

2.1 Registri del modello x86

Registri del modello x86

Il modello di programmazione x86 può differire notevolmente in funzione dell'effettivo processore a cui ci si riferisce e del livello in cui ci si pone (programmazione applicativa a 16 bit, programmazione applicativa a 32 bit, programmazione di sistema)

Questo modello si riferisce ad un processore dal 80386 per un sistema operativo a 32 bit.

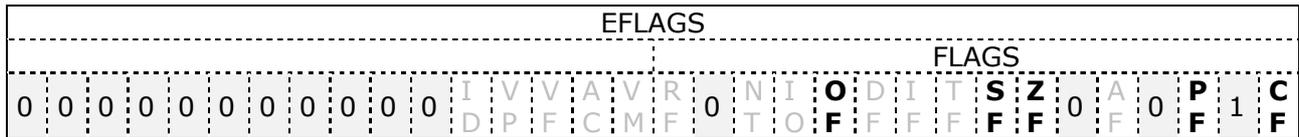
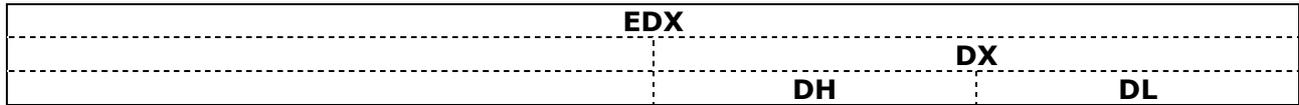
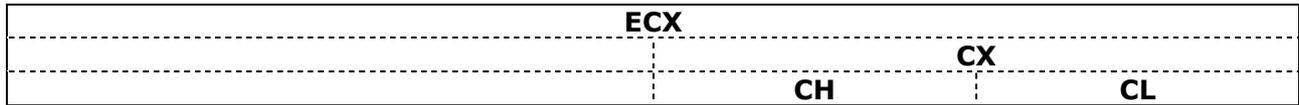
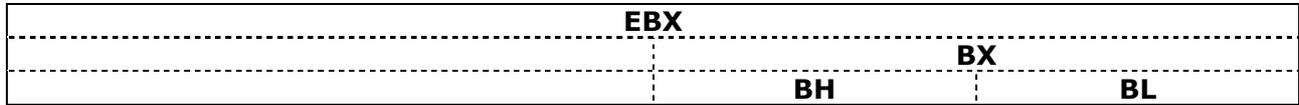
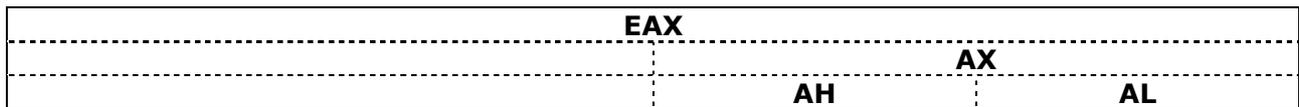
Questa scelta è congruente con l'ambiente di sviluppo usato (Borland C++ Builder) che sviluppa applicazioni per la piattaforma Win32.

Dal punto di vista operativo corrisponde all'impostazione .386 e model flat che usiamo per generare file assembly utilizzabili in ambiente Borland.

Il modello di programmazione si compone dei seguenti elementi:

- Registri interni. I registri per uso generale (general purpose) e i registri specializzati forma l'ambiente di esecuzione delle istruzioni. Queste istruzioni possono fare trasferimenti, operazioni aritmetico/logiche e controllo del flusso del programma.
- Spazio di indirizzamento della memoria. Ogni programma in esecuzione con questo modello può indirizzare uno spazio fino a 4 Gbytes di memoria (2^{32} bytes).
- Spazio di indirizzamento degli I/O. L'accesso agli I/O è una operazione privilegiata e sarà trattata in un apposito capitolo

Banco dei registri



- CS
- DS
- SS
- ES
- FS
- GS

Note:

- I registri EAX, EBX, ECX, EDX, ESI, EDI, EBP sono registri "general purpose" a 32 bit. Servono come sorgente o destinazione per qualunque operazione di trasferimento e aritmetico logica.
- I 16 bit meno significativi di EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP, EIP, EFLAGS possono essere visti come i registri a 16 bit AX, BX, CX, DX, SI, DI, BP, SP, IP e FLAGS.
- Gli 8 bit meno significativi di AX, BX, CX, DX possono essere visti come i registri ad 8 bit AL, BL, CL, DL.
- Gli 8 bit più significativi di AX, BX, CX, DX possono essere visti come i registri ad 8 bit AH, BH, CH, DH.
- Il registro EIP è un registro specializzato di tipo puntatore che contiene sempre l'indirizzo della prossima istruzione da eseguire. Sebbene il programmatore non abbia a disposizione alcuna istruzione per modificare esplicitamente EIP tutte le istruzioni lo modificano implicitamente.
- Il registro ESP è un registro specializzato di tipo puntatore che contiene l'indirizzo della cima della memoria a catasta (stack). Questa memoria che è una porzione dello spazio di memoria indirizzabile è indispensabile per la gestione dei sottoprogrammi, del passaggio dei parametri e delle variabili locali.
- Il registro EFLAGS contiene i bit di flag che vengono modificati dalla ALU in base al risultato delle operazioni aritmetico/logiche. I bit non usati dal processore (riservati per usi futuri) sono indicati con sfondo grigio. I bit che abbiamo studiato sono indicati in grassetto, mentre quelli che non abbiamo studiato sono indicati con testo grigio.
- I registri a 16 bit CS, DS, SS, ES, FS, GS, sebbene visibili nel nostro modello non sono usati perché servono per il modello di programmazione segmentata.

Il modello descritto è fedelmente rappresentato dalla finestra CPU dell'ambiente di debug dell'ambiente di sviluppo Borland Builder.

The screenshot shows the CPU window of a debugger. The main pane displays assembly code for a thread at address 0x00401012. The code includes instructions like 'jmp', 'bound', 'inc', 'sub', 'dec', 'nop', 'mov', 'shl', 'push', 'call', and 'pop'. The right pane shows the state of various registers: EAX, EBX, ECX, EDX, ESI, EDI, EBP, ESP, EIP, EFL, CS, DS, SS, ES, FS, GS, CF, PF, ZF, SF, TF, IF, DF, OF, IO, NF, RF, VM, AC, VF, VP, and ID. The EIP register is highlighted with a mouse cursor, showing the value 00401012. Below the assembly code, a memory dump shows the contents of memory addresses from 00410000 to 00410020, mostly containing zeros.

Address	Instruction	EAX	CF
00401000	EB10 jmp 0x401012	00000000	0
00401002	66623A bound di, [edx]	EBX 7FFDF000	PF 1
00401005	43 inc ebx	ECX 0012FFB0	AF 0
00401006	2B2B sub ebp, [ebx]	EDX 7FFE0304	ZF 0
00401008	48 dec eax	ESI 00000000	SF 1
00401009	4F dec edi	EDI 00000000	TF 0
0040100A	4F dec edi	EBP 0012FFF0	IF 1
0040100B	4B dec ebx	ESP 0012FFC4	DF 0
0040100C	90 nop	EIP 00401000	OF 0
0040100D	E99C524000 jmp +0x0040529c	EFL 00000286	IO 0
00401012	A18F524000 mov eax, [0x40528F]	CS 001B	NF 0
00401017	C1E002 shl eax, 0x02	DS 0023	RF 0
0040101A	A393524000 mov [0x405293], eax	SS 0023	VM 0
0040101F	52 push edx	ES 0023	AC 0
00401020	6A00 push 0x00	FS 003B	VF 0
00401022	E8253E0000 call KERNEL32.GetModu	GS 0000	VP 0
00401027	8BD0 mov edx, eax	ID 0	
00401029	E8D6250000 call __CRTL_VCL_Init		
0040102E	5A pop edx		
0040102F	E8AE3E0000 call CC3250MT.__CRTL		
00401034	E80F260000 call __CRTL_VCLLIB_L		
00401039	6A00 push 0x00		
0040103B	E83C270000 call __ExceptInit		
00401040	59 pop ecx		
00401041	6838524000 push 0x00405238		

Uso specializzato dei registri "general purpose"

Sebbene i registri "general purpose" possano essere usati per qualsiasi scopo, esistono delle istruzioni che li usano in modo specializzato rendendo molto efficiente il loro utilizzo. La sigla del registro spesso deriva proprio da questo uso specializzato.

- EAX: Accumulatore dei risultati; sebbene qualsiasi registro general purpose possa essere usato per questo scopo è preferibile usare EAX (AX, AL) per accumulare risultati di operazioni aritmetico/logiche.
- EBX: Puntatore alla memoria dei dati; anche in questo caso qualunque registro general purpose a 32 bit è in grado di puntare ad una cella di memoria ma EBX è la scelta preferenziale
- ECX: Contatore di iterazioni; qualunque registro può essere usato come contatore in un algoritmo iterativo ma esistono istruzioni specializzate che usano ECX come contatore
- DX: Puntatore alle porte di I/O: le istruzioni di I/O sono fortemente specializzate e solo il registro a 16 bit DX può essere usato come puntatore ad una porta di I/O (spazio di I/O =64K porte)
- ESI: Puntatore alla memoria dei dati sorgente nelle operazioni di copia di stringa
- EDI: Puntatore alla memoria dei dati destinazione nelle operazioni di copia di stringa
- EBP: Puntatore ai parametri del sottoprogramma nello stack