



MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-9

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

Assembly Language Programming

Vize Soruları:

- **SORU 1)** Bir mikroişlemcinin çalışma mantığını (içinde bulunması gereken bileşenler, diğer birimlerle nasıl haberleştiği) kısa ve anlaşılır cümlelerle açıklayınız.
- **SORU 2)** Flip-floplar mikroişlemciler için neden önemlidir?
- **SORU 3)** Komut seti ve belleği kullanım açısından mikroişlemciler nasıl sınıflandırılır, aradaki farkları kısaca anlatınız.
- **SORU 4)** Mikroişlemciler içindeki transistörlerin sayısının artması ne avantaj sağlar?
- **SORU 5)** 8086 mikroişlemcisi ve bileşenleri düşünüldüğünde, bu mikroişlemcinin avantaj ve dezavantajlarını yazınız.
- **SORU 6)** Mikro kod nedir? Neden ihtiyaç vardır ve nerede saklanır kısaca açıklayınız?

Vize Soruları:

MOV AX, 0100H

MOV SS, AX

MOV SP, 1233H

MOV AX, 0AF2H

MOV CH, F1H

MOV CL, 32H

MOV BX, 01H

PUSH AX

POP AX

PUSH BX

POP BX

PUSH CX

POP CX

Vize soruları: Benzer şekilde emülatörde çalıştırabilirsiniz

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 12B3H
MOV AX, 0AF2H
MOV CH, FFH
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP BX
PUSH BX
POP BX
PUSH CX
POP BX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 1233H
MOV AX, 0AF2H
MOV CH, F1H
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP AX
PUSH AX
POP BX
PUSH BX
POP CX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 1233H
MOV AX, 1AF2H
MOV CH, F1H
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP AX
PUSH BX
POP BX
PUSH BX
POP CX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 1232H
MOV AX, 0AF2H
MOV CH, F1H
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP AX
PUSH AX
POP BX
PUSH CX
POP CX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 12B3H
MOV AX, 0AF2H
MOV CH, FFH
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP BX
PUSH BX
POP CX
PUSH CX
POP BX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 1235H
MOV AX, 0AF2H
MOV CH, F1H
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP AX
PUSH AX
POP BX
PUSH BX
POP CX
```

```
MOV AX, 0100H
MOV SS, AX
MOV SP, 1233H
MOV AX, 1AF2H
MOV CH, F1H
MOV CL, 32H
MOV BX, 01H
PUSH AX
POP AX
PUSH BX
POP BX
PUSH CX
POP CX
```

Vize Soruları:

- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, $4K \times 8$ 'lik RAM kullanarak $20K \times 8$ 'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi $\$000F$ 'dir.
 - $000F + FFF = 100E$ adresinden başlayacak
 - $100F - 200E$
 - $200F - 300E$
 - $300F - 400E$
 - $400F - 500E$
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, $8K \times 8$ 'lik RAM kullanarak $24K \times 8$ 'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi $\$001D$ 'dir.
 - $001D + 1FFF = 201C$ adresinden başlayacak
 - $201D - 401C$
 - $401D - 601C$
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, $8K \times 8$ 'lik ROM kullanarak $32K \times 8$ 'lik ROM belleği tasarlanacaktır. ROM belleğin başlangıç adresi $\$00FC$ 'dir.
 - $00FC + 1FFF = 20FB$ adresinden başlayacak
 - $20FC - 40FB$
 - $40FC - 60FB$
 - $60FC - 80FB$
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, $4K \times 8$ 'lik RAM kullanarak $12K \times 8$ 'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi $\$00F6$ 'dir.
 - $00F6 + FFF = 10F5$ adresinden başlayacak
 - $10F6 - 20F5$
 - $20F6 - 30F5$

Vize Soruları:

- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, 8K*8'lik RAM kullanarak 24K*8'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi \$001E'dir.
 - $0001E + 1FFF = 201D$ adresinden başlayacak
 - 201E – 401D
 - 401E – 601D
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, 8K*8'lik ROM kullanarak 32K*8'lik ROM belleği tasarlanacaktır. ROM belleğin başlangıç adresi \$00F2'dir.
 - $000F2 + 1FFF = 20F1$ adresinden başlayacak
 - 20F2 – 40F1
 - 40F2 – 60F1
 - 60F2 – 80F1
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, 4K*8'lik RAM kullanarak 12K*8'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi \$00F0'dir.
 - $000F0 + FFF = 10EF$ adresinden başlayacak
 - 10F0 – 20EF
 - 20F0 – 30EF
- 8 bitlik veri yolu ve 16 bitlik adres yolu olan bir mikroişlemcili sisteminde, 4K*8'lik RAM kullanarak 20K*8'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi \$000B'dir.
 - $000B + FFF = 100A$ adresinden başlayacak
 - 100B – 200A
 - 200B – 300A
 - 300B – 400A
 - 400B – 500A

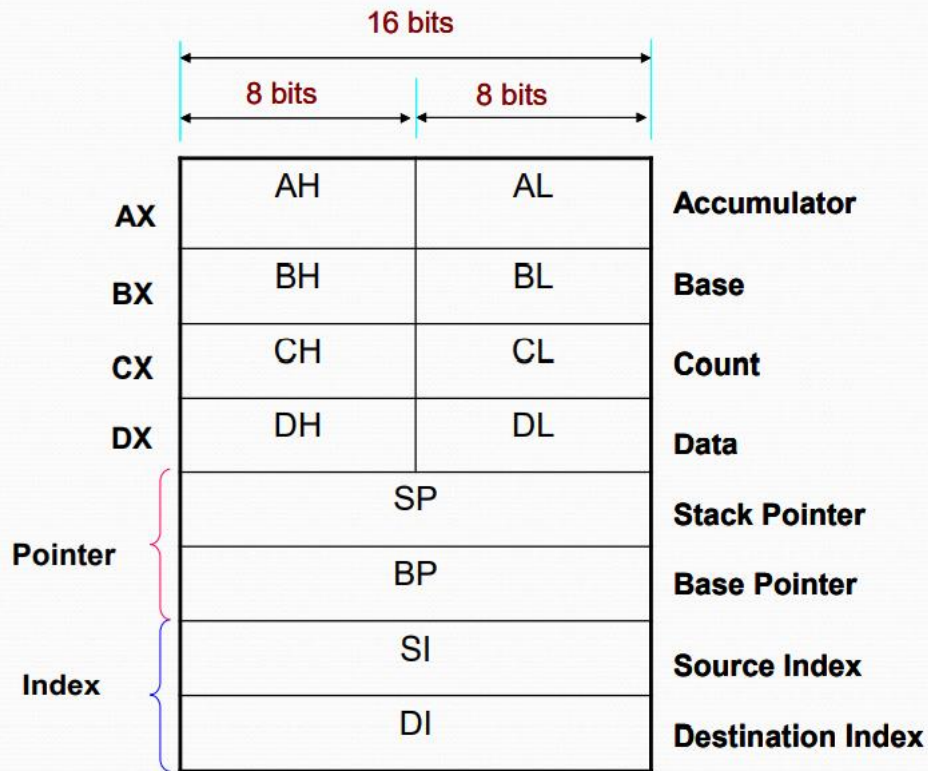
8086 16-Bit Mikroişlemci

- 8086 işlemcisi için yazılan tüm komutlar bugünkü x86 tabanlı bilgisayarlarda çalışabilmektedir.
- CISC mimarisindedir.
- 16 bit veri yolu
- 20 bit adres yolu; 20-bitlik adresleme özelliği; $2^{20} = 1048576 = 1\text{MB}$ lık belleğe erişebilme (Adresler: xxxxx → 00000H-FFFFFFH)
- 8 adet genel amaçlı 16 bitlik register
 - **AX** - accumulator register – akümülatör (**AH / AL**).
 - **BX** - the base address register – adres başlangıcı (**BH / BL**).
 - **CX** - the count register – sayma (**CH / CL**).
 - **DX** - the data register – veri (**DH / DL**).
 - **SI** - source index register – kaynak indisi.
 - **DI** - destination index register – hedef indisi.
 - **BP** - base pointer – temel gösterici.
 - **SP** - stack pointer – yığıt gösterici.

8086 16-Bit Mikroişlemci

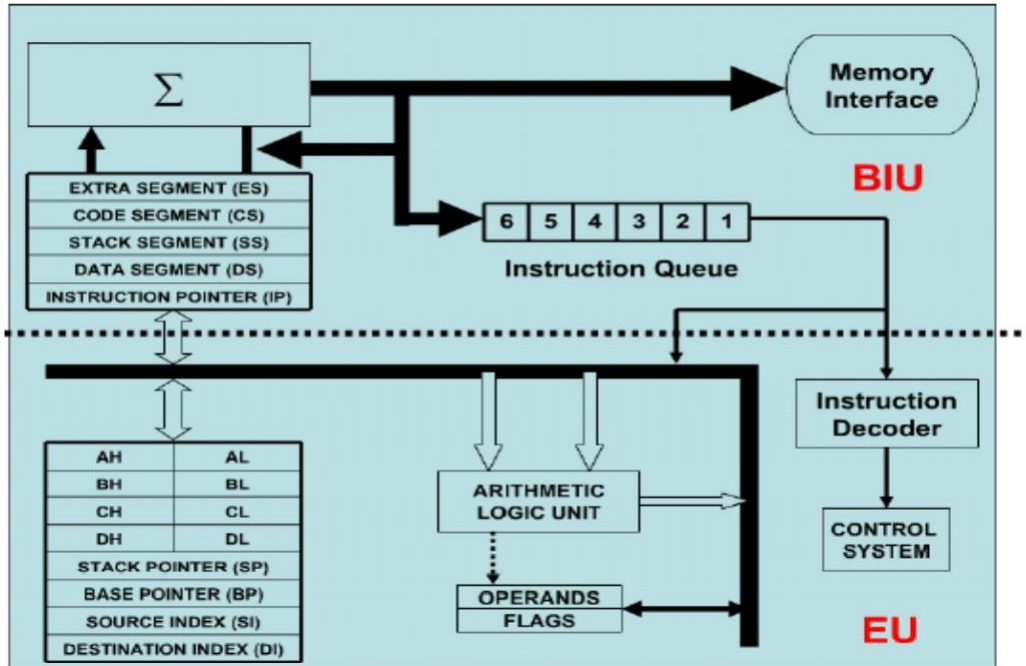
- Segment registerlerinin özel amaçları vardır, bellekte ulaşılabilir bazı bölümleri işaretler.
 - Segment registerları, genel amaçlı registerları ile birlikte çalışarak hafızada herhangi bir bölgeyi işaretleyebilir.
- **CS** – (Code Segment) Mevcut programın bulunduğu bölümü işaretler.
 - **DS** – (Data Segment) Genellikle programda bulunan değişkenlerin bulunduğu bölümü işaretler.
 - **ES** – (Extra Segment) Bu register'ın kullanımı, kullanıcıya bırakılmıştır.
 - **SS** – (Stack Segment) yığının bulunduğu bölümü işaretler.

8086 16-Bit Mikroişlemci



Register	Purpose
AX	Word multiply, word divide, word I /O
AL	Byte multiply, byte divide, byte I/O, decimal arithmetic
AH	Byte multiply, byte divide
BX	Store address information
CX	String operation, loops
CL	Variable shift and rotate
DX	Word multiply, word divide, indirect I/O (Used to hold I/O address during I/O instructions. If the result is more than 16-bits, the lower order 16-bits are stored in accumulator and higher order 16-bits are stored in DX register)

8086 16-Bit Mikroişlemci

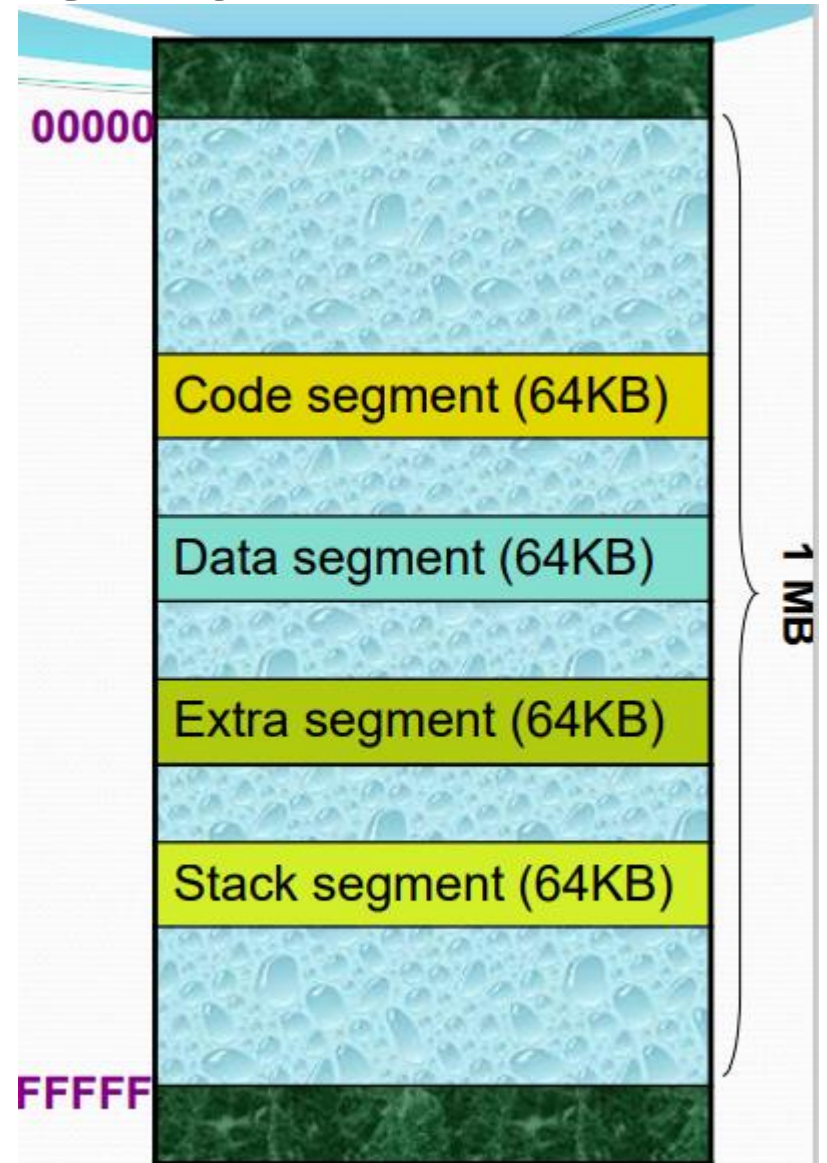


8086 mimarisinin diyagramı

- BIU (Bus Interface Unit): Bus'lar üzerindeki tüm veri ve adres hareketlerini EU için halleder.
 - emirleri getirme (instruction fetching),
 - operandların adreslerini hesaplama,
 - belleğe operand yazma/bellekten operand okuma,
 - emir byte'larını emir kuyruğuna (instruction queue) transfer etme, gibi tüm bus işlemlerini yapar
- EU: Emirleri veya verileri hangi adreslerden getireceğini BIU birimine söyler
 - Emirlerin kodunu çözer / Emirleri icra eder (decode & execute)

8086 16-Bit Mikroişlemci

- 16-bitlik registerlar ile en fazla adreslenebilir bellek uzayı :64 KB
- $2^{16}=65536=64\text{KB}$
→xxx: 0000H-FFFFH



8086 16-Bit Mikroişlemci

- Fiziksel Adres: **segment adresi + offset adresi**
 - Segment Adres: Hex ise;
Segment Register x 10H ile çarpar
 - Segment Adres: Decimal ise;
Segment Register x 16 ile çarpar
- Örn. **CS=0722H** ve **IP=0003H**, mikroişlemci, bir sonraki komutu
 - **07220H** (**0722H**x10H=**07220H**) + 0003H = **07223H**
adresinden okur.

8086 16-Bit Mikroişlemci

emulator: noname.exe_

file math debug view external virtual devices virtual drive help

Load reload step back single step run step delay ms: 0

registers

	H	L
AX	07	10
BX	00	00
CX	01	37
DX	00	00
CS	0722	
IP	0003	
SS	0712	
SP	0100	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

0722:0003

0722:0003

07220: B8 184 ı
07221: 10 016 ı
07222: 07 007 BEEP
07223: 8E 142 ı
07224: D8 216 ı
07225: 8E 142 ı
07226: C0 192 L
07227: BA 186 ıı
07228: 00 000 NULL
07229: 00 000 NULL
0722A: B4 180 ı
0722B: 09 009 TAB
0722C: CD 205 =
0722D: 21 033 ı
0722E: B4 180 ı
0722F: 01 001 ı
07230: CD 205 =
07231: 21 033 ı
07232: B8 184 ı
07233: 00 000 NULL
07234: 4C 076 L
07235: CD 205 =

MOV AX, 00710h
MOV DS, AX
MOV ES, AX
MOV DX, 00000h
MOV AH, 09h
INT 021h
MOV AH, 01h
INT 021h
MOV AX, 04C00h
INT 021h
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
NOP
...

screen source reset aux vars debug stack flags

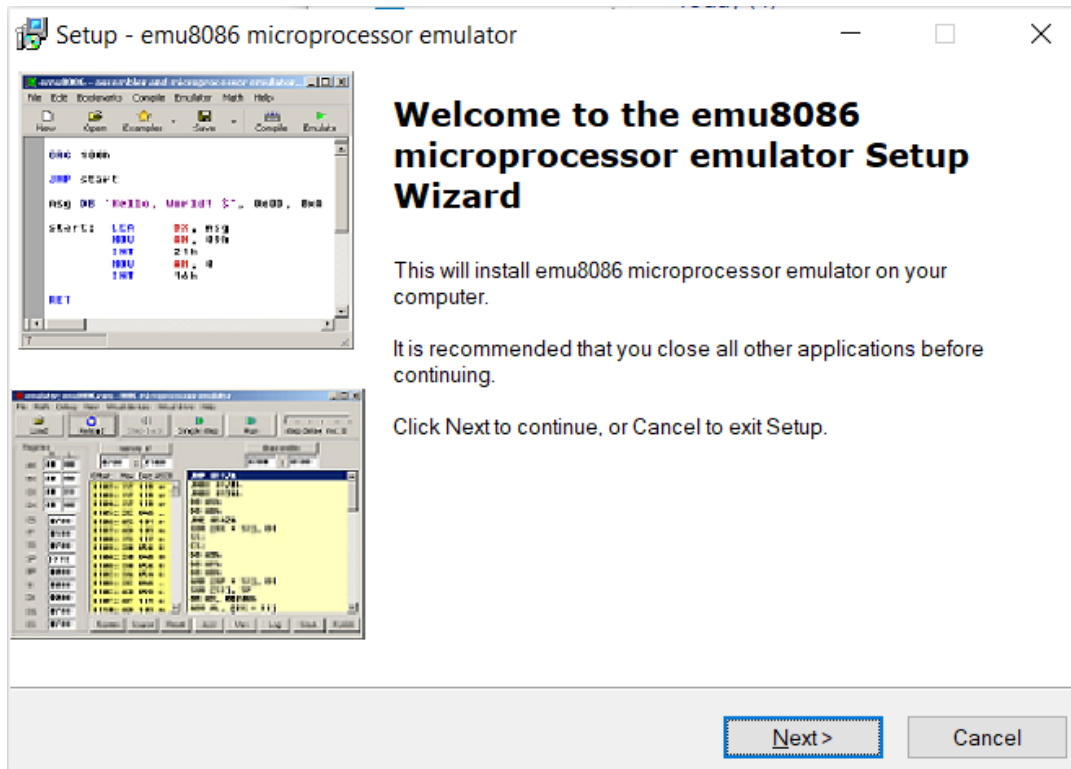
8086 16-Bit Mikroişlemci

- Segment ve adres register çiftleri:
- CS:IP
- SS:SP SS:BP
- DS:BX DS:SI
- DS:DI
- ES:DI

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- <https://emu8086-microprocessor-emulator.softonic.com.tr/>
adresinden emülatörü indirebilirsiniz.



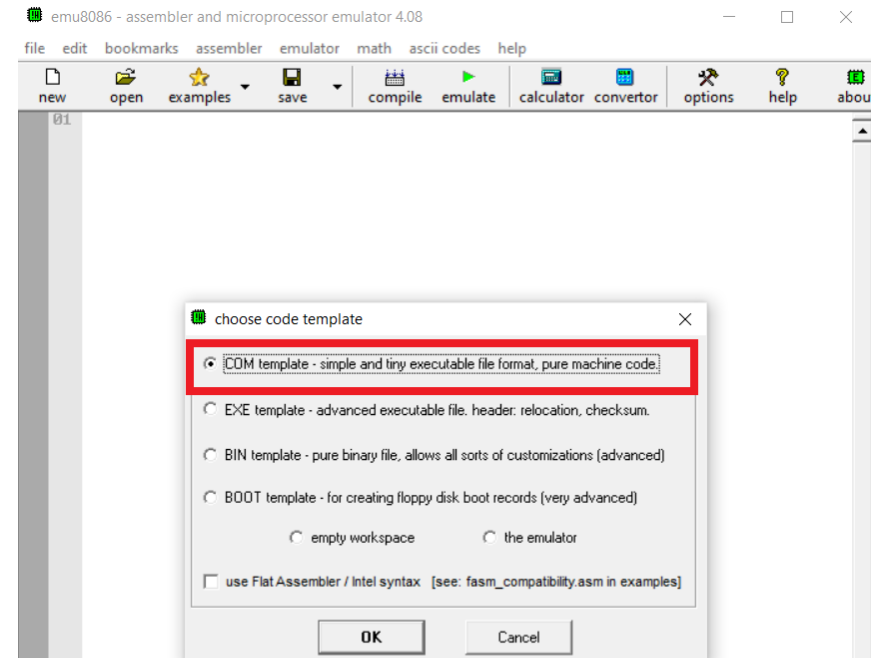
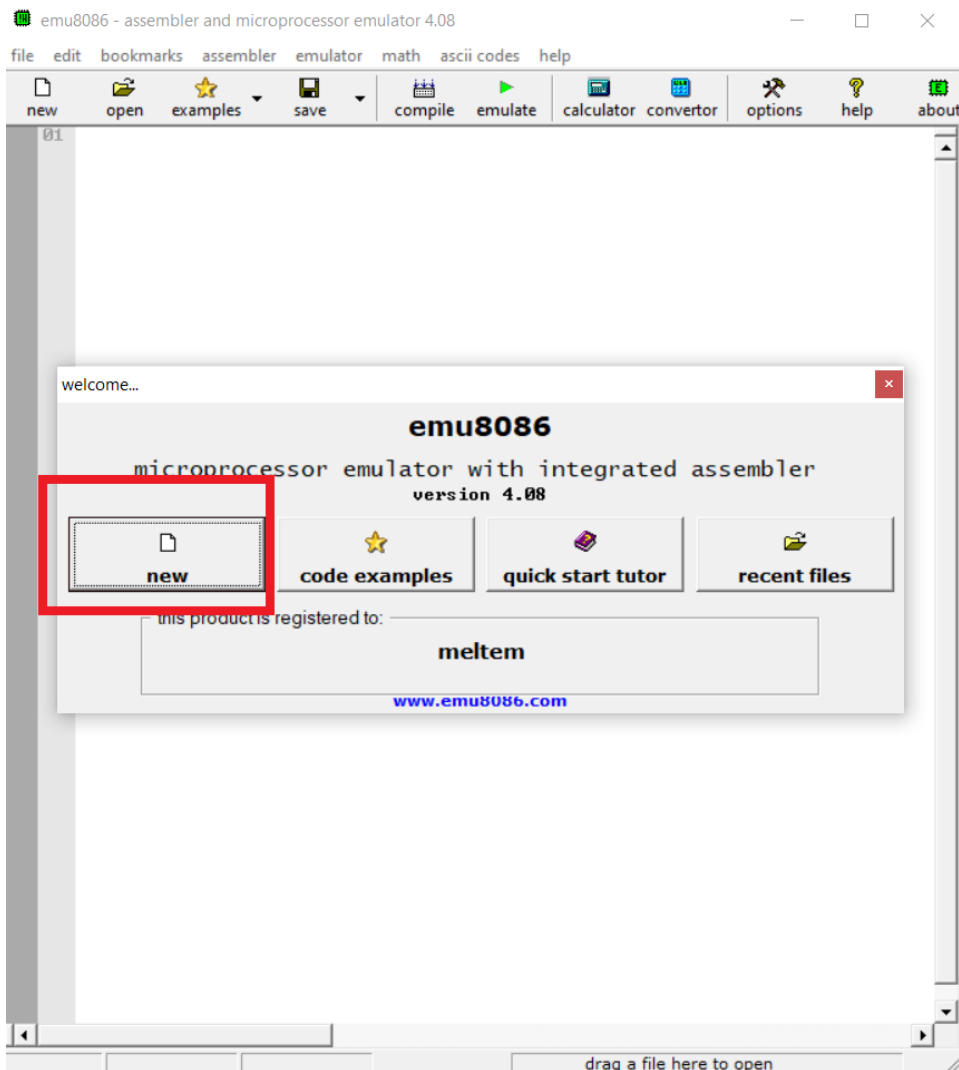
8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- Kurulumu tamamladıktan sonra
27R3VDEFYFX4N0VC3FRTQZX anahtarı ile
aktivasyonunuzu yapın

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- Com template oluşturduk: 64 KB segment var tüm segment registerları aynı adresten başlıyor

The screenshot displays the EMU8086 assembler and microprocessor emulator interface. The main window shows the 'run' button highlighted in red. The 'original source code' window displays the following assembly code:

```
; You may customize this and other start-up templates;  
; The location of this template is c:\emu8086\inc\0_com_template.txt  
  
org 100h  
  
; add your code here  
  
ret
```

The 'registers' window shows the following values:

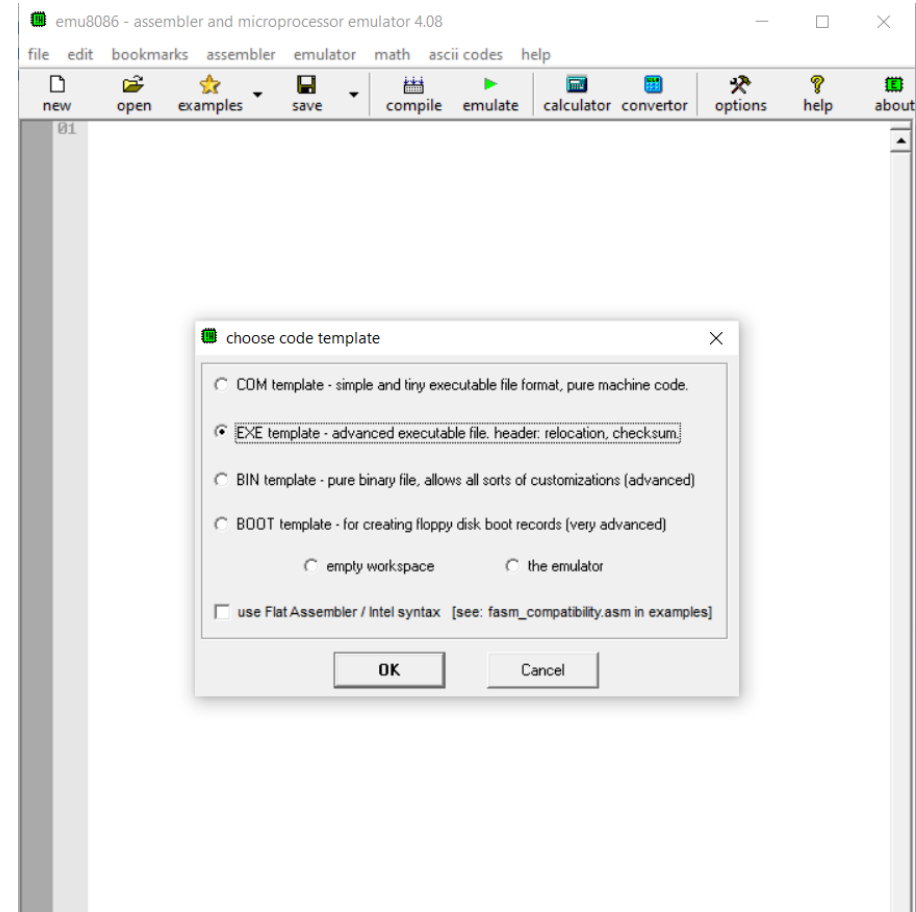
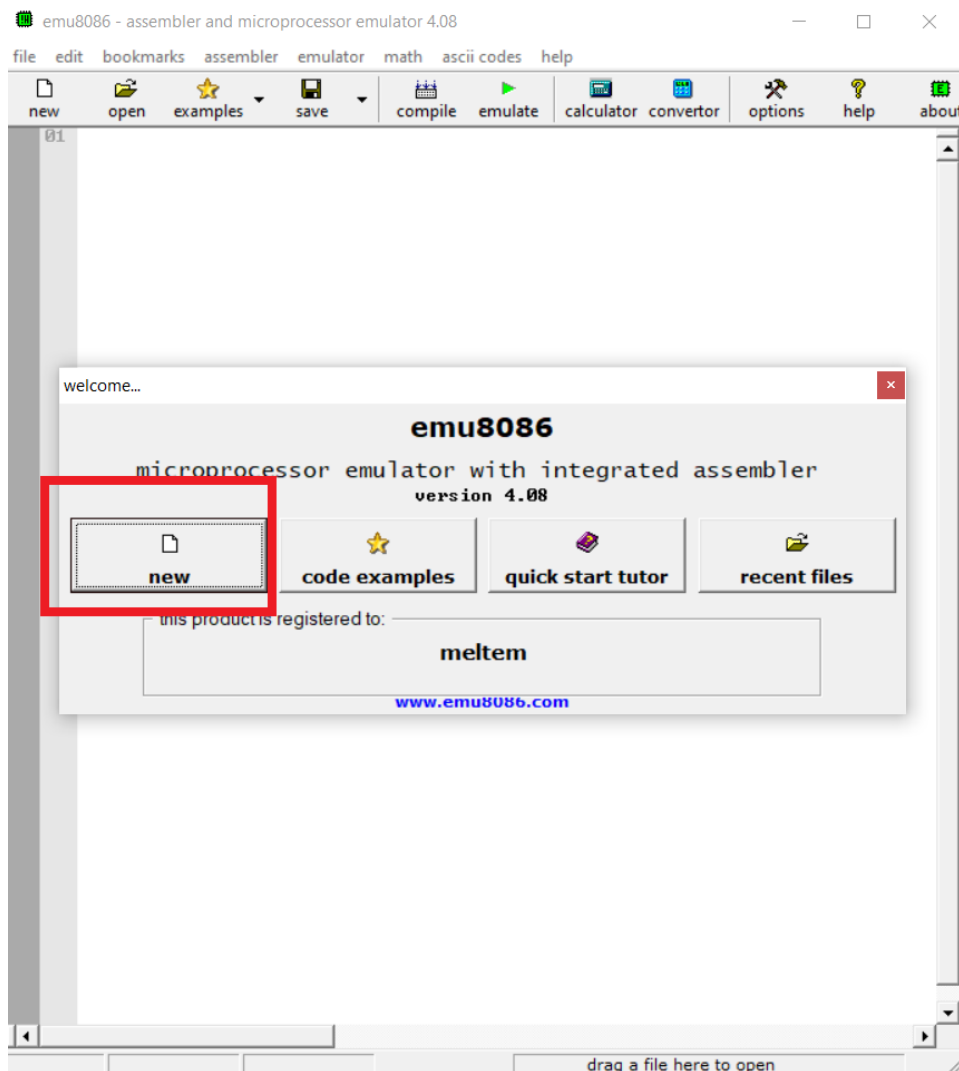
Register	H	L
AX	00	00
BX	00	00
CX	00	01
DX	00	00
CS	0700	
IP	0000	
SS	0700	
SP	0000	
BP	0000	
SI	0000	
DI	0000	
DS	0700	
ES	0700	

The 'debug log - debug.exe emulation' window shows the following output:

```
IX=0000 BX=0000 CX=0001 DX=0000 SP=0000 BP=0000 SI=0000 DI=0000  
IS=0700 ES=0700 SS=0700 CS=0700 IP=0000 NU UP EI PL NZ NA PO NC  
I700:0000 CD20 INT 020h
```

8086 16-Bit Mikroişlemci

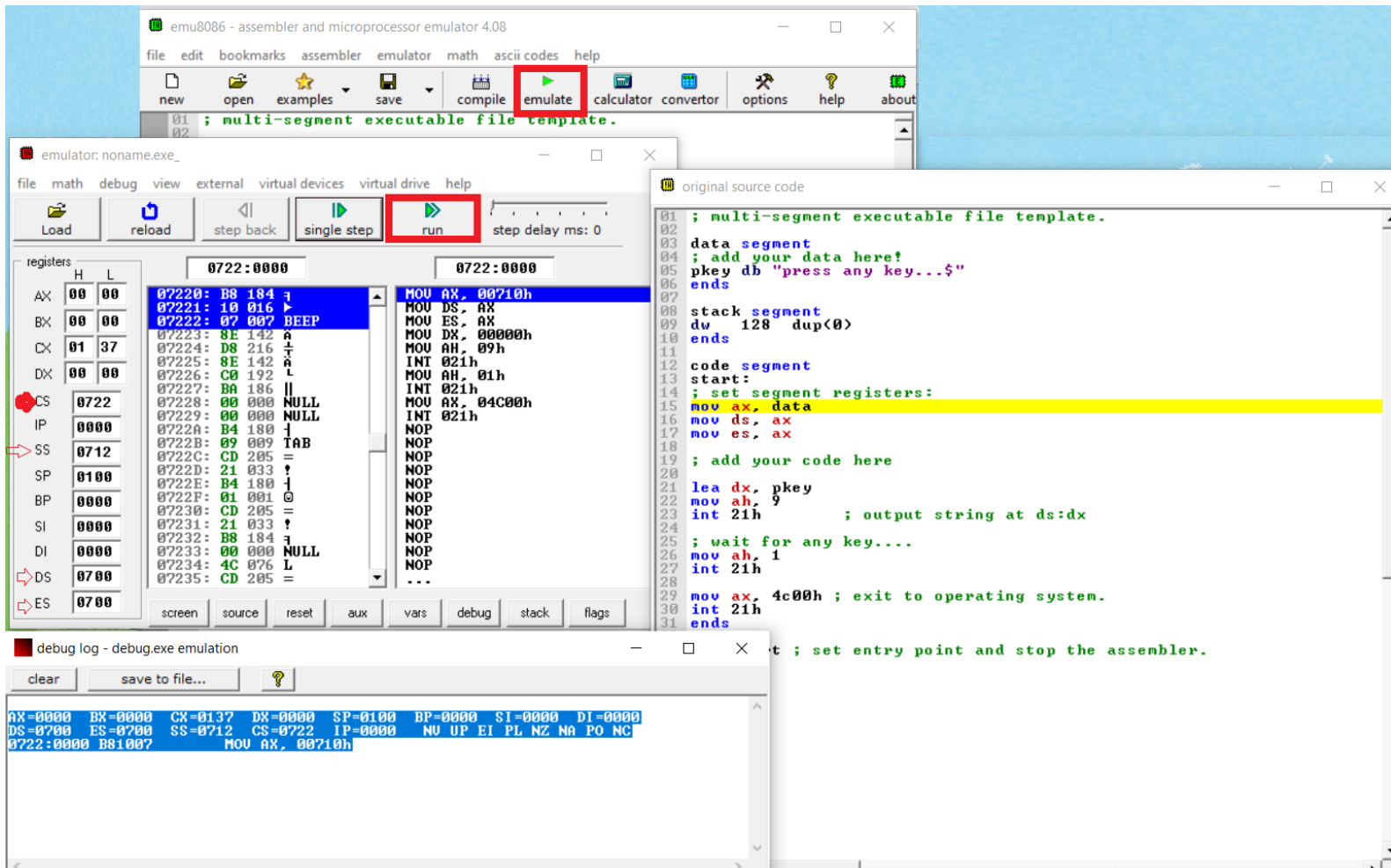
EMU 8086-MICROPROCESSOR EMULATOR



8086 16-Bit Mikroişlemci

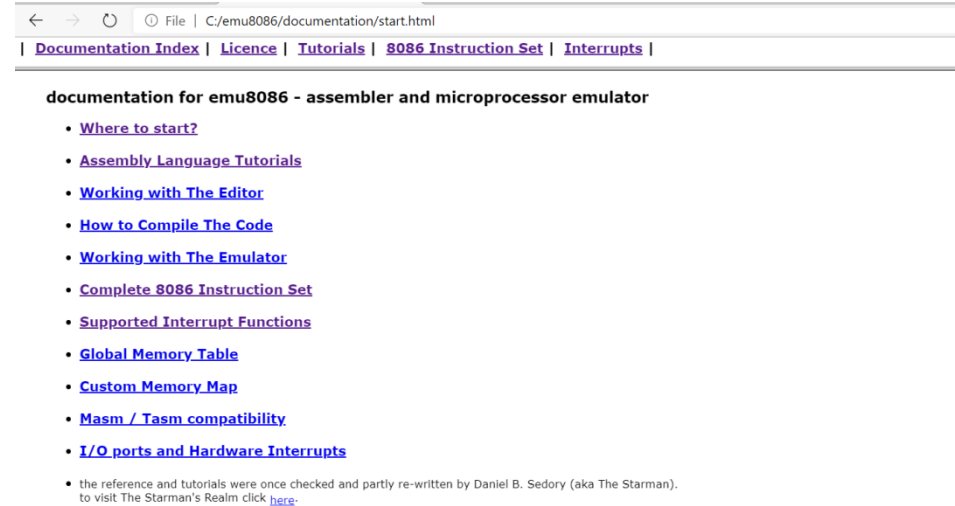
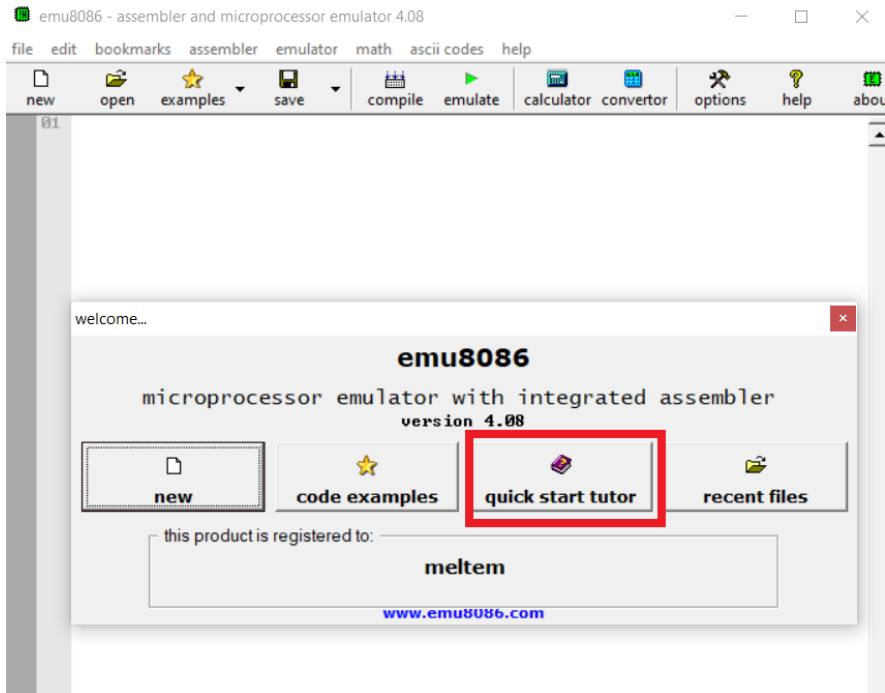
EMU 8086-MICROPROCESSOR EMULATOR

- Exe template oluşturduk: Ayrı ayrı segment registerlarının tanımlamalarını yapabiliyoruz



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR



- **quick start tutor** sekmesinden 8086 ile ilgili tüm detaylı bilgilere ulaşabilirsiniz.

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

; ile açıklama satırı ekleniyor

- emu8086 büyük küçük harf duyarlı değil bu nedenle değişken tanımlamalarına dikkat etmek gereklidir çünkü $ab=Ab=AB=aB$ dir

komut operand1, operand2

Komutlar (instruction) en fazla **2 operand** alabilir

Tek operand ve operandsız komutlar bulunur

exe uzantılı dosya için aşağıdaki kod işletim sistemine donus komutudur her programda eklenmeli:

mov ax, 4c00h

int 21h ; (int: interruptın kısaltmasıdır)

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

Degiskenadi DB deger (DB:define byte 8-bit)

Degiskenadi DW deger (DW:define word16-bit)

sayi1 db 12

Sayi2 dw 15h

sayilar2 db 3 dup(8) ; 3 tane 8 sakla dup(duplicate)

sayilar3 db 2 dup(1,2,3) ; 1,2,3,1,2,3 sakla

sayilar4 db 1,2,4 dup(3),4 ; 1,2,3,3,3,3,4 sakla

sayilar5 db 3 dup(?) ; 3 tane bos alan birak

sayilar6 dw 13h,124Fh,0021h ; sayı dizisi tanımlama

metin db 'okul'

faiz equ 8 ; faiz diye bir sabit tanımlıyoruz

mov ax, faiz ; bu komutu kullandığımızda AX registerina faiz degiskeninin degerini atamis oluyoruz

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

The screenshot displays the EMU 8086 Microprocessor Emulator interface, which is divided into several panes:

- Assembly Code Pane:** Shows the assembly code being executed. The code includes comments in Turkish and assembly instructions like `org 100h`, `mov ax, faiz`, `ret`, and data definitions for `sayilar2` through `sayilar6` and `metin`.
- Registers Pane:** Displays the current values of the 8086 registers. The `AX` register is highlighted, showing a value of `0000`.
- Memory Pane:** Shows the memory address `0700:0100` and the corresponding memory contents. The memory is organized into a table with columns for address, hex value, and ASCII value.
- Original Source Code Pane:** Displays the original source code in a separate window, showing the same assembly code as the main pane.
- Variables Pane:** Shows the current values of variables defined in the code. The variables `SAYI1` and `SAYI2` are highlighted, showing values of `0Ch` and `0015h` respectively.

The interface also includes a menu bar (file, edit, bookmarks, etc.), a toolbar with buttons for Load, reload, step back, single step, run, and step delay, and a status bar at the bottom.