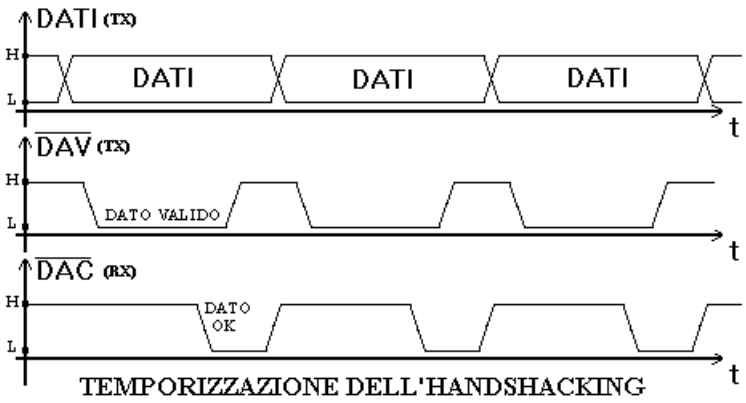
- DAC (Data Accept), che viene generato dalla periferica e avverte la CPU che il dato nel buffer è stato letto e che quindi il buffer è vuoto.
- DAV (Data Valid) , che viene generato dalla CPU e avverte la periferica ha messo un dato nel buffer, e che è possibile leggerlo.

N.B. I segnali di controllo sono, normalmente, ATTIVI BASSI.

Una possibile evoluzione dei segnali di stato può essere la seguente:

1. Si parte dalla condizione iniziale DAV=0 (buffer pieno - Dato valido) DAC=1 (Dato non ancora letto). La CPU ha caricato un carattere nel buffer e ha segnalato alla periferica che è valido, ovvero che tutti i transitori si sono estinti e il dato è leggibile.

2. Può darsi che la periferica non possa leggere immediatamente il carattere nel buffer, perché impegnata in altre funzioni (p.e. nella stampa del carattere precedente o addirittura perché non in linea o spenta). Non appena la periferica e disponibile legge il dato dal buffer, e segnala alla CPU l'avvenuta lettura portando il DAC a zero.
3. La CPU risponde disattivando il DAV e la periferica rimane in attesa.



TEMPORIZZAZIONE DELL'HANDSHACKING

4. Al passo successivo, la CPU prepara il carattere da inviare alla periferica e quando pronta lo carica nel registro buffer, dopodiché segnala alla periferica la validità del dato esaurendo così il ciclo delle segnalazioni.

Da notare che le varie segnalazioni sono insite nei fronti di discesa dei segnali di stato, che pertanto possono essere rimpiazzati da impulsi negativi.

Il protocollo tipico di HANDSHACKING è quello fra il PC (o meglio l'interfaccia parallela) e la stampante. Esso utilizza tre segnali di STROBE (Dato valido), BUSY (stampante non disponibile) e ACKNOWLEDGE (dato stampato). In effetti in questo tipo d'interfaccia, essendo bidirezionale e comunque molto complessa, prevede l'utilizzo anche di altri segnali.

- TECNICHE DI SCHEDULING

Tali tecniche riguardano il colloquio fra la CPU e l'interfaccia con il controller della periferica, attraverso la porta di I/O. Tra i più ricorrenti problemi d'interfacciamento abbiamo:

1. La CPU deve accorgersi che una o più periferiche sono pronte o richiedono una operazione di I/O.
2. Nel caso che ci siano più periferiche collegate alla CPU, quest'ultima deve individuare quella che ha richiesto il servizio di I/O.
3. Nel caso che più periferiche richiedano contemporaneamente una operazione di I/O, la CPU deve decidere a quale periferica concedere per prima il servizio di I/O.

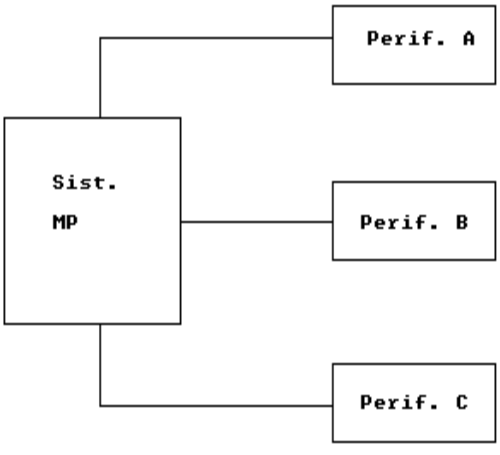
Questo tipo di problemi può essere risolto utilizzando una delle seguenti tecniche di scheduling:

- POLLING (test dei flag)
- INTERRUPT (interruzione)
- DMA (accesso diretto in memoria).

- POLLING

È il sistema più semplice ed è realizzato "via SW", ma presenta problemi di efficienza. Consiste nell'interrogare periodicamente i flag di stato delle periferiche concorrenti alla CPU.

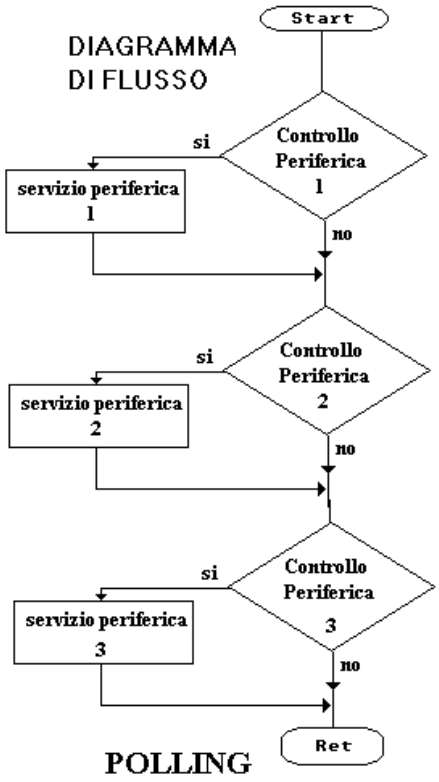
Periferica	Indirizzo Buffer	Indirizzo Flag 'DAC'	Indirizzo Flag 'DAV'
A	1001h	1000h bit 0	1000h bit 4
B	1002h	" " 1	" " 5
C	1003h	" " 2	" " 6



P.e. sono previste 4 periferiche collegate al bus tramite altrettante interfacce costituite ognuna da un registro buffer e da due flag di stato.

PROGRAMMA DI POLLING

```
POLLING:  PUSH  AX          Salva il contenuto
          PUSH  BX          di tutti i  registri
          PUSH  CX          interni della CPU
          PUSH  DX

          MOV   AL,[1000h]  Consulta i flag

          SAR   AL          delle periferiche
          JC    NO_CY_A     e chiama le relative
          CALL  PERIF_A     routine di servizio.

NO_CY_A:  SAR   AL
          JC    NO_CY_B
          CALL  PERIF_B

NO_CY_B:  SAR   AL
          JC    NO_CY_C
          CALL  PERIF_C

NO_CY_C:  POP   DX          Ripristina il
          POP   CX          Contenuto dei
          POP   BX          Registri Interni.
          POP   AX

          RET               ritorno al programma
                            chiamante

PERIF_A:  PUSH  AX          Salva l'accumulatore

          MOV   AL,[1000h]  setta il DAV
          OR    AL,10h      (AL= x x x 1 x x x x)
          MOV   [1000h],AL

          MOV   AL,[BX]     Carica il carattere nel
          MOV   [1001h],AL  Buffer e incrementa il
          INC   BX          Puntatore.

          MOV   AL,[1000h]  Resetta il DAV
          AND   AL,EFh      (AL= x x x 0 x x x x)
          MOV   [1000h],AL

          POP   AX          Ripristina l'accum.

          RET               Ritorna al P.P.

PERIF_B:                    Routine servizio
.....                       I/O periferica 'B'
```



PERIF_C:	Routine servizio I/O periferica 'C'

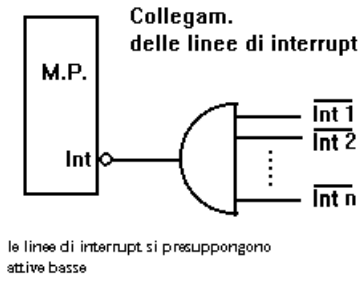
La ruotine di servizio della periferica "A" preleva un carattere dalla memoria (precisamente dalla cella puntata da BX e lo ripone nel buffer, poi incrementa BX. Se BX non viene modificato al di fuori della routine, è possibile trasferire automaticamente un blocco intero di memoria.

- PRIORITÀ

Con la tecnica "POLLING", il problema della priorità è risolto naturalmente disponendo opportunamente la sequenza di test dei vari flag.

- INTERRUPT

In questo caso la richiesta di servizio parte per iniziativa della periferica stessa che invia un segnale alla CPU detto 'INTERRUPT'. Tale segnale arriva ad un determinato piedino del chip CPU (piedino d'interrupt) che quando attivato, genera una chiamata ad una routine di servizio che effettua l'operazione di I/O opportuna. La chiamata è comunque filtrata dal flag d'interrupt presente nel registro di stato, in modo da poter "Mascherare" da programma la chiamata stessa.



In alcune CPU vi sono più piedini d'interrupt, in modo da poter gestire semplicemente le chiamate da parte di più periferiche. Generalmente, oltre al piedino d'interrupt di tipo ordinario, c'è anche un altro piedino per l'interrupt di tipo NON MASCHERABILE.

Nel caso venga attivato l'interrupt di tipo non mascherabile, la richiesta di servizio di I/O non può essere rifiutata da parte della CPU, come avviene per l'interrupt di tipo ordinario, ma deve essere accolta comunque.
L'interrupt di tipo non mascherabile, è generalmente attivato in occasioni di emergenza, p.e. nel caso in cui venga meno l'alimentazione. Esiste in questo caso la possibilità di effettuare una operazione di salvataggio di alcune informazioni particolarmente importanti in una memoria non volatile, nel tempo in cui la tensione di alimentazione scende dal valore intercettato al valore minimo per il funzionamento del sistema.

- SEQUENZA D'INTERRUZIONE

1) La periferica lancia il segnale d'interrupt, portando a livello basso la relativa linea (Il segnale d'interrupt è generalmente ATTIVO BASSO). Nel caso in cui più periferiche possano generare l'interrupt, tutte le linee confluiranno ad una porta OR (se le linee sono attive basse necessita una porta AND) la cui uscita viene collegata all'unico piedino d'interrupt della CPU. In tale modo, basta che una o più periferiche lanci l'interrupt, affinché la CPU venga sollecitata.
2) La CPU finisce di eseguire l'istruzione in corso dopodiché, testa il piedino d'interrupt, e se questo è attivo (ovvero se una periferica lo ha attivato) analizza la situazione del flag d'interrupt nel registro di stato.
3) Se il flag d'interrupt è abilitato, il sevizio di I/O viene concesso e la CPU genera un segnale d'interrupt accolto (Interrupt Acknowledge).
4) Viene prima di tutto salvato il PC (alcune CPU, prevedono il salvataggio automatico del SR, altrimenti, ciò deve essere previsto in fase di programmazione) nello STACK.
5) Il contenuto del PC, viene rimpiazzato con l'indirizzo della routine di servizio di I/O. Nel caso coesistano più periferiche devono essere risolti i problemi:

   - Individuazione della periferica che ha richiesto il servizio

- Gestione della priorità nel caso di richieste simultanee.

Entrambe i due problemi possono essere risolti in uno dei tre modi:

A) La routine di servizio di I/O provvede ad effettuare l'interrogazione dei registri di stato delle interfacce fino a trovare quella che ha generato l'interrupt. Questo sistema è in pratica una via di mezzo fra la tecnica di polling e quella d'interrupt classico (Soluzione SW)

B) Esiste una rete apposita (DAISY CHAIN - INTERRUPT VETTORIZZATO) che provvede, in modo rigido, a individuare la periferica ed eventualmente ad assegnare il servizio.

C) Viene utilizzato un chip dedicato (PIC: Controllore d'interrupt programmabile) che gestisce tutta l'operazione in modo programmabile.

6) La routine di servizio specifica di ogni periferica provvede prima di tutto al salvataggio, e alla fine, al ripristino dei registri interni alla CPU. Svolge l'operazione di I/O, che può consistere, sia nel trasferimento di un solo carattere, sia di un intero blocco di dati.

N.B. Prima di cedere il controllo alla routine di servizio, la CPU DISABILITA l'interrupt. Pertanto, la routine di servizio può risultare ININTERROMPIBILE. Tale situazione può essere variata, riabilitando l'interrupt dall'interno della routine di servizio, tramite l'istruzione opportuna, in modo da poter essere eventualmente interrotta a sua volta da interrupt a priorità maggiori.

Questo spiega l'utilizzo dello stack, al posto di un’area di memoria di tipo ordinario, per il salvataggio del PC, dello SR e dei vari registri interni.

L'interrupt può essere interpretato come una chiamata ad un sottoprogramma, CONTROLLATA IN MODO HW o semplicemente in modo ASINCRONO ovvero in modo indipendente dalla sequenza degli eventi stabiliti dal programma in esecuzione.

7) Effettuata l'operazione, la CPU, ripristina il contenuto originario del PC (ed eventualmente de SR) e ritorna ad eseguire il programma interrotto, a partire dalla istruzione successiva a quella abbandonata prima dell'interruzione.

- INTERRUPT VETTORIZZATO

Questo sistema necessita di un segnale di risposta da parte della CPU che informa la periferica (le periferiche) che l'interrupt è stato accolto (segnale di INTERRUPT ACKNOWLEDGE).

Tale segnale attiva un circuito che informa la CPU, dell'indirizzo della routine che gestisce l'I/O della periferica che ha lanciato l'interrupt.

Il dispositivo che gestisce l'operazione acquisisce l'indirizzo consultando una tabella (VETTORE DI INTERRUPT), depositata in memoria, o all'interno del dispositivo stesso.

L'indirizzo viene immesso generalmente sul bus dati, pertanto, o è costituito solamente da 8 bit ($\Rightarrow$ la routine si trova nelle prime 256 celle di memoria) oppure, viene inviato alla CPU in due volte.

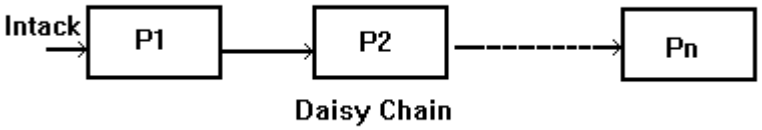
In alcuni casi, la parte più significativa (parte alta dell'indirizzo, viene precaricata una volta per tutte in un particolare registro d'interrupt interno alla CPU, tramite programma, nella fase di inizializzazione).

- CONTROLLO HARDWARE DELLA PRIORITÀ

Il controllo HW della priorità può essere realizzato in modo:

- CENTRALIZZATO Tramite l'utilizzo di un apposito chip specializzato che funziona da decodificatore della priorità (p.e. SN 74LS147).
- DISTRIBUITO In cui la priorità viene realizzata tramite la disabilitazione al lancio dell'interrupt da parte di un dispositivo a priorità maggiore sui dispositivi che lo seguono nell'ordine della priorità.

In pratica, un dispositivo che richiede l'interrupt, disabilita l'interruzione di tutti i dispositivi a priorità minore.



Daisy Chain

Se P1 ha richiesto l'interrupt, cattura per primo il segnale d'interrupt acknowledge, e invia alla CPU l'indirizzo della sua routine di servizio, e nello stesso tempo, non lascia passare il segnale ai dispositivi successivi che non ricevendolo saranno come disabilitati all'interrupt.

Un tipico sistema di questo tipo è detto "DAISY CHAIN".

- LOGICA D'INTERRUPT 8086

	Vettore d'interrupt	
Tipo	Locazione	Sorgente
0	00000h	Trap (Divisione per zero)
1	00004h	Trap (Passo Singolo)
2	00008h	NMI
3	0000Ch	Interrupt Software
4	00010h	Trap (Overflow)
5-33	00014h 0007Ch	Riservato a prodotti SW e HW della Intel
32-255	00080h 003FCh	Disponibili dall'utente

Come già anticipato, la logica d'interrupt si basa esclusivamente sul sistema vettorizzato. Il vettore d'interrupt è disposto dalla logica di inizializzazione nel primo KByte di memoria come specificato di seguito.

Ogni componente del vettore è composta da 4 byte; i primi due identificano l'indirizzo della cella ove risiede la relativa routine di servizio, mentre i secondi due identificano il codice del segmento. Sono quindi disponibili 256 componenti, ad ognuna delle quali corrisponde un particolare tipo d'interrupt.

P.e. locazione= 00080h (INT 32) $\Rightarrow$ Contenuto (4 Byte) = 65h, 82h, DDh, 8Ah.

Nel caso venga generato un interrupt di "tipo 32" (Ovvero quello relativo alla cella 00080h) la CPU carica i registri come segue:

CS = 8ADDh (Codice/indirizzo di Segmento)
IP = 8265h (Displacement)

Pertanto la CPU effettua un salto all'indirizzo:

```
8ADD0h +
  8265h =
----------
 93035h   (Indirizzo fisico della routine di servizio)
```

L'interrupt può essere generato in vari modi:

- Interrupt Non Mascherabile (NMI). In questo caso la CPU effettua un salto alla locazione di memoria specificata dalla componente n. 2 del vettore d'interrupt (Cella 00008h)

- TRAPs dovuti a occorrenze anomale del funzionamento della CPU, come "divisione per zero" (INT tipo 0), oppure a "Overflow" (INT tipo 3) oppure a modo di funzionamento in Single-Steep (INT tipo 1). Quest'ultima eventualità viene generata ad ogni singolo passo del programma, e serve per un eventuale debug.
- Interrupt di tipo software, causati da programma tramite l'istruzione INT il cui operando è il tipo d'interrupt richiesto. P.e. INT 4 ⇒ è simulato un overflow. Nel caso sia assente l'operando, è assunto per default l'interrupt di tipo 3 (INT software).
- Interrupt generato da una richiesta di I/O, per attivazione del pin di INTR (Interrupt Request) della CPU, come accade nel sistema ordinario.

Nel caso che l'interrupt sia abilitato, il tipo (e quindi la relativa componente del vettore) viene identificata dalla CPU tramite il codice della periferica che viene immesso sul bus dati da essa stessa, o da un chip ausiliario (PIC Intel 8259) che ne gestisce l'interrupt, in risposta al segnale di INTACK.

P.e. codice periferica = 56h ⇒ Interrupt tipo 56h ⇒ la componente del vettore si trova alla cella 158h.

L'abilitazione o la disabilitazione dell'interrupt è definita al solito dal flag "IF" del registro di stato.

È possibile settare (STI) o resettare (CLI) il flag d'interrupt dell’8086 tramite le due apposite istruzioni. Tali istruzioni possono essere utilizzate quindi per abilitare (STI) o disabilitare (CLI) l’interrupt.

Un’altra istruzione necessaria per il buon funzionamento dell’interrupt è la RETI (return from interrupt) che serve per terminare la routine di servizio della periferica. In pratica, quest'istruzione riprende il contenuto dello stack e lo ripone nei registri di programma (CS e IP) in modo da continuare l’esecuzione del programma dal punto interrotto. L’istruzione RETI si differenzia dalla RET ordinaria solamente per il codice sorgente, in quanto la traduzione in linguaggio macchina è esattamente la stessa (C3h).

- DMA

Questa tecnica consiste nel permettere il passaggio diretto del dato dalla memoria al dispositivo di I/O (o viceversa) senza passare da registri intermedi o altri supporti. Per la sua realizzazione è necessario un dispositivo dedicato detto "DMAC" il quale provvede a svolgere l'operazione in modo autonomo, anzi, quando esso opera, la CPU è inattiva ovvero è in uno stato detto di "WAIT" nel quale le sue uscite sono in "ALTA IMPEDENZA". Questo stato della CPU ha lo scopo di lasciare la completa disponibilità del BUS al DMAC in modo che esso possa operare al suo posto. D'altra parte, la CPU, non può svolgere operazioni di DMA in quanto la sua architettura a "BUS UNICO" non lo permette. Infatti, per poter utilizzare questa tecnica è necessario indirizzare contemporaneamente sia la memoria che il dispositivo di I/O. Invece, il DMAC, dispone, oltre che del controllo del bus indirizzi, anche di CANALI DEDICATI, ognuno associato ad una porta di I/O.

L'utilizzo del DMAC, permette di svolgere operazioni di I/O in modo estremamente veloce, grazie al fatto che esso dimezza il tempo per l'operazione facendo il trasferimento diretto. L'efficienza del DMAC è ulteriormente giustificata anche dal fatto che il trasferimento avviene "A BLOCCHI", mentre con le altre tecniche si sposta solo un byte per ogni singola sessione. Si deve tenere conto anche che le operazioni svolte dal DMAC sono dettate da una logica di tipo "CABLATO" e che quindi sono di per se più veloci in quanto non è necessario interpretare alcun programma.

1 Spiegare la tecnica di multiplexing del bus nel sistema 8086.

I/O

2 Spiegare come avviene la gestione fisica della memoria nel sistema 8086.

I/O

3 Spiegare quali sono i segnali relativi alla CPU 8086

I/O

4 Spiegare le differenze e il funzionamento del modo MAX e MIN nell'8086.

I/O

5 Spiegare com'è organizzato il circuito di reset automatico in un sistema a microprocessore.

I/O

6 Spiegare la funzione del sistema di reset nel microprocessore

I/O

7 Spiegare il funzionamento del sistema di reset manuale e quali sono le relative problematiche

I/O

8 Spiegare com'è organizzata dal punto di vista temporale la relazione fra i segnali della CPU 8086.

I/O

9 Dire quali sono le problematiche che devono essere risolte tramite le tecniche d'interfacciamento ed in particolare quelle relative ai sistemi a microprocessore.

I/O

10 Dire quali sono le parti che compongono una interfaccia.

I/O

11 Definire quali sono le problematiche che possono essere risolte tramite le tecniche di collegamento al bus. Dire anche quali sono le differenze fra le due tecniche viste.

I/O

12 Utilizzando il sistema "Memory Mapped" collegare al bus due disply a 7 segmenti in modo tale che l'indirizzo associato sia 9A6BFh.

I/O

13 Utilizzando il sistema "isolated I/O" collegare al bus due disply a 7 segmenti in modo tale che l'indirizzo associato sia F4F7h.

I/O

14 Dire come può essere interfacciata una tastierina esadecimale con un sistema a microprocessore.

I/O

15 Dire quali sono i segnali e le istruzioni utilizzati nella tecnica Isolated I/O nel sistema 8086 con modo minimo.

I/O

16 Dire quali sono i segnali e le istruzioni utilizzati nella tecnica Isolated I/O nel sistema 8086 con modo massimo.

I/O

17 Spiegare quali problematiche sono connesse alle tecniche di Handshacking.

I/O

18 Spiegare quali sono e le relative problematiche connesse alle tecniche di scheduling.

I/O

19 Dire com'è organizzata la tecnica di Polling e come essa risolve i relativi problemi

I/O

20 Spiegare il funzionamento della tecnica di Interrupt e come essa risolve i relativi problemi.

I/O

21 Spiegare qual è la sequenza d'interruzione in un microprocessore

I/O

22 Spiegare come nel sistema d'interrupt è utilizzato il circuito di “Daisy Chain” e come è organizzata la vettorizzazione.

I/O

23 Spiegare come è organizzata la tecnica d'interrupt nel sistema 8086

I/O

24 Specificare l’indirizzo di memoria dove si trova la componente dell’interrupt n. 28h nel sistema 8086 e calcolare l’indirizzo fisico della routine sapendo che nella componente c’è il valore 06 4C 7B AA.

I/O

25 Specificare quali sono le possibilità di lanciare un interrupt nel sistema 8086 e quali sono le istruzioni per la relativa gestione.

I/O

26 Spiegare il sistema di DMA e perché risulta più efficiente.

I/O

27

I/O

28

I/O