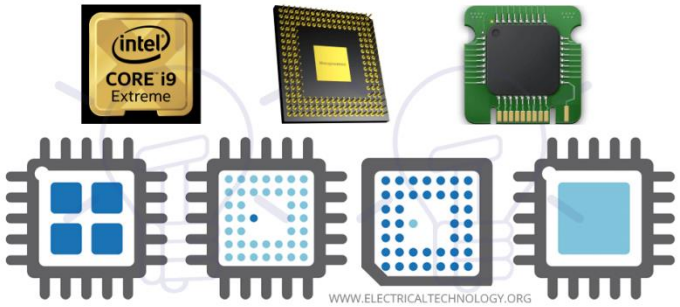


MİKROİŞLEMCİLER

Dr. Meltem KURT PEHLİVANOĞLU

W-9

MİKROİŞLEMCİLER

Digital Logic +

Digital Design +

Computer Architecture +

Microprocessors +

Microcontrollers +

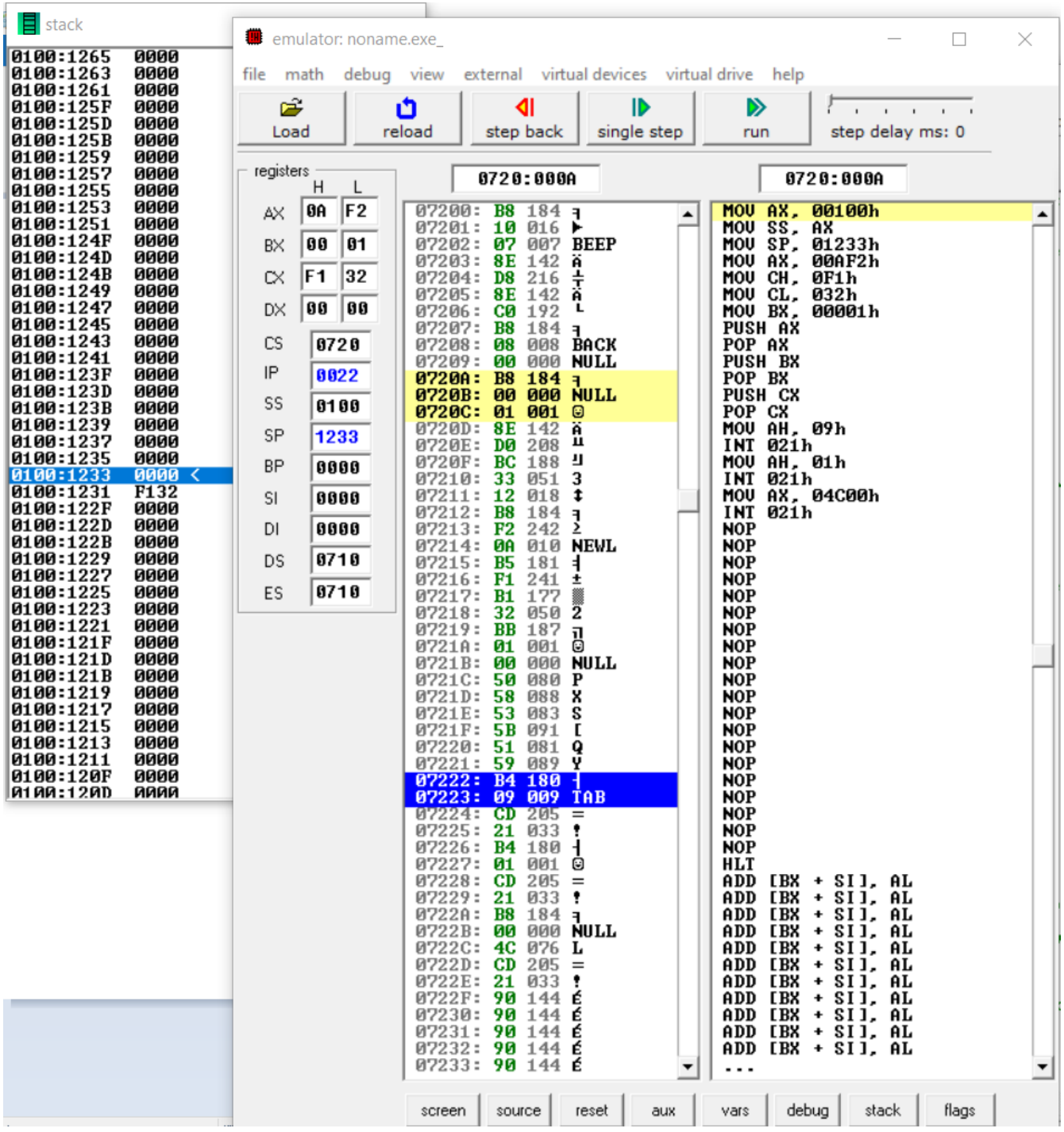
Assembly Language Programming

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)** Mikrokod 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



- 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*8'lik RAM kullanarak 20K*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*8'lik RAM kullanarak 24K*8'lik RAM belleği tasarlanacaktır. RAM belleğin başlangıç adresi $001D'dir.
  - 0001D + 1FFF = 201C adresinden başlayacak
  - 201D – 401C
  - 401D – 601C
- 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 $00FC'dir.
  - 000FC + 1FFF = 20FB adresinden başlayacak
  - 20FC – 40FB
  - 40FC- 60FB
  - 60FC – 80FB
- 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 $00F6'dir.
  - 000F6 + 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=1MB lık belleğe erişebilme (Adresler: xxxxx→ 00000H-FFFFFH)
- 8 adet genel amaçlı 16 bitlik register

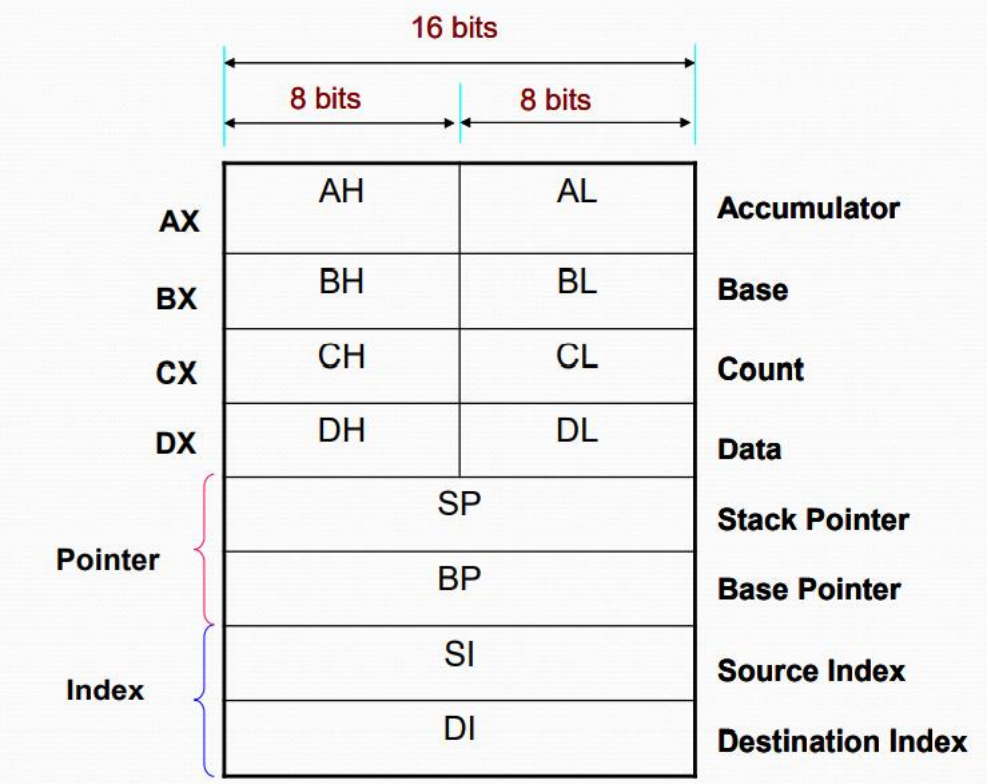
- **AX** - accumulator register – akümülatör (**AH / AL**).
- **BX** - the base address register – adres başlangıcı (**BH / BL**).
- **CX** - the count register – sayma (**CH / CL**).
- **DX** - the data register – veri (**DH / DL**).
- **SI** - source index register – kaynak indisi.
- **DI** - destination index register – hedef indisi.
- **BP** - base pointer – temel gösterici.
- **SP** - stack pointer – yığıt gösterici.

8086 16-Bit Mikroişlemci

- Segment registerlarının özel amaçları vardır, bellekte ulaşılabilir bazı bölümleri işaretler.
- Segment registerları, genel amaçlı registerları ile birlikte çalışarak hafızada herhangi bir bölgeyi işaretleyebilir.

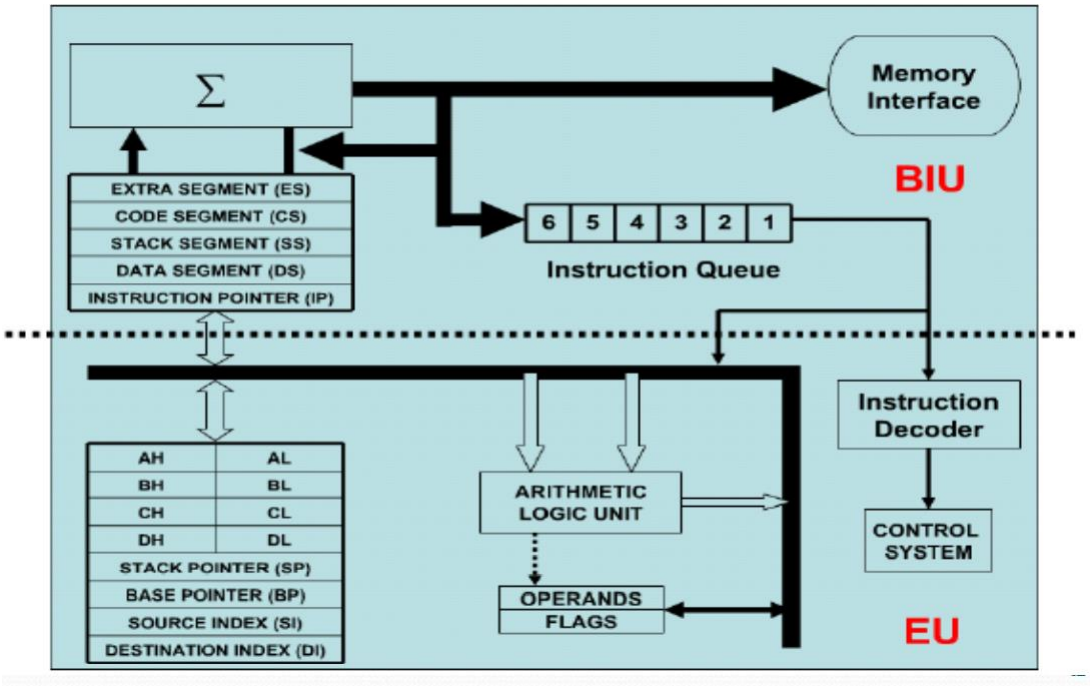
- **CS** – (Code Segment) Mevcut programın bulunduğu bölümü işaretler.
- **DS** – (Data Segment) Genellikle programda bulunan değişkenlerin bulunduğu bölümü işaretler.
- **ES** – (Extra Segment) Bu register'ın kullanımı, kullanıcıya bırakılmıştır.
- **SS** – (Stack Segment) yığının bulunduğu bölümü işaretler.

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

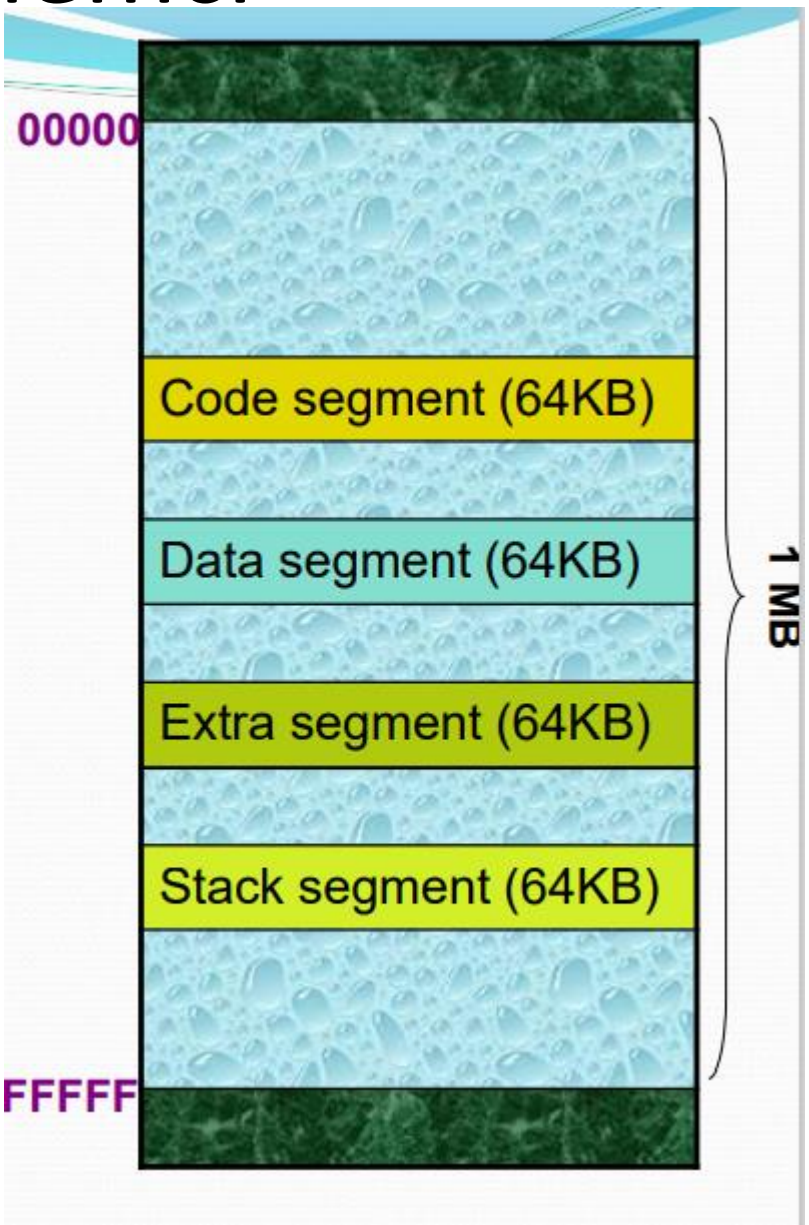


8086 mimarisinin diyagramı

- 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=64KB →xxxx: 0000H-FFFFH



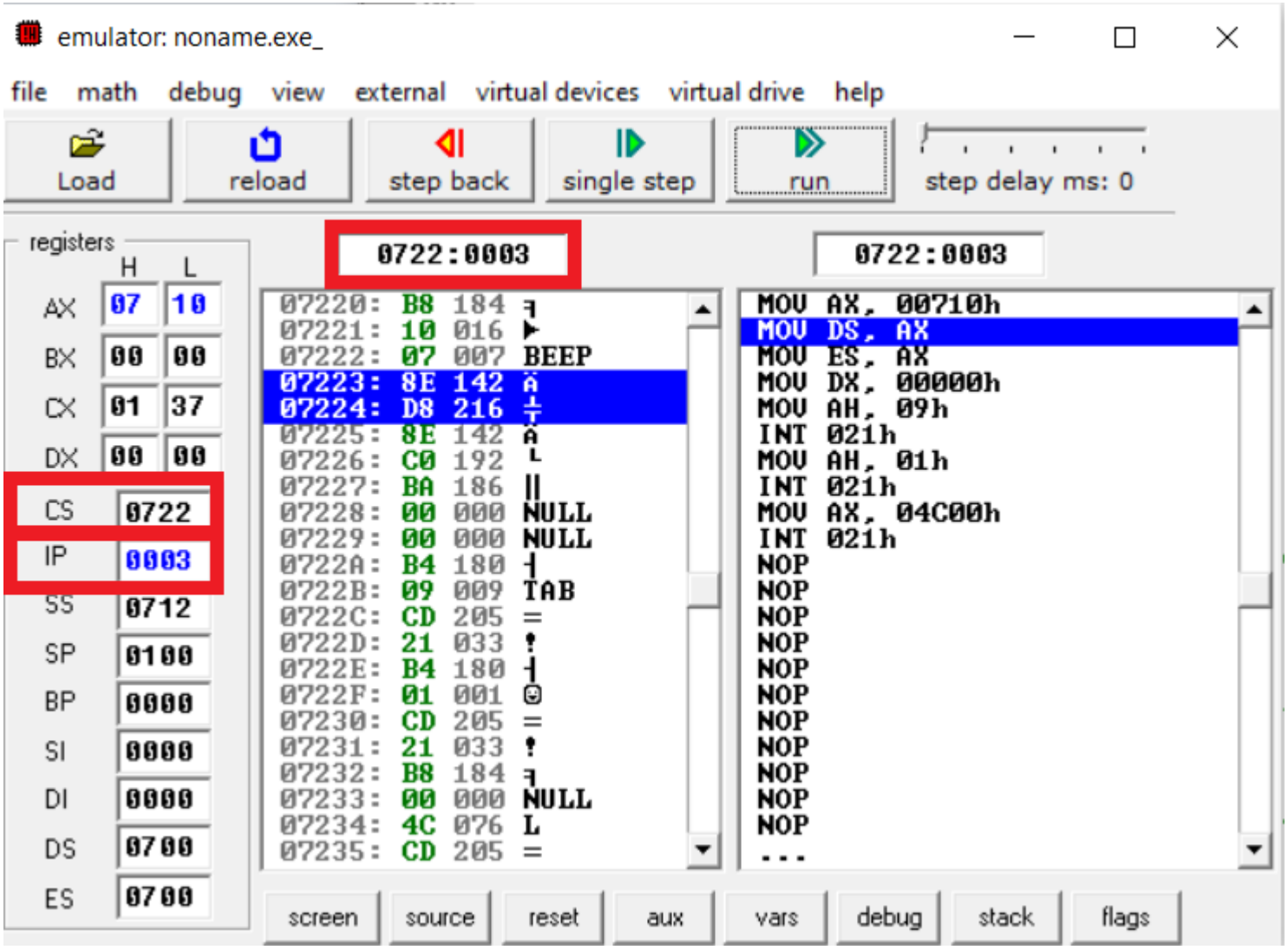
8086 16-Bit Mikroişlemci

- 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
  - **07220**H (**0722**Hx10H=**07220**H) + 0003H = **07223H** adresinden okur.

8086 16-Bit Mikroişlemci



8086 16-Bit Mikroişlemci

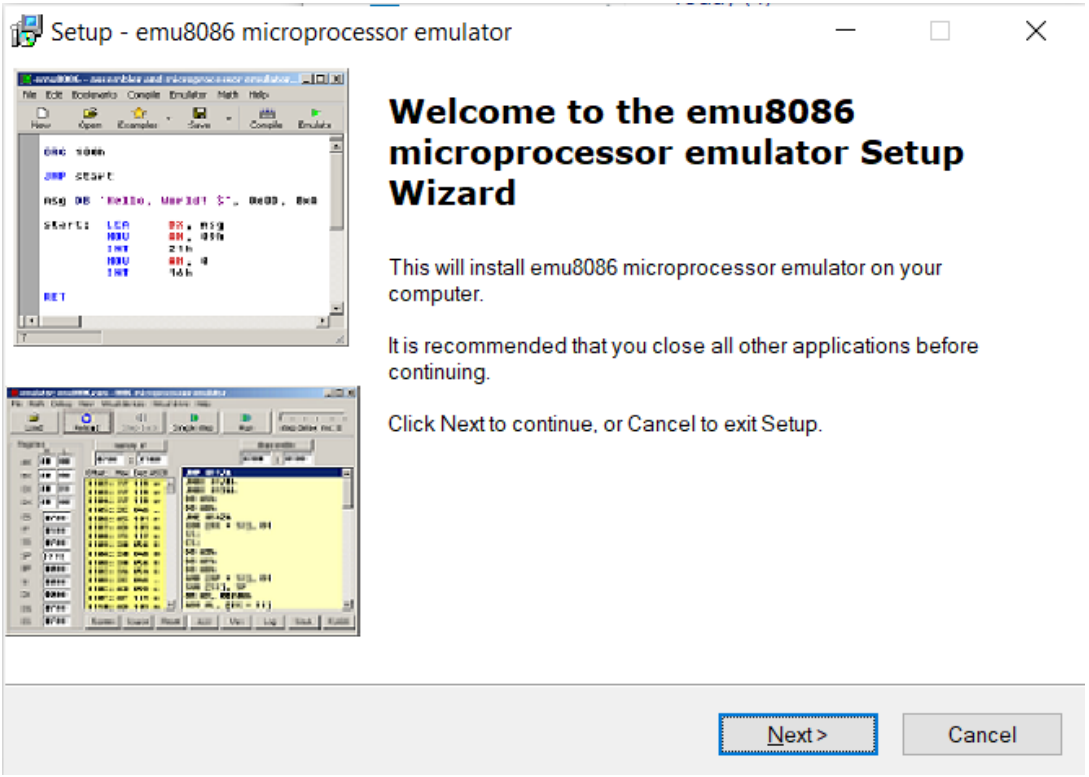
- Segment ve adres register çiftleri:
- CS:IP
- SS:SP SS:BP
- DS:BX DS:SI
- DS:DI
- ES:DI

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

- https://emu8086-microprocessor-emulator.softonic.com.tr/

adresinden emülatörü indirebilirsiniz.



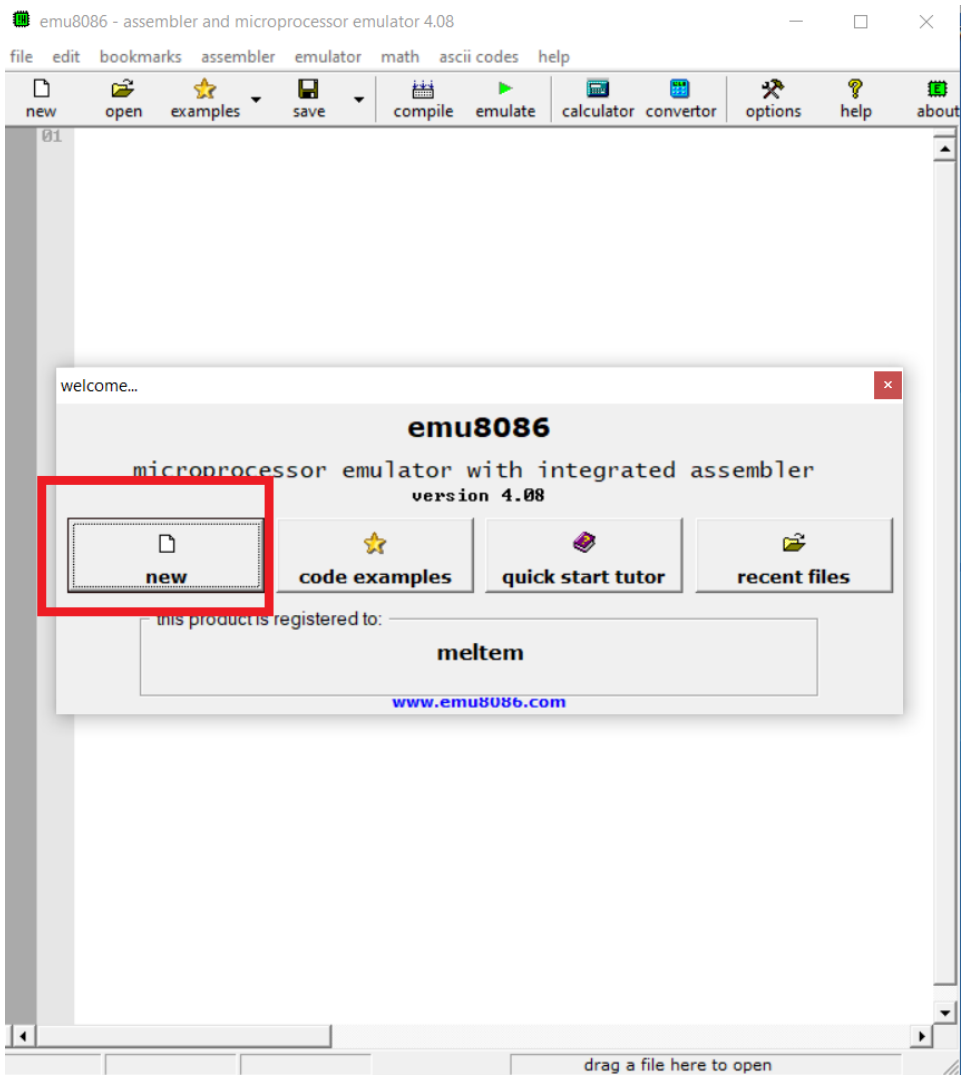
8086 16-Bit Mikroişlemci

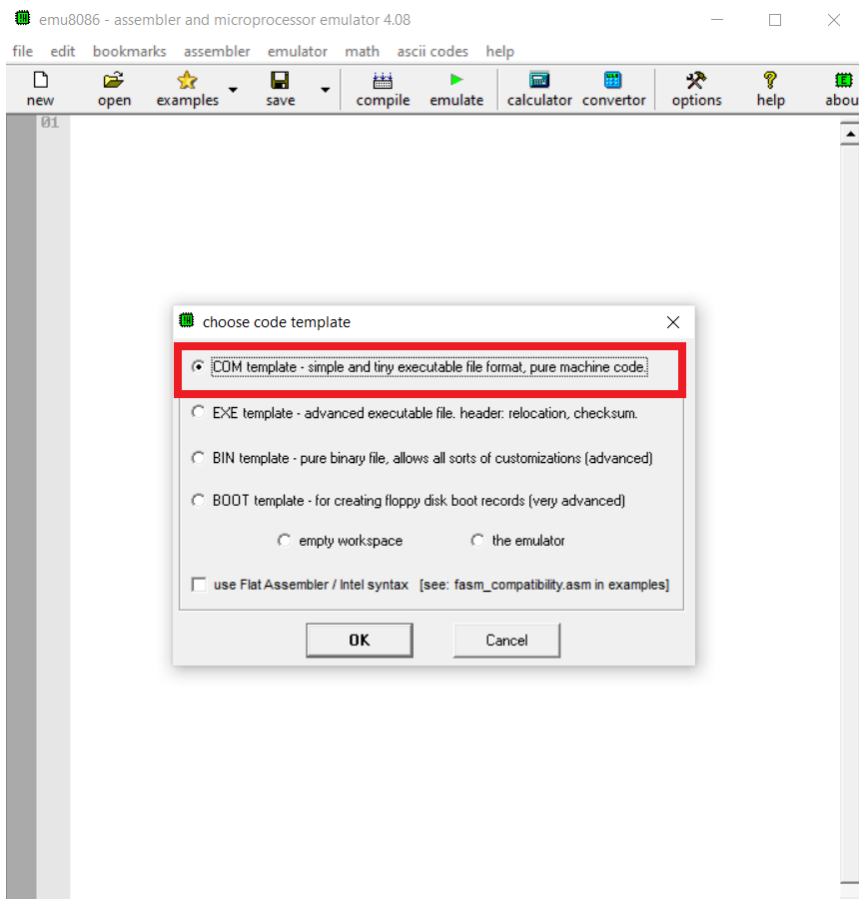
EMU 8086-MICROPROCESSOR EMULATOR

- Kurulumu tamamladıktan sonra 27R3VDEFYFX4N0VC3FRTQZX anahtarı ile aktivasyonunuzu yapın

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

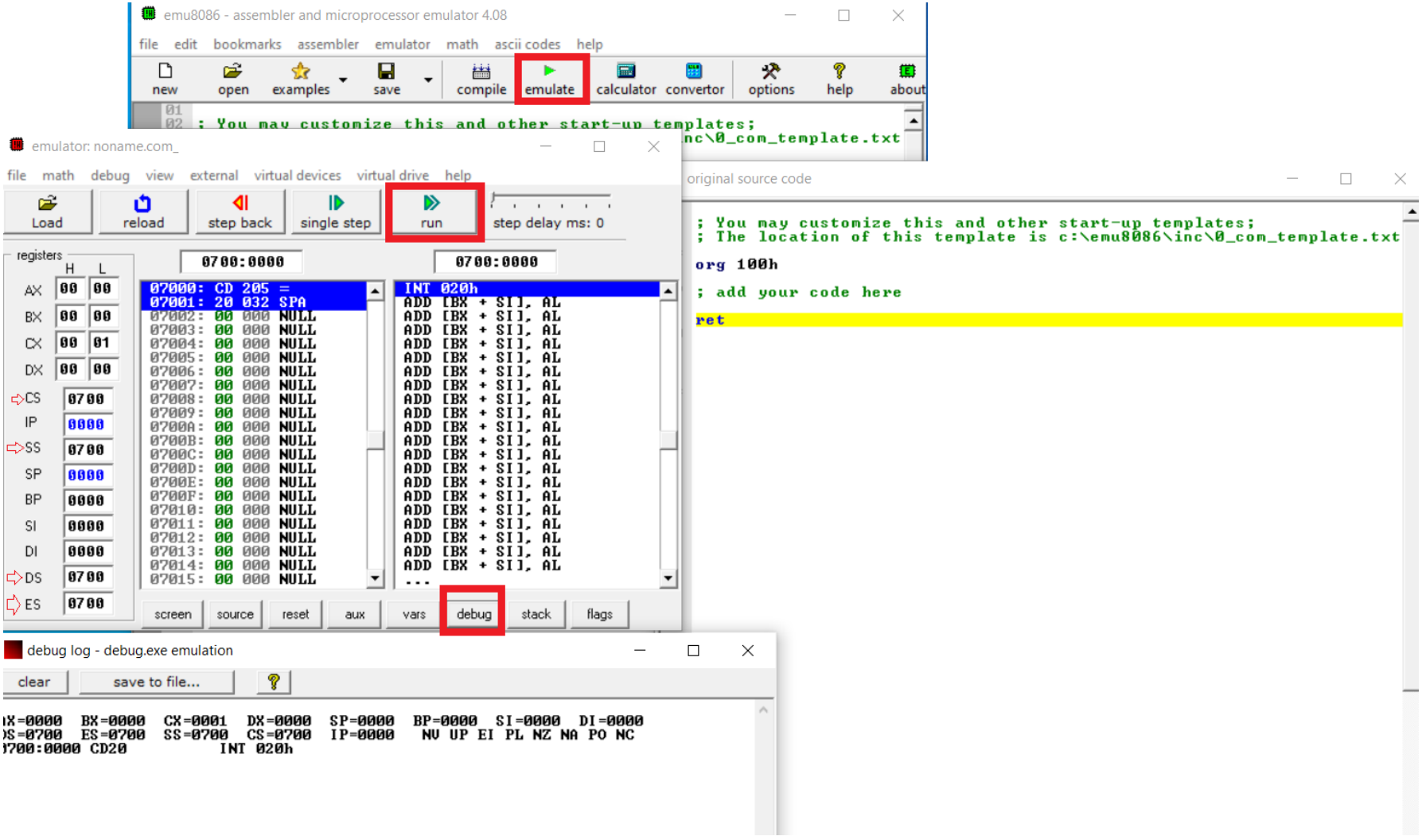




8086 16-Bit Mikroişlemci

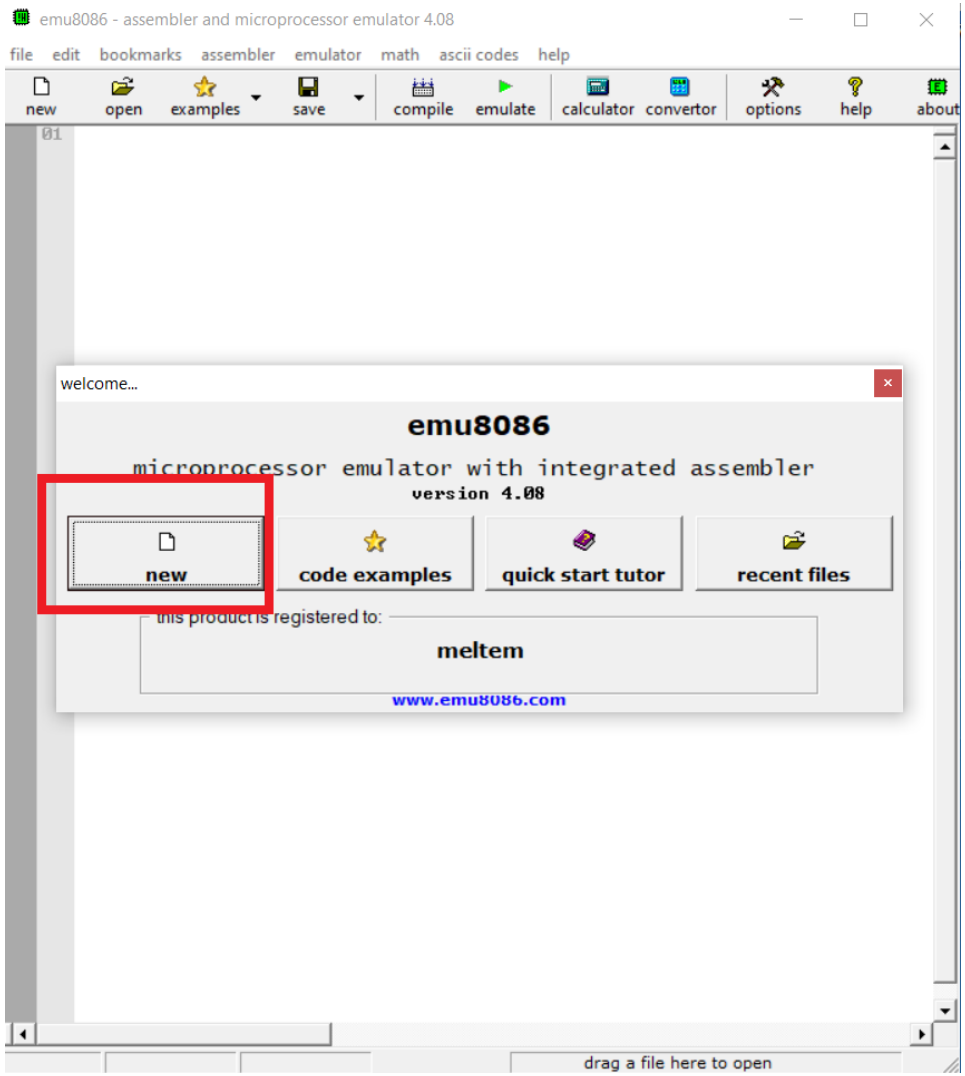
EMU 8086-MICROPROCESSOR EMULATOR

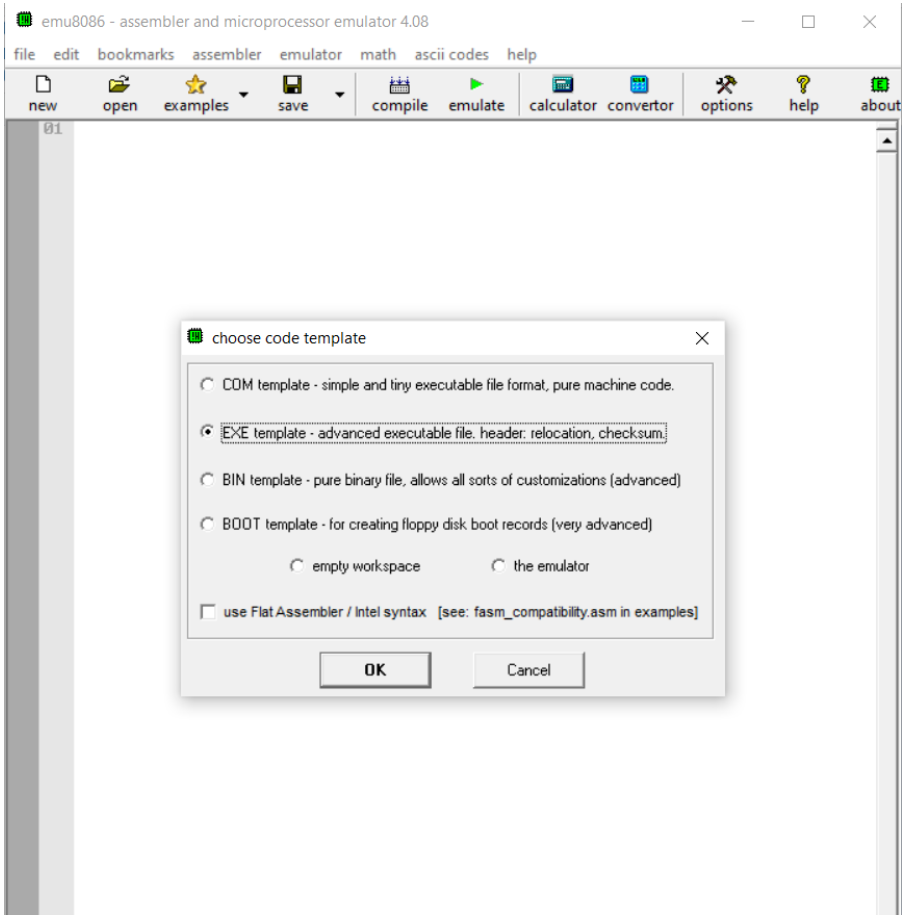
- Com template oluşturduk: 64 KB segment var tüm segment registerları aynı adresten başlıyor



8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR

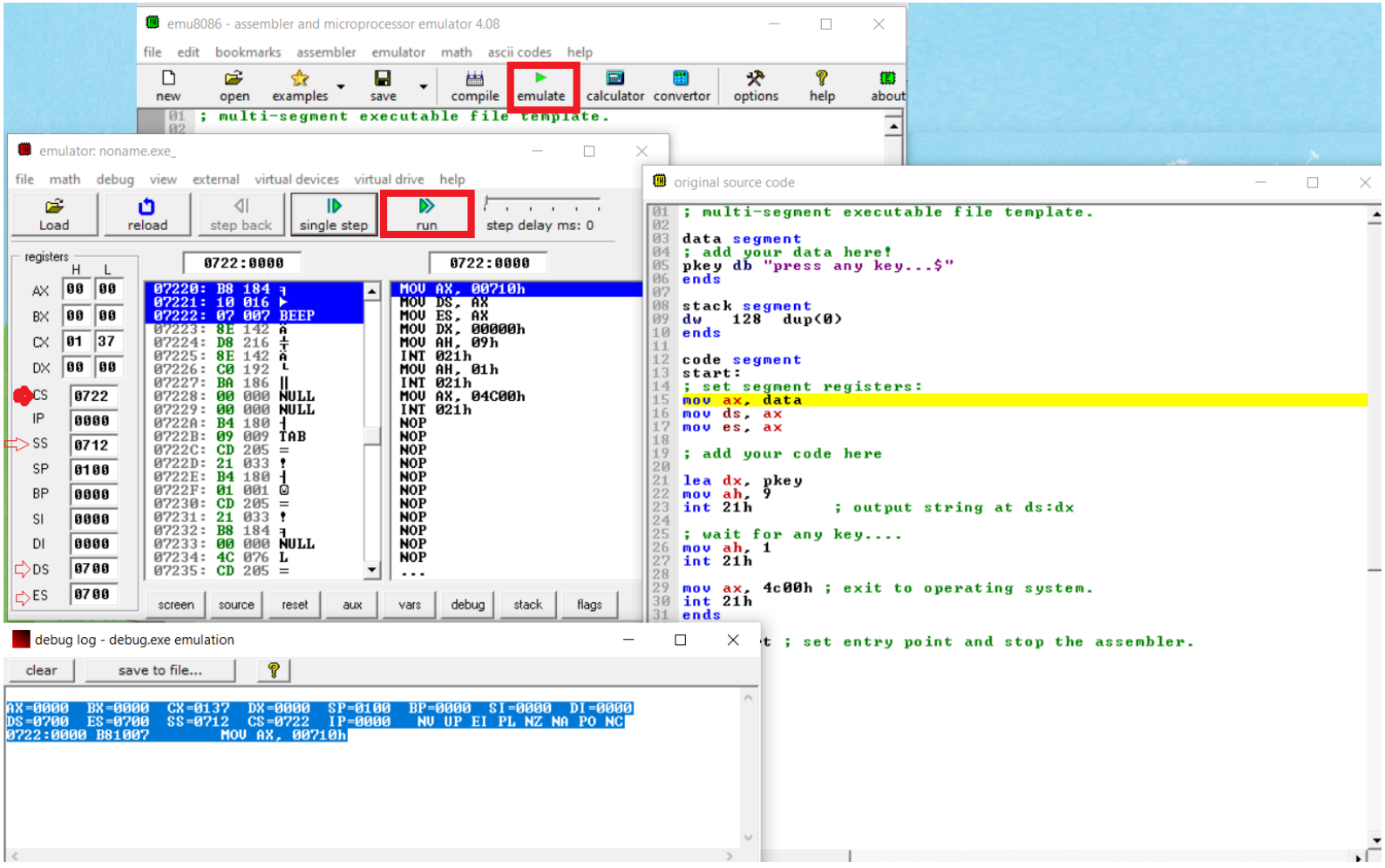




8086 16-Bit Mikroişlemci

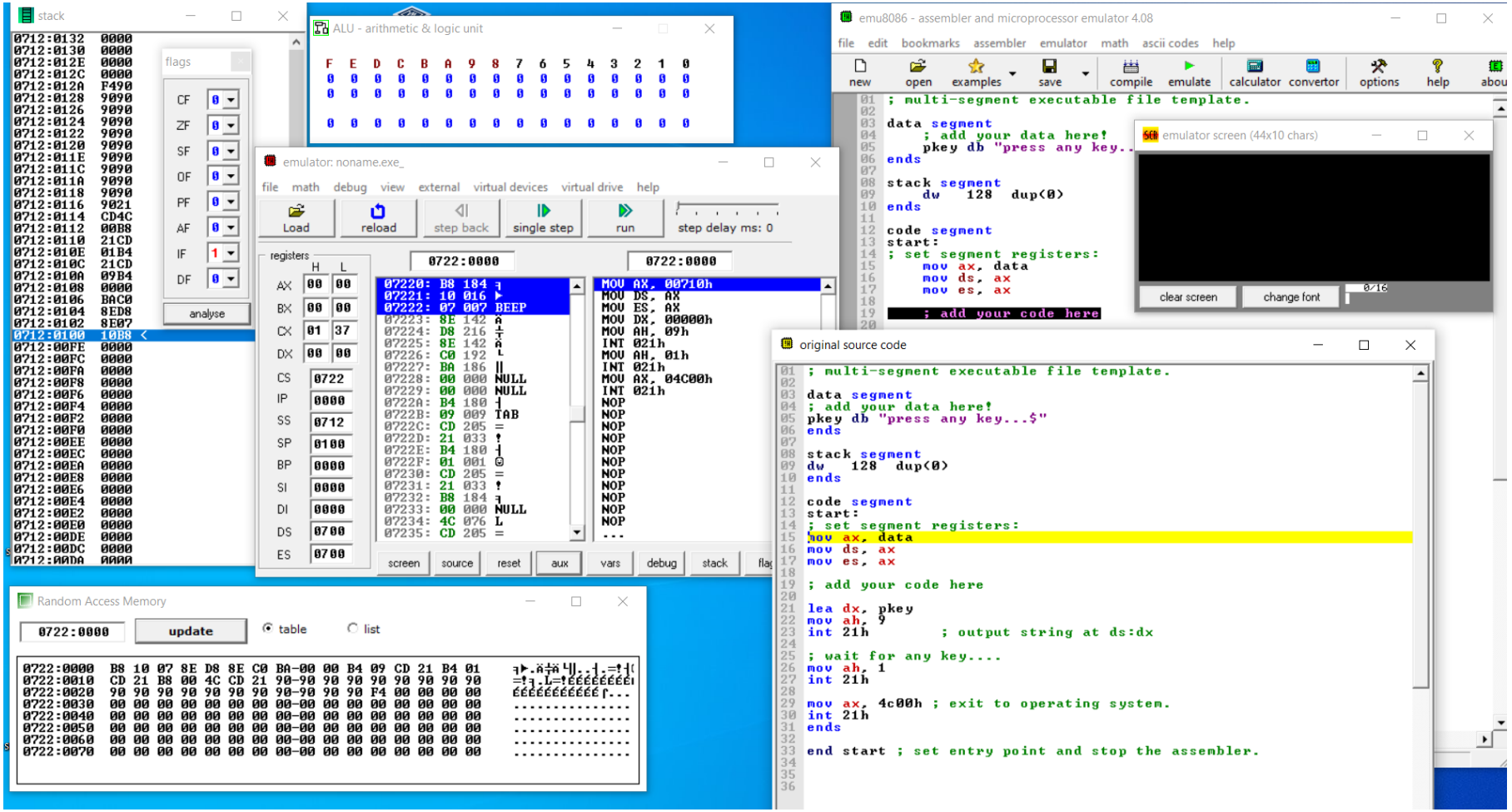
EMU 8086-MICROPROCESSOR EMULATOR

- Exe template oluşturduk: Ayrı ayrı segment registerlarının tanımlamalarını yapabiliyoruz



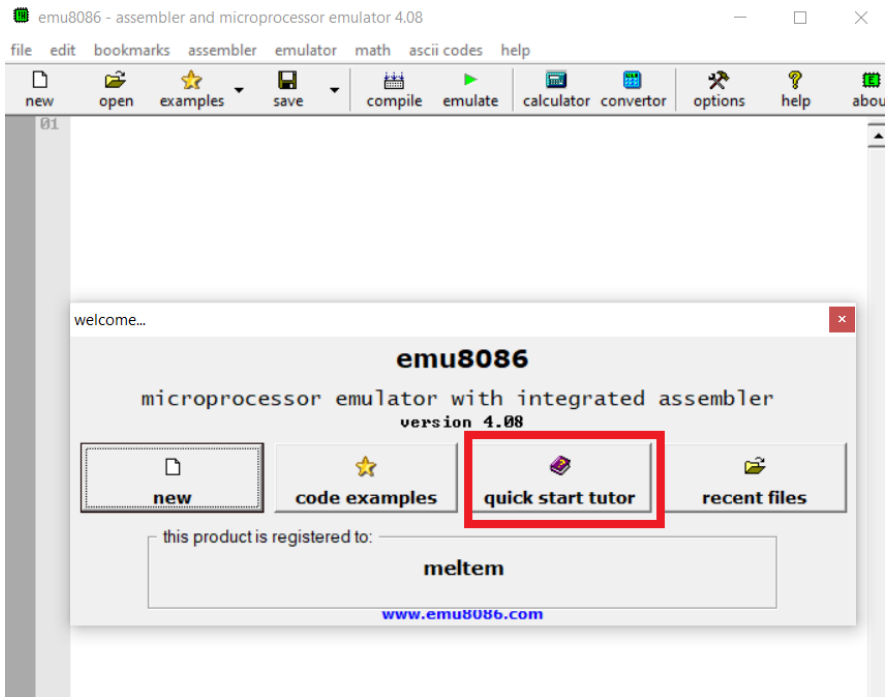
8086 16-Bit Mikroişlemci

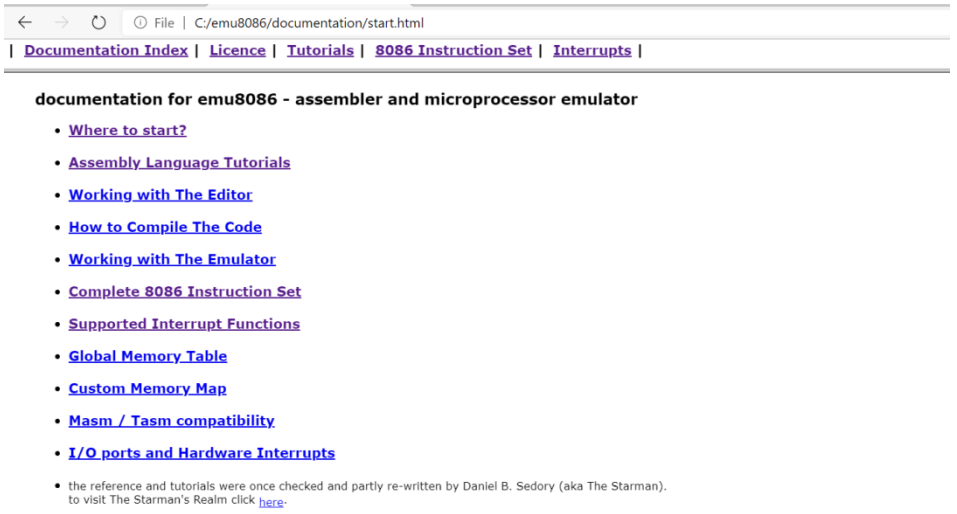
EMU 8086-MICROPROCESSOR EMULATOR



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 isletim 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(dublicate)
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 tanimlama
metin db ‘okul’
faiz equ 8   ; faiz diye bir sabit tanimliyoruz
mov ax, faiz ; bu komutu kullandigimizda AX registerina faiz degiskeninin degerini atamis oluyoruz
```

8086 16-Bit Mikroişlemci

EMU 8086-MICROPROCESSOR EMULATOR