FBU CPU

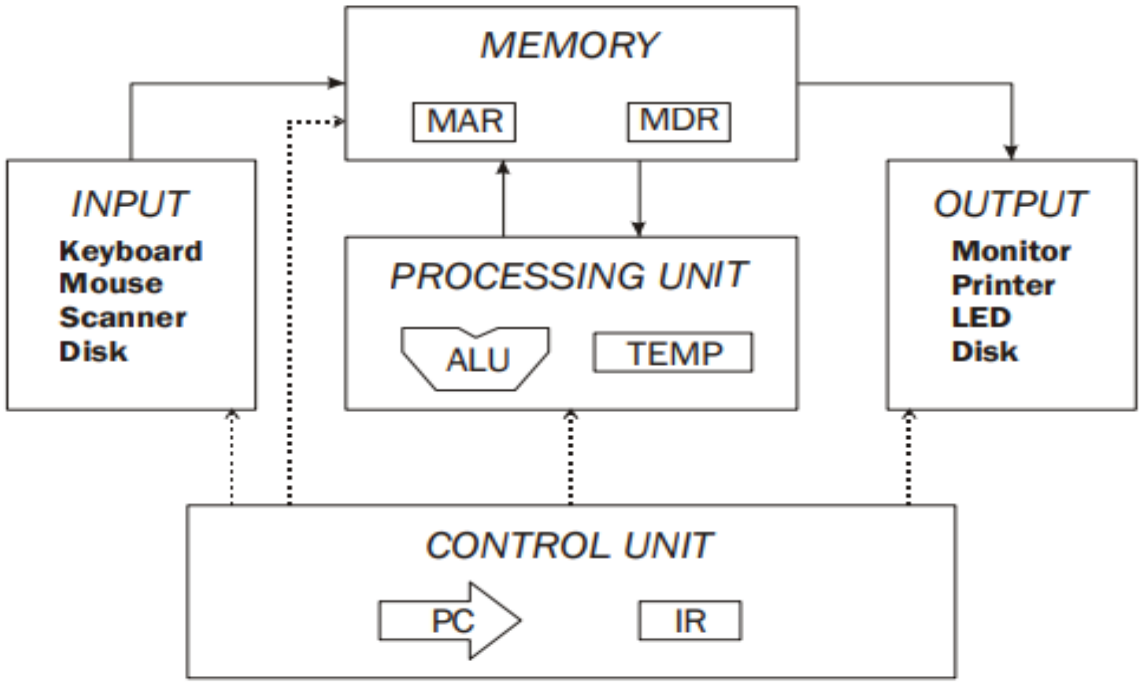
İLHAN ERSOY
HASAN DEMİR
CÜNEYT BALCI
MEHMET ÇOLAK

Proje tanımı:

- Bu proje kapsamında FB-CPU isminde bir işlemcinin tasarımı yapılacak ve tasarlanan işlemci üzerinde makine dili ile yazılan çeşitli kod parçacıkları yazılacaktır. Proje sonunda basit bir işlemcideki RAM, Kontrol Ünitesi ve Saklayıcıların bir arada çalışıp, makine dilindeki kod parçacıklarını nasıl yürütebildiği gözlemlenecektir.
- FBU-CPU Von Neumann mimarisinde bir cpu dur.

Von Neumann mimarisi Nedir?

- Temel olarak 4 elemanı vardır.
- • Saklayıcılar (Şekil 2'de Processing Unit'in altındaki Temp değişkeni)
- • Bellek (RAM)
- • İşlem Ünitesi (ALU)
- • Kontrol Ünitesi



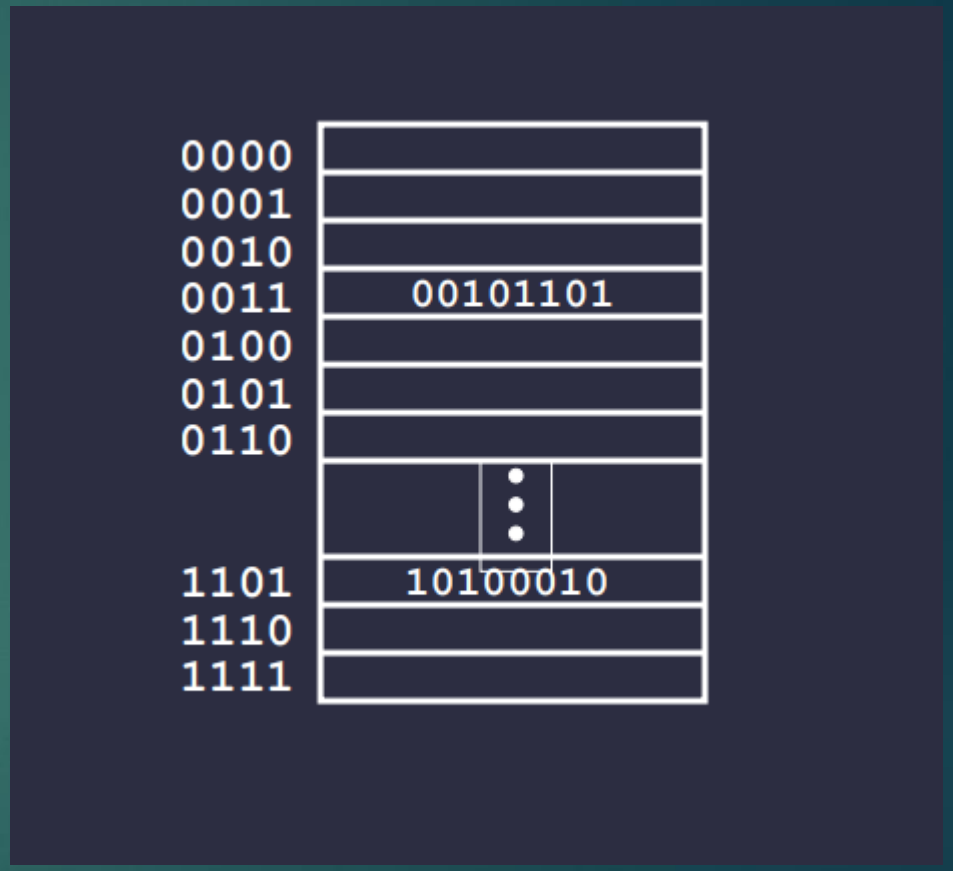
Şekil 2. Von Neumann Mimarisi

- • Von Neumann mimarisinde, kullanılan ünitelerin görevleri:
- • Bellek, operasyon komutlarını ve değişkenleri tutmaktadır.
- • İşlemci Ünitesi, aritmetik ve mantık işlemlerini yapmaktadır.
- • Kontrol Ünitesi, komutların çözülmesi için gereklidir.

Mimarinin bileşenlerini tanıyalım

BELLEK

- 2 k x m saklama alanı vardır
- Adres
  - 2^k adet farklı saklama alanı vardır.
- İçerik
  - Her bir saklama alanı, m bitliktir.
- Basit operasyonlar:
- Yükleme (Load)
  - Bellekteki bir adresten verinin okunup, bir saklayıcıya yazılmasıdır.
- Kaydetme (Store)
  - Bir saklayıcıdaki içeriğin, bellekteki bir adrese yazılmasıdır.



Belleğe Erişim

- İşlemci ünitesinin belleğe erişimi iki saklayıcı ile olmaktadır:
- MAR: Memory Address Register
- MDR: Memory Data Register
- A lokasyonundan yükleme yapmak için:
- 1. MAR saklayıcısına adresi'i yazılır .
- 2. Read sinyalini aktif edilir.
- 3. MDR'den gelen veriokunur.
- X değerini A lokasyonuna yazmak için:
- 1. X değerini MDR'e yazılır.
- 2. A değerini MAR'a yazılır.
- 3. Write sinyalini aktif edilir.

İşlemci Ünitesi

- ALU = Aritmetik ve Lojik Ünitesi
- Bir çok ünite bulunabilir

(Toplama, çarpma, NOT, AND, …)

- İçerisindeki saklayıcılarda geçici değişkenler saklanmaktadır.
- Saklayıcılarda işlem sonuçları tutulabilir.
- ACC : Accumulator, aritmetik işlem sonuçlarının tutulduğu saklayıcıdır.

Kontrol Ünitesi

Programın akışını yönetmektedir.

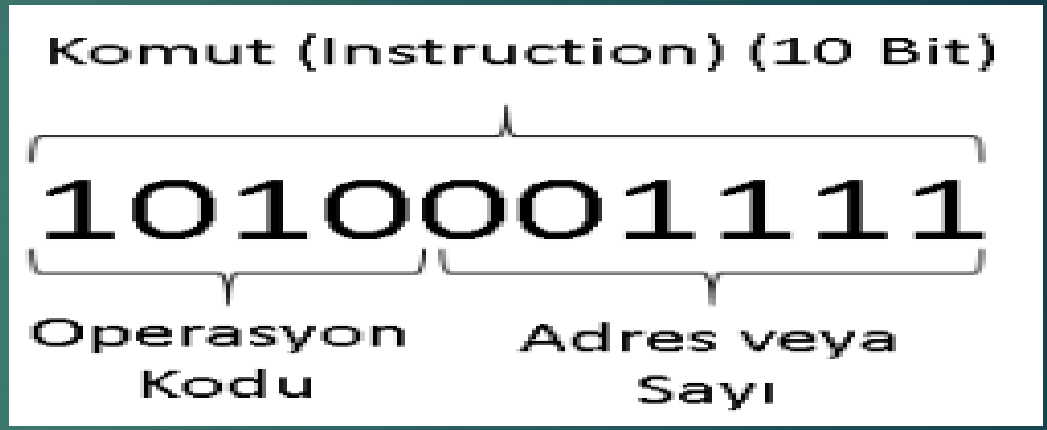
- • • Instruction Register (IR) şu anki koşturulan komutun adresini tutmaktadır.
- • Program Counter (PC) bir sonraki koşturulacak olan komutun adresini tutar.
- • Kontrol Ünitesi:
  - • Bellekten program counter'ın göstediği komutu okur.
  - • Sistemin geri kalanına, yapılması gereken işlemleri yaptırır.
  - • Bir komut birden çok cycle sürebilir.

- Saklayıcılar
- • PC (6 Bit): RAM üzerinde hangi satırdaki komutun alınacağını belirler. 6 bit olmasının nedeni RAM'in 2^6 lokasyonu olmasındandır. Dolayısıyla PC değeri RAM'deki her yeri gösterebilmektedir.
- • MAR (6 Bit): Memory Address Register isminde bir saklayıcıdır. Bu saklayıcı RAM'in adres girişine bağlanmıştır. RAM'in 2^6 lokasyonu olduğu için MAR 6 bitliktir. Saklayıcı RAM'in içerisindedir.
- • MDRIn (10 Bit): Memory Data Register In, RAM'e bir veri yazılacağı zaman kullanılan saklayıcıdır. RAM'in bir lokasyonu 10 bitlik olmasından ötürü, saklayıcı 10 bittir. Saklayıcı RAM'in içerisindedir.
- • RAMWr (1 Bit): RAM'e veri yazılacağı durumlarda aktif edilmektedir. 1 olmadığı durumlarda RAM'e veri yazılmaz. Saklayıcı RAM'in içerisindedir.
- • MDROut (10 Bit): Memory Data Register, RAM'den veri okunacağı zaman kullanılan saklayıcıdır. RAM'in bir lokasyonu 10 bit olmasından dolayı, saklayıcı 10 bittir. Saklayıcı RAM'in içerisindedir.
- • IR (10 Bit): Instruction Register, RAM'den okunan kodun (instruction) saklandığı saklayıcıdır.
- • ACC (10 Bit): Accumulator, aritmetik işlem sonuçlarının tutulduğu saklayıcıdır.

Komutların bellekten okunması ve çözümlenmesi

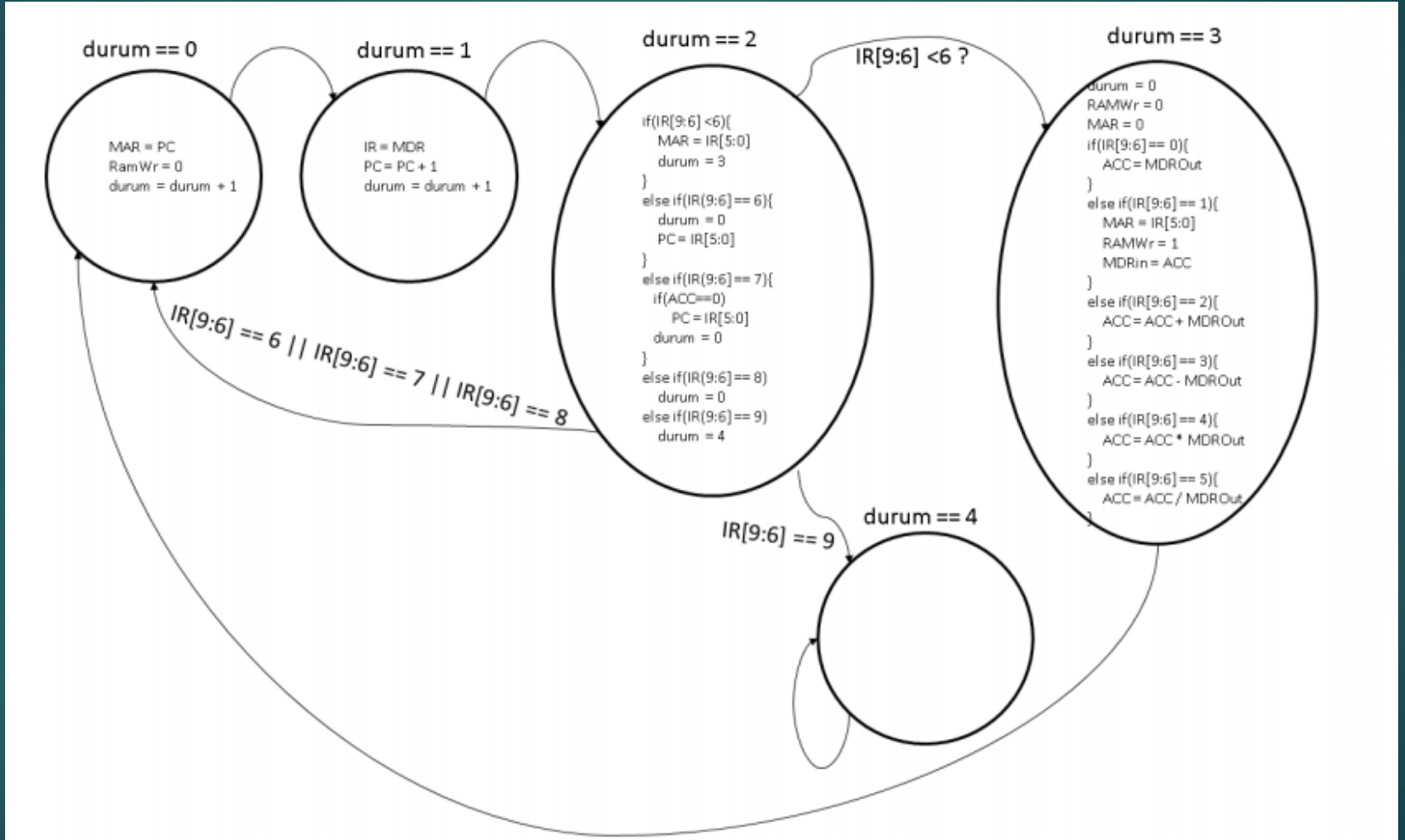
Durum makinesinde komutlar aşağıdaki kurala göre uygulanacaktır.

Komut Adı	Görevi	Operasyon Kodu
LOD ADDR	Yükleme (Load), Bellekteki verilen adresin içerisinden değeri alıp, ACC saklayıcısına yerleştirir. ACC = *(ADDR)	0000
STO ADDR	Kaydetme (Store), ACC'nin içerisindeki değeri alıp, bellekte verilen adrese yazar. *(ADDR) = ACC	0001
ADD ADDR	Bellekteki verilen adresteki değeri alır, ACC ile toplayıp, ACC'nin üzerine yazar. ACC = ACC +*(ADDR)	0010
SUB ADDR	Bellekteki verilen adresteki değeri alır, ACC ile çıkartıp, ACC'nin üzerine yazar. ACC = ACC - *(ADDR)	0011
MUL ADDR	Bellekteki verilen adresteki değeri alır, ACC ile çarpıp, ACC'nin üzerine yazar. ACC = ACC * (*(ADDR))	0100
DIV ADDR	Bellekteki verilen adresteki değeri alır, ACC ile bölüp, ACC'nin üzerine yazar. ACC = ACC / (*(ADDR))	0101
JMP SAYI	PC = Sayı olur.	0110
JMZ SAYI	ACC'ın değeri 0 ise, verilen sayı değerini PC'e atar, değilse işlem yapmaz.	0111
NOP	No Operation, hiçbir işlem yapılmaz.	1000
HLT	Uygulama durur	1001



FBU CPU PROJEMİZİN DURUM MAKİNESİ GÖSTERİMİ VERİLMİŞTİR

Bu çizim cpu nun işleyişini anlmak için önemlidir.



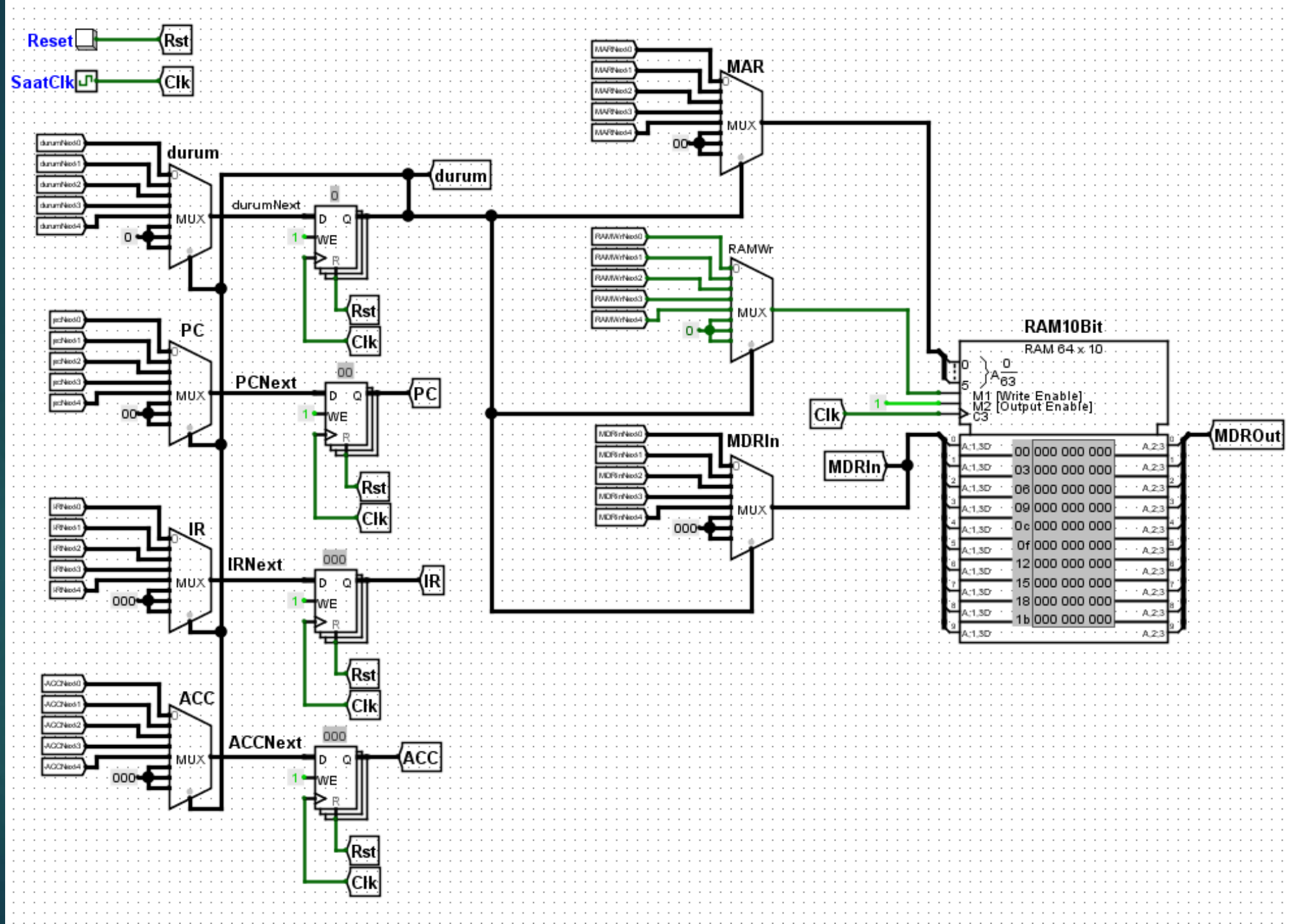
Şekil 5. FB-CPU Durum Makinası Gösterimi

- CPU durum makinasının mimarisi çizilmelidir.
- Durum makinasında kurduğumuz mantık yapıları çeşitli devre elemanlarıyla sağlanmaktadır.
- Örneğin eğer(if)- değilse eğer(else if) kısımlarını MUX(multiplexer)

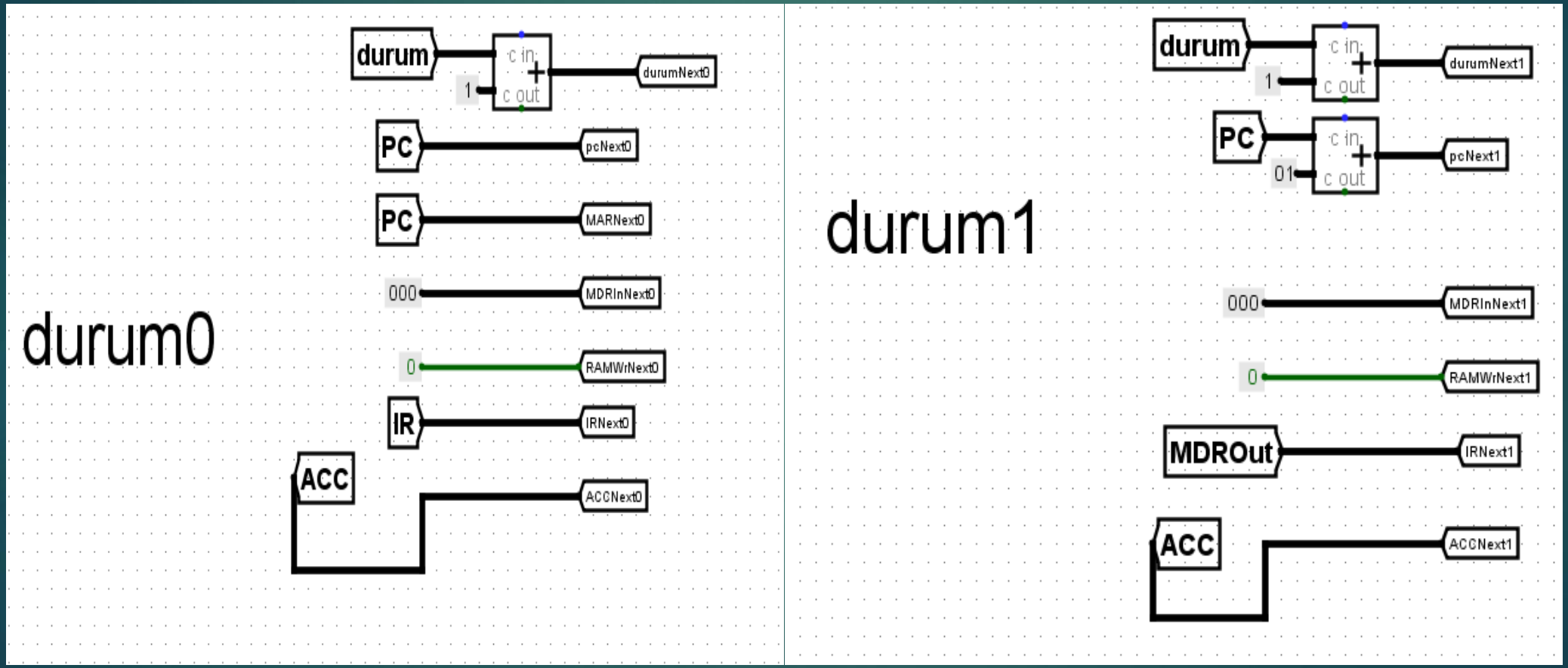
adını verdiğimiz yapılarla gerçekleştiriyoruz MUX adı verilen bu yapıların n seçme biti ve 2^n sayıda data bitleri bulunur seçme bitlerinden beslenen değere ait data biti seçilir ve operasyon tamamlanmış olur.

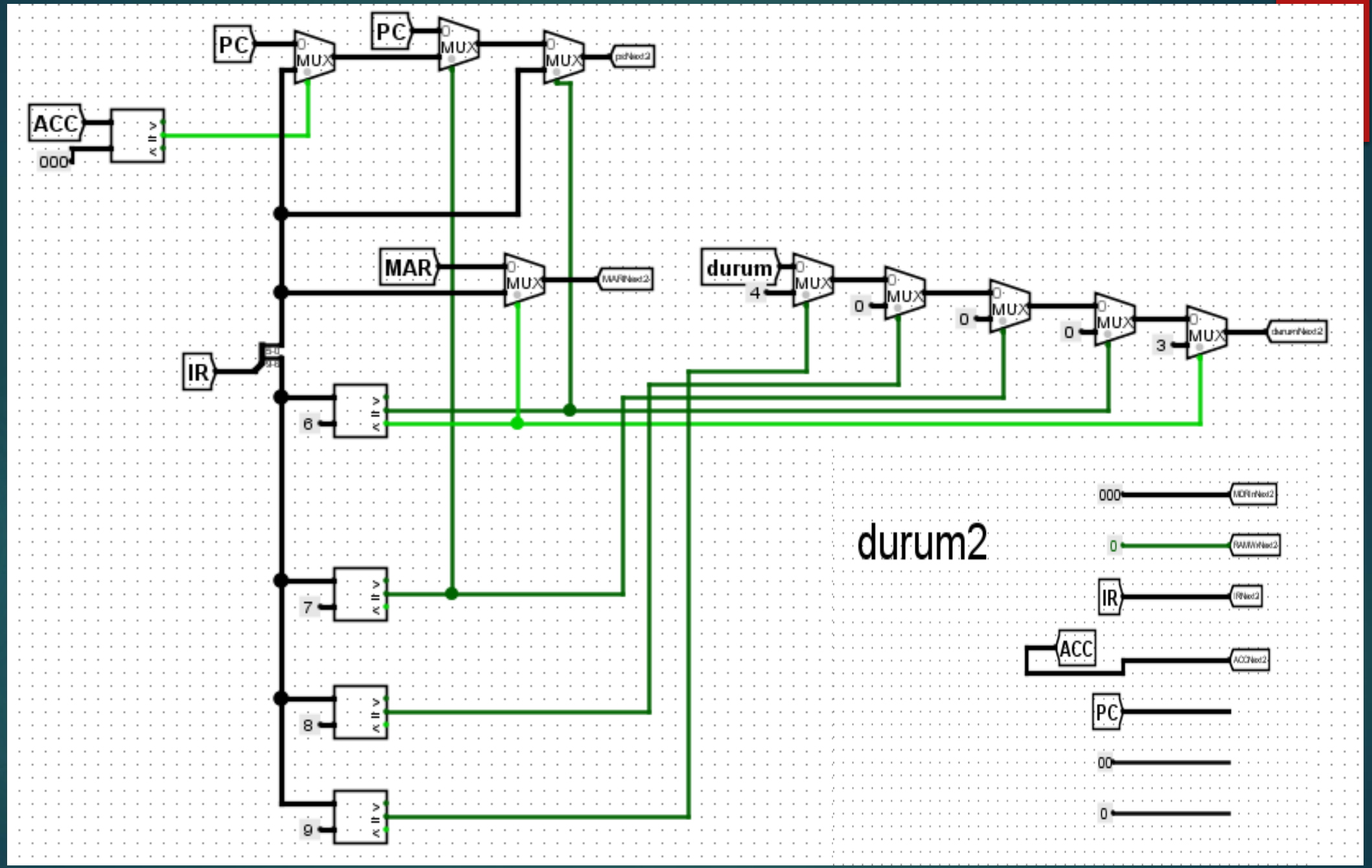
- Az sonraki slaytlarda bu durumları 1,2,3 ve 4. durum olarak parça parça inceleyeceğiz.

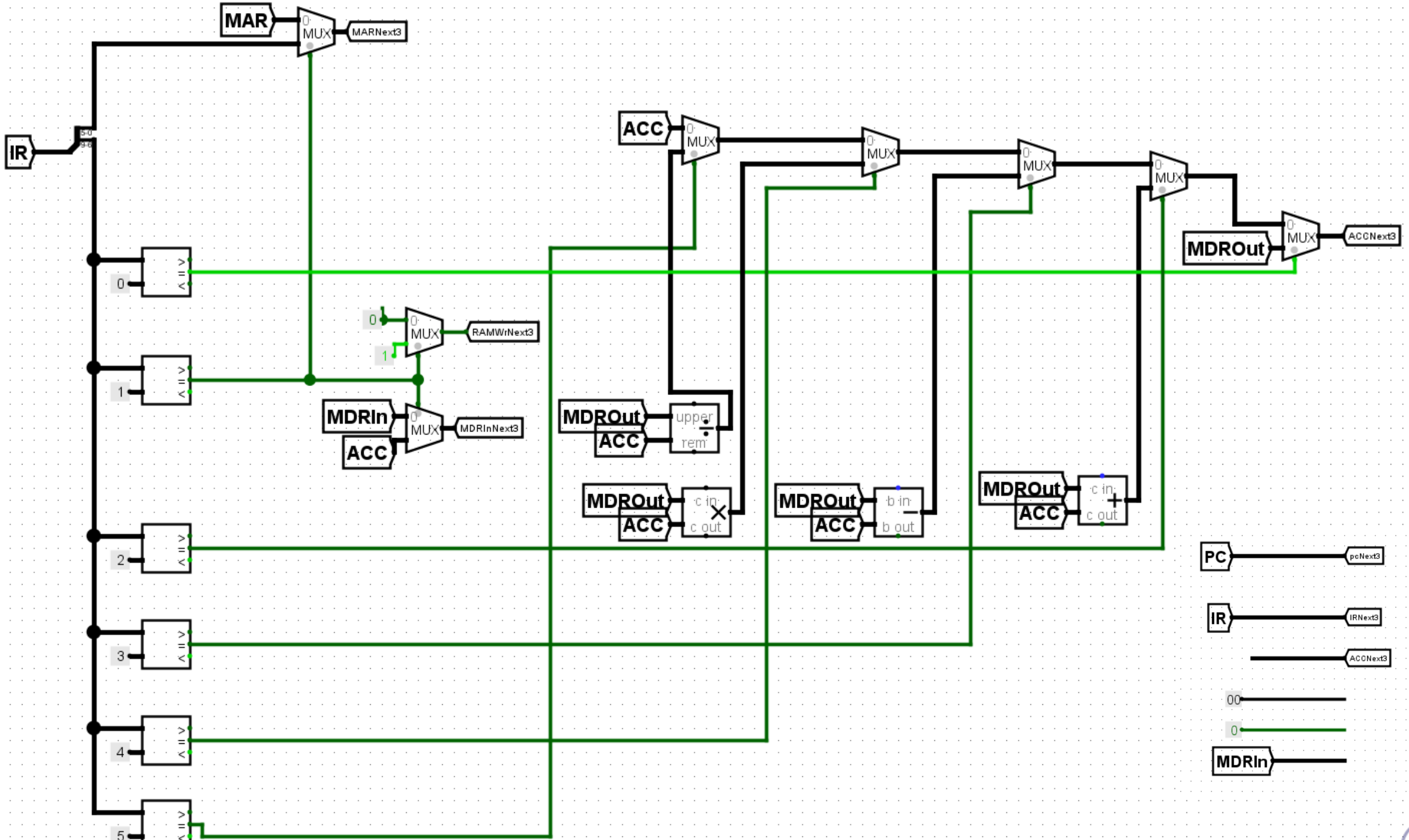
- Durumlar modellenirken tünel adını verdiğimiz simgeler kullanılmıştır bunun sebebi bir kablo karmaşasını engellemektir.
- Sonraki slaytlarda durum makinesi diyagramında ifade edilen durumlar donanım olarak ifade ediliyor.


Reset
Rst
SaatClk
Clk
durum
MUX
durumNext
D Q
WE
R
PC
PCNext
IR
IRNext
ACC
ACCNext
MAR
RAMWr
MDRIn
000
00
RAM10Bit
RAM 64 x 10
M1 [Write Enable]
M2 [Output Enable]
MDROut
00 000 000 000
03 000 000 000
06 000 000 000
09 000 000 000
0c 000 000 000
0f 000 000 000
12 000 000 000
15 000 000 000
18 000 000 000
1b 000 000 000

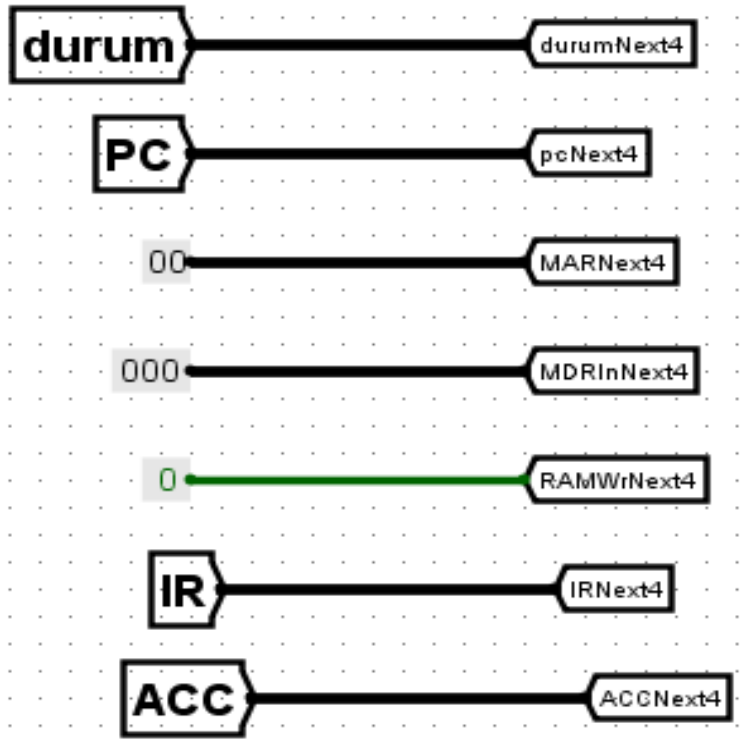
Kullanılan simülatörde durumların modellenmesi:




PC
MUX
PC
MUX
MUX
pcNext2
ACC
000
MAR
MUX
MARNext2
durum
4
MUX
0
MUX
0
MUX
0
MUX
3
MUX
durumNext2
IR
6
7
8
9
durum2
000
MDRinNext2
0
RAMWNext2
IR
IRNext2
ACC
ACCNext2
PC
00
0


MAR
MUX
MARNext3
IR
5:0
0
1
2
3
4
5
RAMWrNext3
MDRIn
ACC
MDRInNext3
ACC
MUX
MUX
MUX
MUX
MUX
MDROut
ACCNext3
MDROut
ACC
upper
rem
MDROut
ACC
c in
c out
MDROut
ACC
b in
b out
MDROut
ACC
c in
c out
PC
pcNext3
IR
IRNext3
ACCNext3
00
0
MDRIn

durum4


durum
durumNext4
PC
pcNext4
00
MARNext4
000
MDRInNext4
0
RAMWrNext4
IR
IRNext4
ACC
ACCNext4

- Tasarladığımız işlemcide 10 adet komut çalıştırılabilmektedir.
- Bu vasıtayala makine dilinde çeşitli yazılımlar işlemci üzerinde yürütülebilmektedir.
- Örneğin
- FB-CPU için bellekte 50 ve 51 adresteki iki sayının toplamını 52 no'lu adrese kaydeden uygulamayı inceleyelim
- 0: 0000_110010 // LOD 50, (ACC = *50), Hex = 32
- 1: 0010_110011 // ADD 51, ACC = ACC + (*51), Hex = B3
- 2: 0001_110100 // STO 52, (*52) = ACC, Hex = 74
- 3: 1001_000000 // Halt, Hex = 240
- 50: 0000000101 // Hex = 5
- 51: 0000001010 // Hex = A