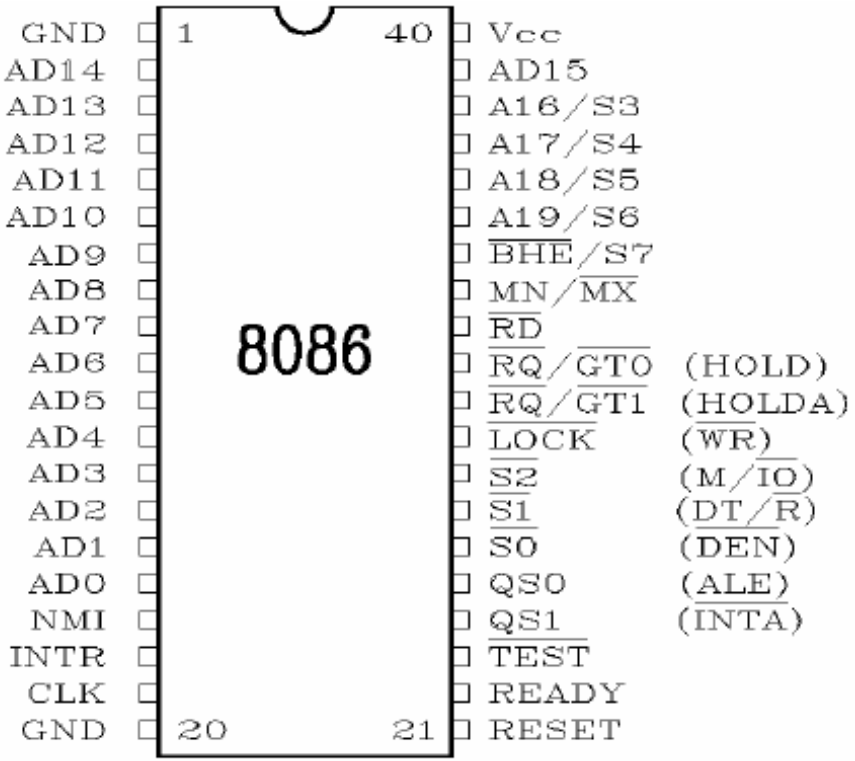
ARCHITETTURA DEL MICROPROCESSORE INTEL 8086 (iAPX86/10)

- Chip con 40 piedini e 29000 transistori
  - Progettato a metà degli anni ’70, periodo in cui la densità di integrazione era relativamente bassa (“solo 29000 transistori”) e non esistevano integrati commerciali con più di 40 piedini
  - I progettisti hanno dovuto bilanciare differenti esigenze: obiettivi e costi (con il vincolo della tecnologia)

- Conseguenze:
  - Numero limitato di registri di CPU
  - Registri e percorsi interni a 16 bit
  - Indirizzo fisico a 20 bit solo in uscita dalla CPU
  - Segmentazione della memoria
  - Bus multiplexed (dati e indirizzi)
    Bus dati e indirizzi su linee condivise (AD15÷AD0) con estensione (AD19÷AD16)
  - Previsione di componenti di corredo (coprocessore)

- F = 5 MHz
  - Tempo medio per istruzione 15 * 20ns = 3 μs



ARCHITETTURA DEL MICROPROCESSORE INTEL 8086 (iAPX86/10)

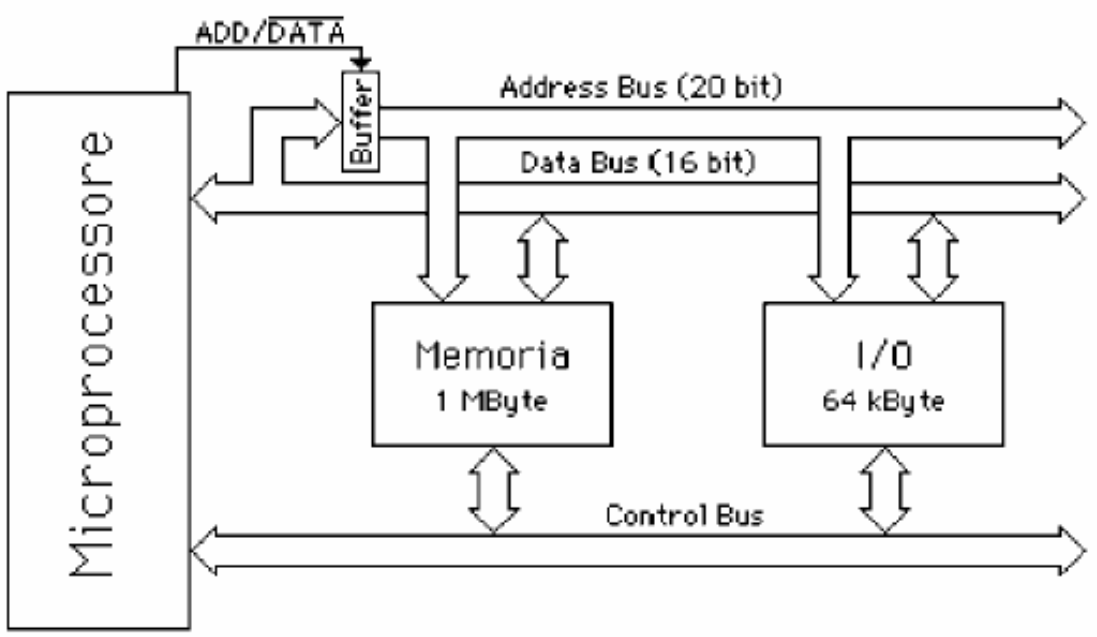
- **Bus Indirizzi**: 20 bit *1 MB di memoria indirizzabile*
  - 2^{20} = 1.048.576 locazioni di memoria di 8 bit
  - primo byte ha indirizzo 0, ultimo byte ha indirizzo FFFFF*H*

- **Bus Dati**: 16 bit per l'8086 (differente dall'8088 che era 8 bit)
  - Dati e istruzioni

- 16 bit ALU e operazioni aritmetiche e logiche

- 14 registri interni da 16 bit

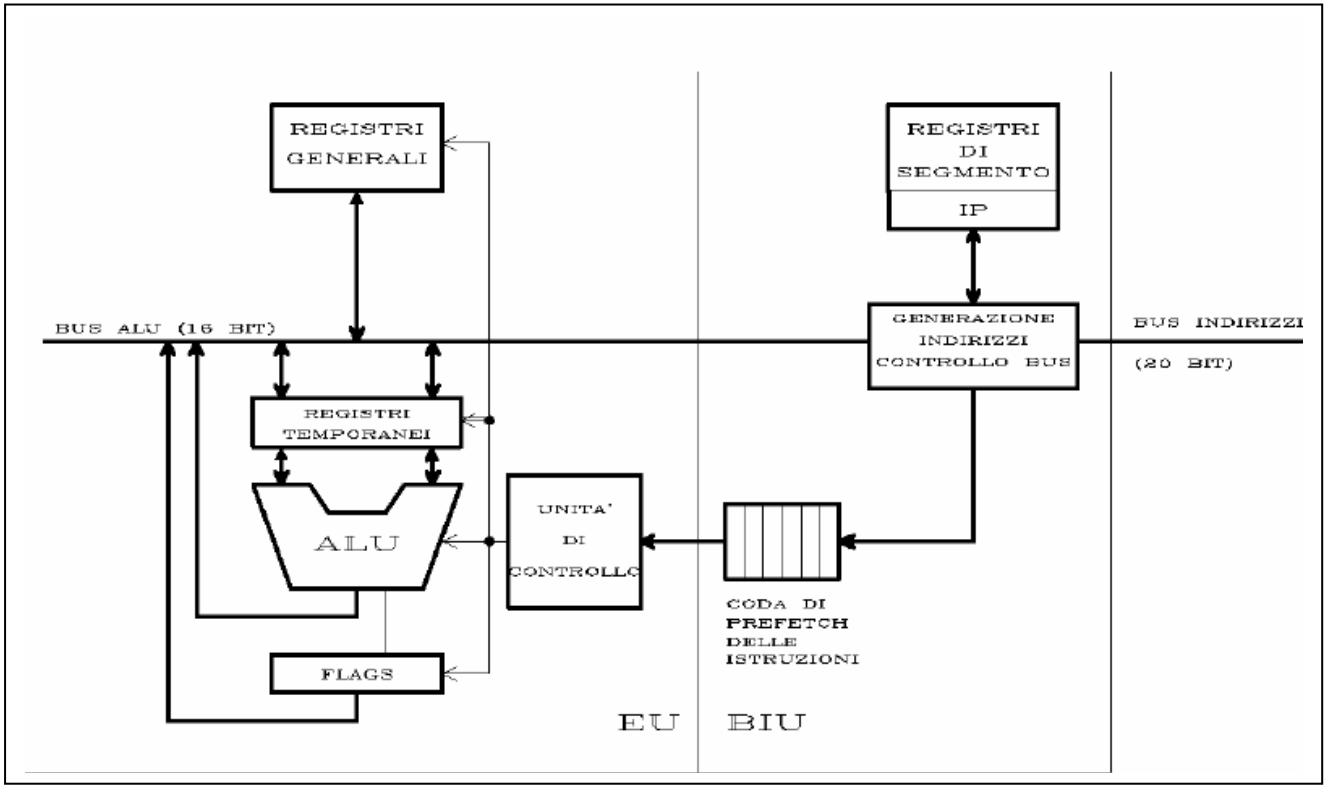
- **Memoria segmentata**



Il microprocessore o CPU

La CPU è costituita da due blocchi funzionali:

- **il Bus Interface Unit (BIU)**: esegue tutte le richieste dell'EU che coinvolgono il mondo esterno (e quindi il bus di sistema), cioè i trasferimenti di dati tra la CPU e la memoria o i dispositivi di I/O
  - accede alla memoria per istruzioni (fase di fetch) e dati (operand assembly)
  - controllo logico del BUS
  - scrive i risultati
  - calcola gli indirizzi fisico a 20 bit
- **l'Execution Unit (EU)**: costituita fondamentalmente da **ALU**, **registri** e **bus interno** a 16 bit; essa non ha connessioni dirette con il bus di sistema (cioè, con il mondo esterno).
  - **esegue le istruzioni (fase di execute)**
  - **fornisce dati e indirizzi al BIU**
  - **modifica registri generali e registro flag**



SEGMENTAZIONE DELLA MEMORIA

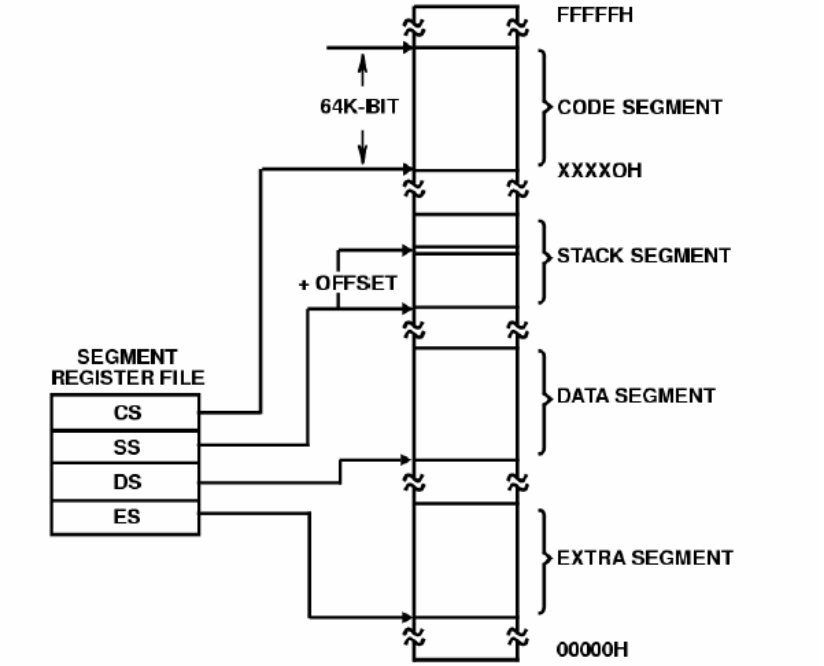
Lo spazio di memoria viene visto come un gruppo di segmenti

- ogni segmento è una unità logica di memoria indipendente, indirizzabile separatamente, che inizia con un indirizzo multiplo di 16
- ogni segmento è al massimo di 64 KByte

Ogni programma in esecuzione può accedere direttamente a

- 64 KByte di codice – CS
- 64 KByte di stack – SS
- 64 + 64 KByte di dati – DS e ES

Per accedere a codice o dati contenuti in altre zone è opportuno modificare i registri segmento in modo opportuno.



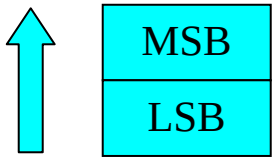
SEGMENTAZIONE DELLA MEMORIA

I segmenti possono essere
- Contigui
- Disgiunti
- Sovrapposti parzialmente
- Sovrapposti totalmente

NB. Una locazione di memoria può essere contenuta in più segmenti.

In tutta la famiglia di processori INTEL l'ordinamento dei dati in memoria è del tipo LITTLE ENDIAN
- Un numero di 16 bit viene memorizzato in ordine inverso

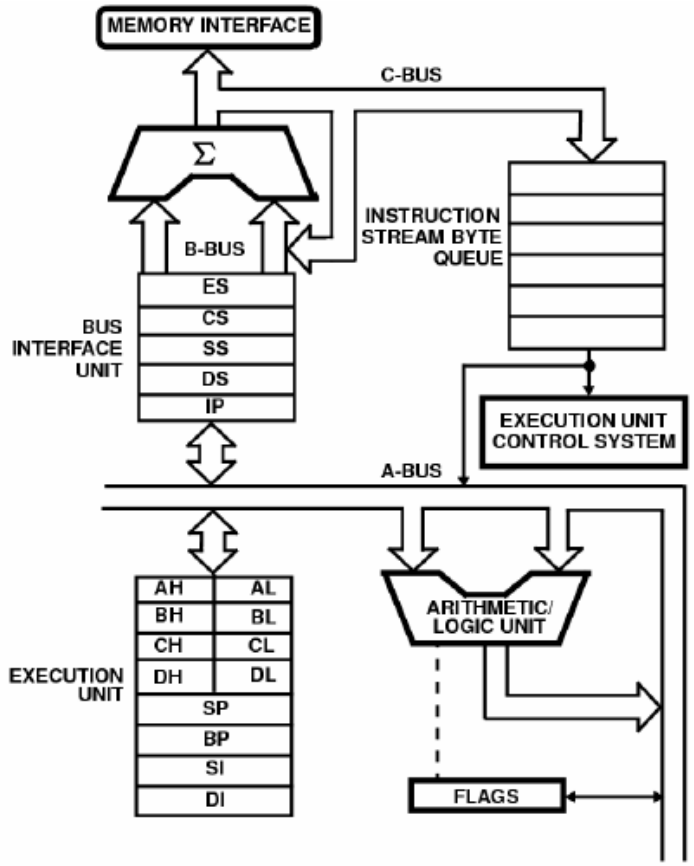
Es. ABCD => CDAB



Registri interni

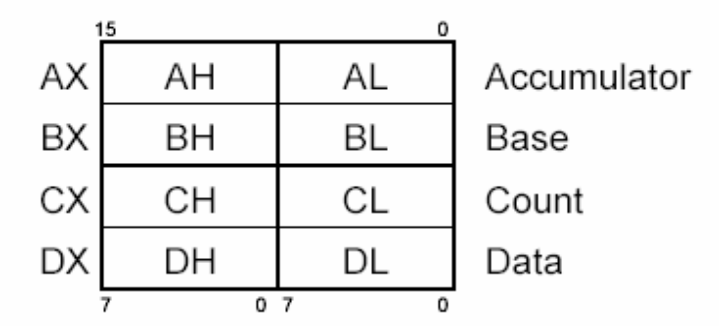
Il microprocessore 8086 contiene al suo interno 3 set di 4 registri, l'istruction pointer IP e il registro dei flag, da 16 bit ciascuno.

- Alta velocità di memorizzazione degli elementi.
- **Mancanza di Ortogonalità** cioè della possibilità per le istruzioni di utilizzare uno qualsiasi dei registri come operando.
  - istruzioni che utilizzano alcuni i registri generali in modo implicito, ad esempio: le istruzioni che agiscono sullo stack coinvolgono sempre, in modo implicito, il registro SP
  - istruzioni che funzionano solo con particolari registri generali



Data register:

- Possono essere anche visti come 8 registri da 8 bit



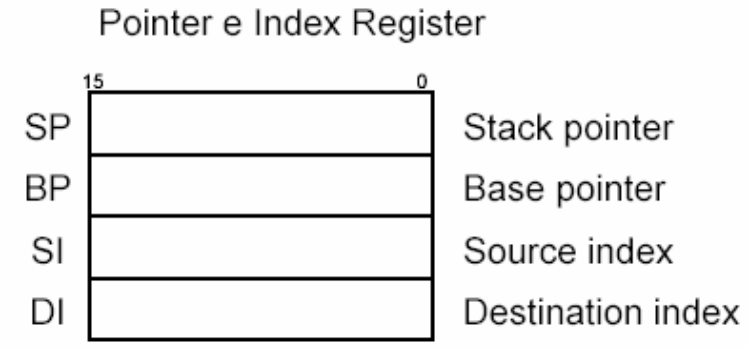
NB: I data register sono utilizzabili indifferentemente per tutte le operazioni aritmetico-logiche, a eccezione di alcune istruzioni di manipolazione delle stringhe

Il registro AX viene detto anche **registro accumulatore**

- registro privilegiato per certe istruzioni, per esempio *ADD AX, VAR* effettua una somma del valore di VAR e lo somma al valore di AX mettendo, accumulando il risultato in AX stesso

$$AX \leftarrow AX + M[DS:VAR]$$

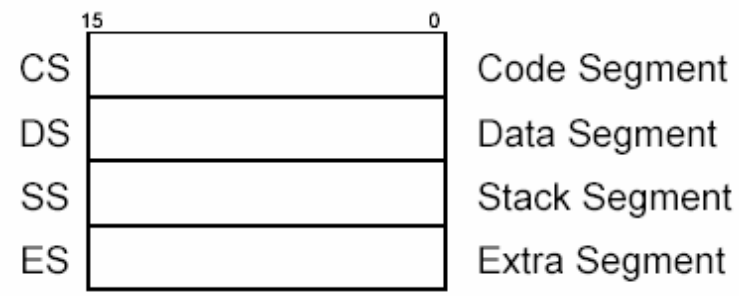
Index e Pointer register



NB: I registri Pointer assumono per definizione come segmento di riferimento quello di STACK, mentre i registri Index sono utilizzati all'interno del segmento DATA, salvo per alcune istruzione di manipolazione di stringhe.

Registri di Segmento

La CPU può accedere direttamente a 4 segmenti per volta (massimo 256k byte).
Quattro registri di segmento puntano ai quattro segmenti correntemente attivi:



- **CS (Code Segment)** punta al segmento codice corrente: il segmento da cui vengono ottenute le istruzioni da eseguire.
- **SS (Stack Segment)** punta al segmento contenente lo stack corrente.
- **DS (Data Segment)** punta al segmento dati corrente: in genere tale segmento contiene variabili di programma.
- **ES (Extra Segment)** punta al segmento extra corrente: in genere anche questo segmento viene utilizzato per memorizzare dati.

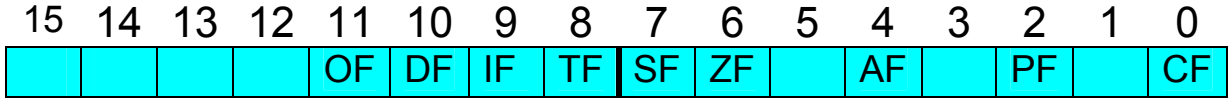
Instruction Pointer

Contiene, in ogni istante, l'offset (distanza in byte) dell'istruzione successiva dall'inizio del segmento codice corrente (CS). Il program counter classico coincide con <**CS:IP**>.

I programmi non hanno accesso diretto all'IP, ma le istruzioni lo modificano implicitamente

Registro Flag

Tale registro contiene 5 flag di stato: OF, SF, ZF, AF, PF e CF; e 3 di controllo: DF, IF e TF



- ***OF (overflow flag)*** vale 1 quando c'è overflow (è significativo solo in presenza di operazioni aritmetiche).
- ***SF (sign flag)*** viene posto a 1 quando il risultato di una operazione sulla ALU è un numero negativo.
- ***ZF (zero flag)*** viene posto a 1 quando il risultato di un'operazione aritmetica è zero.
- ***AF (auxiliary flag)*** viene posto ad 1 in presenza di riporto nelle operazioni in codifica BCD.
- ***PF (parità flag)*** vale 1 se il byte meno significativo del risultato contiene un numero pari di 1.
- ***CF (carry flag)*** viene posto a 1 se
  - nelle somme c'è un riporto sul bit più significativo
  - nelle sottrazioni per il bit più significativo è necessario un prestito.

- ***DF (direction flag)*** assume significato solo nelle operazioni con stringhe dove identifica l'organizzazione di queste in memoria
- ***IF (interrupt flag)*** controlla l'abilitazione delle interruzioni
- ***TF (trap flag)*** pone il processore nello stato "single step" nel debbugging

I flag di stato vengono modificati dall'EU in base al risultato delle operazioni logiche e aritmetiche.

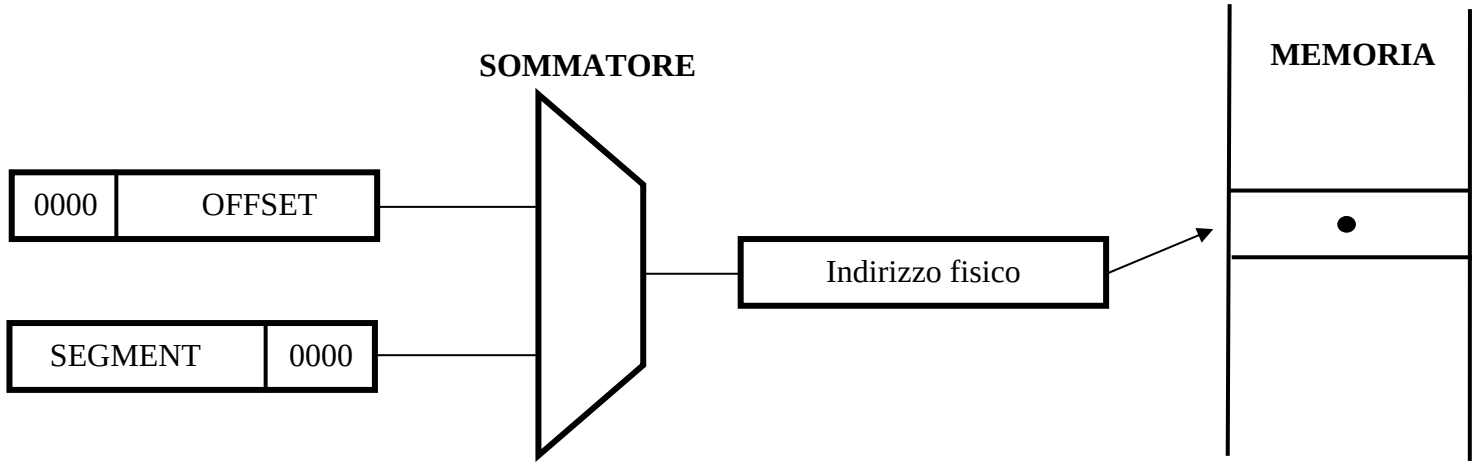
Esiste un gruppo di istruzioni che permette al programma di controllare il contenuto di tali flag a fini decisionali.

I flag di controllo possono essere settati o azzerati dal programma al fine di modificare il comportamento della CPU.

Generazione dell'indirizzo fisico

Un indirizzo fisico è un valore di 20 bit che identifica in modo univoco una locazione della memoria di 1M byte.

Il **BIU** converte la coppia **<segmento:offset>,** entrambi quantità di 16 bit senza segno, in indirizzo fisico.



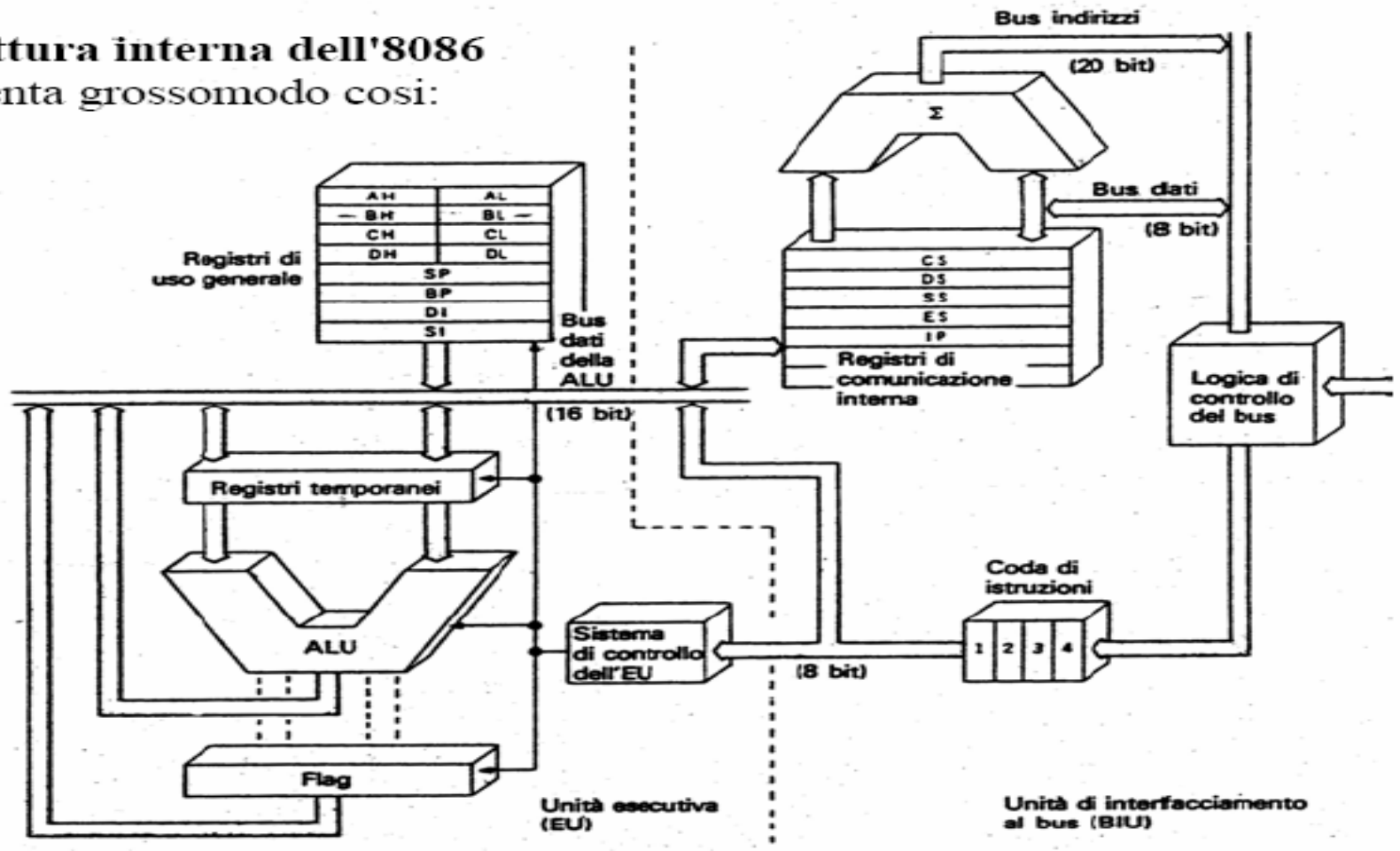
INDIRIZZO FISICO = SEGMENT * 16 + IP

NB: la moltiplicazione per 16 può essere notevolmente velocizzata da un semplice shift a sinistra di 4 posizioni della rappresentazione binaria del numero.

- Le istruzioni da eseguire vengono ricavate dal segmento codice corrente, pertanto l'indirizzo fisico della successiva istruzione è dato da: <CS:IP>, cioè CS * 16 + IP
- Le istruzioni che agiscono sullo stack utilizzano il segmento stack corrente:
  - SS contiene l'indirizzo del segmento
  - SP contiene l'offset del top dello stack
- Gli operandi che fanno riferimento alla memoria (variabili di programma) di norma risiedono nel data segment corrente (DS) però, il programma può dire al BIU di utilizzare uno qualunque dei quattro segmenti correntemente disponibili. L'offset della variabile viene invece calcolato dall'EU e dipende dalla modalità di indirizzamento specificata nell'istruzione.

Struttura del processore 8086

La **struttura interna dell'8086** si presenta grossomodo cosi:

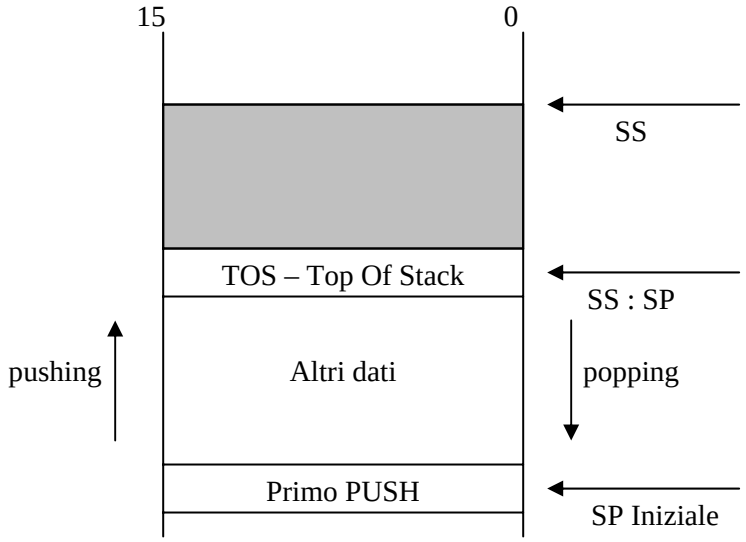


Stack

Lo stack è identificato mediante i registri SS (Stack Segment) e SP (Stack Pointer). Mentre SS contiene la base dello stack corrente, SP contiene il puntatore alla cima, cioè contiene lo scostamento della cima dello stack rispetto a SS.
In memoria possono coesistere più stack, ognuno al massimo di 64Kbyte, ma uno solo è quello corrente

- L'8086 ha istruzioni per gestirlo
- Serve al processore in occasione di interrupts (salvataggio dello stato)
- Le chiamate a *Procedure* usano lo stack per il successivo ripristino degli indirizzi
- E' conveniente averne uno da usare come memoria temporanea

N.B.: SP indica la cima dello stack ma SS non è il fondo dello stack



Il processore non verifica eventuali condizioni illegali

- Il Programmatore potrebbe includere il codice per la verifica di errori nell'uso dello stack
- Si ha Overflow quando SP ha un valore inferiore dell'indirizzo di inizio dello stack array => SP è negativo. Si ha Underflow se SP diventa più grande del suo valore iniziale

Interrupt

Un interrupt (interruzione) è un evento che si verifica in momenti non prevedibili. L'effetto è quello di trasferire il controllo a una nuova locazione di memoria, in cui è collocata una procedura, detta **routine di servizio dell'interrupt** (ISR – Interrupt Service Routine). Al termine della procedura, si rientra nel programma principale.

Gli interrupt possono essere:

- **Hardware**
  - mascherabili
  - non mascherabili

- **Software**: hanno origine dall'esecuzione del programma
  - direttamente (per es., l'esecuzione di un'istruzione INT)
  - indirettamente - condizioni eccezionali (per es., una divisione per zero)

Dal punto di vista di ciò che avviene a livello hardware:

- viene eseguita un push dei registri flags, CS e IP per salvare la situazione corrente e poterla ripristinare al termine del servizio
- vengono caricati i nuovi valori di CS e IP dalla tabella degli interrupt
- vengono azzerati i flag TF (trap per single step) e IF (interrupt)

Gli interrupt nell'8086

Le locazioni da 0H a 3FFH contengono una tabella (Interrupt Vector Table) con 256 ingressi. Ogni ingresso contiene due valori di 16 bit che forniscono l'indirizzo della routine di servizio dell'interrupt e che vengono caricati nei registri CS e IP quando l'interrupt viene accettato.
I primi cinque elementi della tabella sono dedicati a particolari tipi di interrupt predefiniti nell'8086. I successivi 27 elementi sono riservati all'hardware del sistema di elaborazione e non devono essere utilizzati. I rimanenti elementi (da 32 a 255) sono disponibili per le routine di servizio e del sistema operativo dell'utente.

Un programma può anche generare esplicitamente un interrupt di tipo n, mediante l'istruzione **INT n**

- **Interrupt 0 (Divide Error)** - segnala un errore durante un operazione di divisione (ad es., divisione per zero).
- **Interrupt 1 (Single Step)** - un'istruzione dopo il settaggio di TF (permette di eseguire una singola istruzione all'interno di un programma - utilizzato dal debugger)
- **Interrupt 2 (Non-Maskable Interrupt)** - è l'interrupt hardware di priorità più alta e non è mascherabile – di norma, è riservato ad eventi importanti e urgenti (ad es., una caduta di tensione, un errore nella memoria, un errore sul bus di sistema)
- **Interrupt 3 (One Byte Interrupt)** - utilizzato dal debugger per i breakpoint
- **Interrupt 4 (Interrupt on Overflow)** - condizione di overflow (OF = 1) e viene eseguita l'istruzione INTO; permette di gestire l'eventuale condizione di overflow

L'azzeramento di IF disabilita il riconoscimento di ulteriori interrupt hardware nella routine di servizio a meno che tale riconoscimento non venga riabilitato esplicitamente all'interno della routine di servizio stessa.
La routine di servizio deve terminare con un'istruzione **IRET (Interrupt RETurn)**, al fine di ripristinare correttamente la situazione presente al momento in cui si è verificata l'interruzione.