

Dr. Máté Eörs
docens

Képfeldolgozás és Számítógépes Grafika Tanszék
Árpád tér 2. II. em. 213
6196, 54-6196
(6396, 54-6396)

<http://www.inf.u-szeged.hu/~mate>

mate@inf.u-szeged.hu

Máté: Assembly programozás

1. előadás

1

Tantárgy leírás:

<http://www.inf.u-szeged.hu/oktatas/kurzusleirasok/IB676.xml>

Elérhető a honlapomról a

Követelmények, tematika

link-en

Máté: Assembly programozás

1. előadás

2



Hétfőn és pénteken
15 és 16 óra között a
Tanszékcsoporti
Irodában
(Árpád t. 2. I. em)
800 Ft (késspénztér)
beszerezhető

Máté: Assembly programozás

1. előadás

3

Gépi, nyelvi szintek

5. Probléma orientált nyelv szintje
fordítás (fordító program)
4. **Assembly nyelv szintje**
fordítás (assembler)
3. Operációs rendszer szintje
részben értelmezés (operációs rendszer)
2. Gépi utasítás szintje
ha van mikroprogram, akkor értelmezés
1. Mikroarchitektúra szintje
hardver
0. Digitális logika szintje

Máté: Assembly programozás

1. előadás

4

Pentium 4. (T1.11. ábra)

Lapka	Dátum	MHz	Tranz.	Mem.	Megjegyzés
I-4004	1971/4	0.108	2300	640	Első egylapkás mikroproc.
I-8008	1972/4	0.108	3500	16 KB	Első 8 bites mikroroc.
I-8080	1974/4	2	6000	64 KB	Első általános célú mikroproc.
I-8086	1978/6	5-10	29000	1 MB	Első 16 bites mikroroc.
I-8088	1979/6	5-8	29000	1 MB	Az IBM PC pocesszora
I-80286	1982/6	8-12	134000	16 MB	Memória védelem
I-80386	1985/10	16-33	275000	4 GB	Első 32 bites mikroproc.
I-80486	1989/4	25-100	1.2M	4 GB	8 KB beépített gyorsítótár
Pentium	1993/5	60-233	3.1M	4 GB	Két csővezeték, MMX
P. Pro	1995/3	150-200	5.5M	4 GB	Két szintű beépített gyorsítótár
P. II	1997/5	233-400	7.5M	4 GB	Pentium Pro + MMX
P. III	1999/2	650-1400	9.5M	4 GB	SSE utasítások 3D grafikához
P. 4	2000/11	1300-3800	42M	4 GB	Hyperthreading + több SSE

Máté: Assembly programozás

1. előadás

Nem kell

5

Pentium 4

Nagyon sok előd (kompatibilitás!), a fontosabbak:

- **4004**: 4 bites,
- **8080**: 8 bites,
- **8086, 8088**: 16 bites, 8 bites adat sín.
- **80286**: 24 bites (nem lineáris) címtartomány (16 K darab 64 KB-os szegmens).
- **80386**: **IA-32** architektúra, az Intel első 32 bites gépe, lényegében az összes későbbi is ezt használja.
- **Pentium II** –től **MMX** utasítások.

Máté: Assembly programozás

1. előadás

Nem kell

6

A Pentium 4 üzemmódjai

real (valós): az összes **8088** utáni fejlesztést kikapcsolja (valódi **8088**-ként viselkedik).
Hibánál a gép egyszerűen összeomlik, lefagy.

virtuális 8086: a **8088**-as programok védett módban futnak (ha **WINDOWS**-ból indítjuk az **MS-DOS**-t, és ha abban hiba történik, akkor nem fagy le, hanem visszaadja a vezérlést a **WINDOWS**-nak).

védett: valódi **Pentium 4**-ként működik.

Ilyenkor 4 védelmi szint lehetséges (**PSW**):

0: kernelmód (operációs r.), **1, 2**: ritkán használt, **3**: felhasználói mód.

Máté: Assembly programozás

1. előadás

7

Általános regiszterek (32, 16 illetve 8 bitesek)

dword	word	higher byte	lower byte	
EAX	AX	AH	AL	Accumulátor (szorzás, osztás is)
EBX	BX	BH	BL	Base Register (címző)
ECX	CX	CH	CL	Counter Register (számláló)
EDX	DX	DH	DL	Data Register (szorzás, osztás, I/O)

80386-től

AH	AL
AX	
EAX	

Máté: Assembly programozás

1. előadás

8

A 8088-80286-os processzorokon a további regiszterek is 16 bitesek voltak, a 80386-os processzorral kezdődően 32 bitesre egészítették ki 16 magasabb helyértékű bittel, az így kapott regiszterek elnevezése előtt egy **E** betűvel egészült ki.

A 80386-os processzorokon az általános és index regiszterek 16 bites alacsonyabb helyértékű része az eredeti névvel, a teljes regiszter az **E**-vel kiegészített névvel érhető el.

A 80386-os és a Pentium processzorok tudnak 8086/8088-asként is működni (l. később).

Máté: Assembly programozás

1. előadás

9

Vezérlő regiszterek (32, eredetileg 16 bitesek)

- **ESI** (Source Index) a forrás adat terület indexelt címzéséhez használatos (**SI** az **ESI** 16 bites része)
- **EDI** (Destination Index) a cél adat terület indexelt címzéséhez használatos (**DI** az **EDI** 16 bites része)
- **EBP** (Base Pointer) a stack indexelt címzéséhez használatos
- **ESP** (Stack Pointer) verem mutató: a stack-be (verembe) utolsónak beírt elem címét (80286-ig **SP** az **SS** által mutatott szegmensbeli relatív címet) tartalmazza
- **EIP** (Instruction Pointer) utasítás számláló: az éppen végrehajtandó utasítás logikai címét (80286-ig **IP** az **CS** által mutatott szegmensbeli relatív címet) tartalmazza
- **EFLAGS (PSW)** a processzor állapotát jelző regiszter

Máté: Assembly programozás

1. előadás

10

Szegmens regiszterek (16 ill. 32 bitesek)

A szegmens regiszterek bevezetésének eredeti célja az volt, hogy nagyobb memóriát lehessen elérni.

- **CS** (Code Segment) utasítások címzéséhez
- **SS** (Stack Segment) verem címzéséhez
- **DS** (Data Segment) (automatikus) adat terület címzéséhez
- **ES** (Extra Segment) másodlagos adat terület címzéséhez
- **FS, GS** (nincs külön neve) 80386-től

Máté: Assembly programozás

1. előadás

11

Memóriaszervezés:

- A **CS, SS, DS, ES, FS, GS** regiszterek a visszafelé kompatibilitást biztosítják a régebbi gépekkel.
- **16 K db szegmens** lehetséges, de a **WINDOWS**-ok és **UNIX** is csak **1** szegmenst támogatnak, és ennek is egy részét az operációs rendszer foglalja el,
- minden szegmensben belül a címtartomány: **0 - 2³²-1**, 80288-ig a szegmens regiszterek 16 bitesek, a szegmensek lehetséges kezdőcíme **(0-2¹⁶-1)*16**, a szegmensben belül a címtartomány: **0 - 2¹⁶-1**
- **Little endian** tárolási mód: az alacsonyabb címen van az alacsonyabb helyértékű bájt.

Máté: Assembly programozás

1. előadás

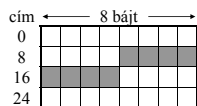
12

Memória modellek

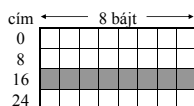
ASCII kód 7 bit + paritás → Byte (bájt)

Szó: 2 vagy 4 bájt, dupla szó 4 vagy 8 bájt.

Igazítás (alignment), **5.2. ábra:** hatékonyabb, de probléma a kompatibilitás (a **Pentium 4**-nek két ciklusra is szüksége lehet egy szó beolvasásához).



Nem igazított 8 bájtos szó
a 12-es címtől



8 bájtos szó
8 határra igazítva

Máté: Assembly programozás

1. előadás

13

Az Intel 8086/8088 – Pentium társzervezése

A memória byte szervezésű.

Egy byte 8 bitből áll. word, double word.

Byte sorrend: Little Endian (LSBfirst).

A negatív számok 2-es komplementus kódban.

szegmens, szegmens cím

a szegmensen belüli relatív cím:

logikai cím, OFFSET, Effective Address (EA),
displacement, eltolás

lineáris cím, virtuális cím, fizikai cím (Address)

Máté: Assembly programozás

1. előadás

14

A 8086/8088 FLAGS (STATUS) bitjei (flag-jei)

-	-	-	-	O	D	I	T	S	Z	-	A	-	P	-	C
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- O** (Overflow) Előjeles túlszordulás
- D** (Direction) A string műveletek iránya, **0**: növekvő, **1**: csökkenő
- I** (Interrupt) **1**: Maszkolható megszakítás engedélyezése, **0**: tiltása
- T** (Trap) **1**: „single step” (debug), **0**: Automatikus üzemmód
- S** (Sign) Az eredmény legmagasabb helyértékű bit-je (előjel bit)
- Z** (Zero) **1** (igaz), ha az eredmény **0**, különben **0** (hamis)
- A** (Auxiliary Carry) Átvitel a 3. és 4. bit között (decimális aritmetika)
- P** (Parity) Az eredmény alsó 8 bitjének paritása
- C** (Carry) Átvitel előjel nélküli műveleteknél

Máté: Assembly programozás

1. előadás

15

A 80286 FLAGS (STATUS) bitjei kiegészültek

-	NT	IOPL		O	D	I	T	S	Z	-	A	-	P	-	C
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

- NT** (Nested Task) A jelenleg futó taszk védett módon egy másik taszkba van ágyazva
- IOPL** (Input/Output Privilege Level) A taszk privilégium szintje CPL. A taszk csak akkor férhet hozzá az I/O porthoz, ha $CPL \leq IOPL$.
00 a legmagasabb, 11 a legalacsonyabb privilégium szint.

Máté: Assembly programozás

1. előadás

Nem kell

16

A 80386 EFLAGS magasabb helyértékű bitjei

80386-tól **FLAGS** további 16 bittel egészült ki, az eddigi bitek jelentése megmaradt.

														VM	RF
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16

- RF** (Resume) Nyomkövetésnél használatos. Ha 0, akkor minden utasítás végrehajtása után debug kivétel (#DE – debug exception) generálódik.
- VM** (Virtual Mode) Egy vagy több 1 MB-s DOS memória partíciót engedélyez, több DOS program futhat egyidejűleg védett módban a gépen

Máté: Assembly programozás

1. előadás

Nem kell

17

A 80486 EFLAGS 18. bitje

															AC	VM	RF
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		

- AC** (Alignment Check) Ha be van állítva és nem word címen word-re vagy nem doubleword címen doubleword-re hivatkozunk, akkor felhasználói (3-as privilégium) szinten kivétel (exception) képződik.

Máté: Assembly programozás

1. előadás

Nem kell

18

A Pentium EFLAGS további bitje

											ID	VIP	VIF	AC	VM	RF
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	

- **VIF** (Virtual Interrupt Flag) virtuális módú **I** másolata.
- **VIP** (Virtual Interrupt Pending) Ha 1, akkor egy megszakítás függőben van.
- **ID** (IDentification) Ha 1, akkor a processzor azonosítani tudja magát (támogatja a **CPUID** utasítást).

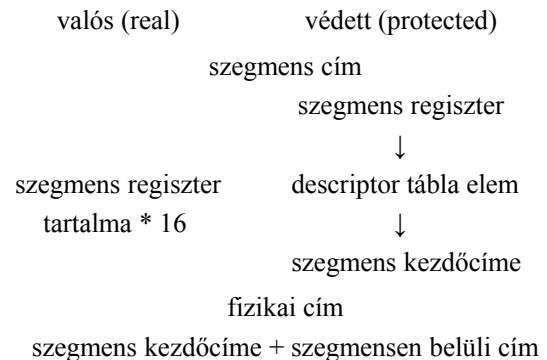
Máté: Assembly programozás

1. előadás

Nem kell

19

A fizikai cím meghatározása



Máté: Assembly programozás

1. előadás

20

Deszkriptor formátumok (8 bájt)

80286

00000000	00000000	6
Elérési jogok	B23-B16	4
Base (B15-B0)		2
Limit (L15-L0)		0

80386 – 80486 – Pentium

B31-B24	G	D	0	AV	L19-L16	6
Elérési jogok					B23-B16	4
Base (B15-B0)						2
Limit (L15-L0)						0

G = 0: limit bájtokban **G** = 1: 4 KB-os lapokban
D = 0: 16 bites mód **D** = 1: 32 bites mód
AV = 0: a szegmens nem érhető el **AV** = 1: elérhető

Máté: Assembly programozás

1. előadás

Nem kell

21

Az I8086/88 címzési rendszere

Operandus megadás

Adat megadás

- **Kódba épített adat** (immediate – közvetlen operandus)
MOV AL, 6 ; AL új tartalma 6
MOV AX, 0FFH ; AX új tartalma 000FFH
- **Regiszter címzés:**
MOV AX, BX

Az egyik cím mindig regiszter!

A többi adat megadás esetén az automatikus szegmens regiszter: **DS**

Máté: Assembly programozás

1. előadás

22

Direkt memória címzés: a címrészen az operandus logikai címe (eltolás, displacement)

MOV AX, SZO ; AX új tartalma SZO tartalma

MOV AL, KAR ; AL új tartalma KAR tartalma

Valahol a **DS** által mutatott szegmensben:

SZO DW 1375H

KAR DB 3FH

(**DS:SZO**) illetve (**DS:KAR**)

MOV AX, KAR ; hibás

MOV AL, SZO ; hibás

Máté: Assembly programozás

1. előadás

23

- **Indexelt címzés:** a logikai cím:
a 8 vagy 16 bites eltolás + **SI** vagy **DI** (esetleg **BX**) tartalma

MOV AX, 10H[SI]

MOV AX, -10H[SI]

MOV AX, [SI]

Regiszter-indirekt címzés: eltölési érték nélküli indexelt címzés

MOV AX, [BX]

MOV AX, [SI]

- **Bázis relatív (bázisindex) címzés:** a logikai cím:
eltolás + **BX** + **SI** vagy **DI** tartalma

MOV AX, 10H[BX][SI]

MOV AX, [BX+SI+10H]

Máté: Assembly programozás

1. előadás

24

Stack (verem) terület címzés

Automatikus szegmens regiszter: **SS**

Megegyezik a bázis relatív címzéssel, csak a **BX** regiszter helyett a **BP** szerepel.

Máté: Assembly programozás

1. előadás

25

Program terület címzés

Automatikus szegmens regiszter: **CS**

A végrehajtandó utasítás címe: **(CS:IP)**

Egy utasítás végrehajtásának elején:

IP = IP + az utasítás hossza.

- IP relatív címzés:**

IP = IP + a 8 bites előjeles közvetlen operandus

- Direkt utasítás címzés:** Az operandus annak az utasításnak a címe, ahova a vezérlést átadni kívánjuk.

Közele (NEAR): **IP** <= a 16 bites operandus

Távoli (FAR): **(CS:IP)** <= a 32 bites operandus.

CALL VALAMI ; az eljárás típusától függően
; **NEAR** vagy **FAR**

Máté: Assembly programozás

1. előadás

26

- Indirekt utasítás címzés:** Bármilyen adat címzési móddal megadott szóban vagy dupla szóban tárolt címre történő vezérlés átadás. Pl.:

JMP AX ; ugrás az AX-ben tárolt címre

JMP [BX] ; ugrás a (DS:BX) által címzett ; szóban tárolt címre.

JMP FAR [BX] ; ugrás a (DS:BX) által ; címzett dupla szóban tárolt címre.

Máté: Assembly programozás

1. előadás

27

Az utasítások szerkezete

prefixum	operációs kód	címzési mód	operandus
0 - 2 byte	1 byte	0 - 1 byte	0 - 4 byte

Prefixum:

utasítás ismétlés, explicit szegmens megadás vagy **LOCK**

MOV AX, CS:S ; S nem a DS,

; hanem a CS regiszterrel címzendő

Operációs kód: szimbolikus alakját mnemonic-nak nevezzük

Címzési mód byte: hogyan kell az operandust értelmezni

Operandus: mivel kell a műveletet elvégezni

Máté: Assembly programozás

1. előadás

28

Címzési mód byte

A legtöbb utasítás kód után szerepel. Szerkezete:

7	6	5	4	3	2	1	0
Mód		Regiszter			Reg/Mem		

Ha a műveleti kód legalacsonyabb helyértékű bit-je

0, akkor **byte**-os művelet,

1, akkor **word**-ös (szavas) művelet.

Máté: Assembly programozás

1. előadás

29

Regiszter			Reg/Mem jelentése, ha Mód =			
	byte	word	00	01	10	11
000	AL	AX	BX + SI + DI	„00” + 8 bit displ.	„00” + 16 bit displ.	R e g i s t e r
001	CL	CX				
010	DL	DX	BP + SI + DI			
011	BL	BX				
100	AH	SP	SI DI			
101	CH	BP				
110	DH	SI	16 bit d.	BP+8 bit d.	BP+16 bit d.	
111	BH	DI	BX	„00”+8 bit	„00”+16 bit	
Máté: Assembly programozás						
			1. előadás		30	

Máté: Assembly programozás

1. előadás

30

Szimbolikus alakban az operandusok sorrendje, gépi utasítás formájában a gépi utasítás kód mondja meg a regiszter és a memória közti adatátvitel irányát.

Máté: Assembly programozás

1. előadás

31

Pl. az alábbi két utasítás esetén a címzési mód byte megegyezik:

MOV AX, 122H[SI+BX]
 ; hexadecimálisan **8B 80 0122**

MOV 122H[SI+BX], AX
 ; hexadecimálisan **89 80 0122**

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	0
Mód		Regiszter			Reg/Mem		
+16 bit d.		AX			SI+BX		

Máté: Assembly programozás

1. előadás

32

Nem kell

Az általános regiszterek és **SI**, **DI**, **SP**, **BP** korlátlanul használható, a többi (a szegmens regiszterek, **IP** és **STATUS**) csak speciális utasításokkal. Pl.:

MOV DS, ADAT ; hibás!

MOV AX, ADAT ; helyes!

MOV DS, AX ; helyes!

A többi regiszter nem lehet aritmetikai utasítás operandusa, sőt, **IP** és **CS** csak vezérlés átadó utasításokkal módosítható, közvetlenül nem is olvasható.

Máté: Assembly programozás

1. előadás

33