```
`timescale 1ns / 1ps


 module islemci(
   input clk,   //burada ödevde verildiği gibi giriş çıkışları belirliyoruz.
   input rst,
   input  [31:0] buyruk,
   output  reg [31:0] ps
    );
   reg [31:0] yazmac_obegi[255:0];    //ödevde istendiği gibi yazmaç öbeğini ve veri belleğini reg ile
//tanımlıyoruz
   reg [31:0] veri_bellek[127:0];
   reg [31:0] immDuzenleme; //immDuzenleme diye reg tanıdım bazı buyrukların anlık değerleri
//karmaşık gelebiliyor onların anlık değerlerini düzenlemek için


   initial begin
      ps = 0;          //ilk başta ps=0 başlıyor yazmaç obeği[0] her zaman 0 değerine sahip  başlangıçta
//atadım.
      yazmac_obegi[0]=0;
      immDuzenleme[31:0]=0;


   end


   always@(posedge clk or posedge rst) begin
   if(rst)begin
     ps<=0;    //rst=1 iken ps en başa dönmeli diğer değerler de 0 olmalı.
     yazmac_obegi[0]<=0;
     immDuzenleme[31:0]<=0;
   end
   else begin
   ////////////////////////////////////////////////////////////////
      ps<=ps+4;              //rst=0 olduğu zaman ps 4 arttırdım ki bir sonraki buyruğun yerini göstersin.
      if(buyruk[6:0]==51)begin //son 7 bitin değeri yani opcode değeri 51 ise
```

```
        if(buyruk[14:12]==0)begin      //işkodu3 0 ise
          if(buyruk[30]==0)begin//add   //ve buyruktaki 30. Eleman 0 ise toplama yapılacak.
           yazmac_obegi[buyruk[9:7]]<=yazmac_obegi[buyruk[17:15] +yazmac_obegi[buyruk[22:20]];
         //iki yazmacı topla hedef yazmacına yaz 32 bitlik veriler
           yazmac_obegi[0]=0; //eğer hedef yazmacı yazmacobegi[0] ise 0 olsun çünkü hep 0 olmalı
           ps<=ps+4;   //sonraki buyruğa geç çevrimi bitir ps 4 arttır.
          end
          else if(buyruk[30]==1)begin // önceki if koşullarıyla birlikte buyruktaki 30. Eleman 0 ise
//çıkarma yapılacak.

            yazmac_obegi[3]<=yazmac_obegi[buyruk[17:15]]-yazmac_obegi[buyruk[22:20]]; //iki
//kaynak yazmaçlarını birbirinden çıkar hedefe yaz.
            yazmac_obegi[0]=0; //eğer hedef yazmacı yazmacobegi[0] ise 0 olsun çünkü hep 0 olmalı

            ps<=ps+4; ;   //sonraki buyruğa geç çevrimi bitir ps 4 arttır.

          end
        end
        else if(buyruk[14:12]==4)begin//xor// opcode=51 ise ve işkodu3 4 ise xor yap
          yazmac_obegi[buyruk[9:7]]<=yazmac_obegi[buyruk[17:15]]^yazmac_obegi[buyruk[22:20]];
//kaynak yazmacını birbirleriyle xorla hedefe yaz.

          yazmac_obegi[0]=0; //bu yukarıdaki sebeple aynı bundan sonra yazmaya gerek yok. Yazmaç
//obegi[0] hedef yazmacı olabilir bu durumda değeri değişmemesi 0 olması gerekir.
           ps<=ps+4; ;   //sonraki buyruğa geç çevrimi bitir ps 4 arttır.

        end
        else if(buyruk[14:12]==7)begin//and işkodu 7 ise and yap
         yazmac_obegi[buyruk[9:7]]<=yazmac_obegi[buyruk[17:15]]&yazmac_obegi[buyruk[22:20]];
//iki kaynak yazmacını andle hedefe yaz.
          yazmac_obegi[0]=0;
          ps<=ps+4;
```

```
        end
        else if(buyruk[14:12]==6)begin//or //aynı mantık or yapıyoruz bu seferde.
          yazmac_obegi[buyruk[9:7]]<=yazmac_obegi[buyruk[17:15]]| yazmac_obegi[buyruk[22:20]];
          yazmac_obegi[0]=0;
           ps<=ps+4;
        end
      end
      ////////////////////////////////////////////////
      else if(buyruk[6:0]==19)begin
        if(buyruk[14:12]==0)begin//addi
          if(buyruk[31]==0)begin //opcode 19 ise işkodu3 0 ise addi işlemi yapıyoruz anlıkla rs1
//toplayıp hedefe yazıyoruz.

            immDuzenleme[31:12]=0; //burada işaretle genişletiyoruz anlığın son bit 0 ise 0 la değilse
//hepsini 1 yapıyoruz
          end
          else if(buyruk[31]==1)begin
            immDuzenleme[31:12]=1048575;
          end
          immDuzenleme[11:0]=buyruk[31:20]; //buyruğun son 12 biti anlık immDüzenlemeye
//atıyoruz en değerli 20 biti yukarıda işaretle genişletiyoruz.

          yazmac_obegi[0]=0;
            yazmac_obegi[buyruk[9:7]]<=yazmac_obegi[buyruk[17:15]] + immDuzenleme[31:0];
//anlıkla topluyoruz rs1i hedef yazmacına yazıyoruz.
            ps<=ps+4;
        end
      end
      ////////////////////////////////////////////////
      else if(buyruk[6:0]==103)begin//jalr //opcode 103se jalr işlemi yapıyoruz. Ps+4 kaynağa
//yazıyoruz anlığı düzenliyoruz. Ps'na anlıkla hedefi toplayıp veriyoruz anlık işaretle genişletiliyor.
        yazmac_obegi[buyruk[9:7]]<=ps+4;
```

```
        if(buyruk[31]==0)begin
            immDuzenleme[31:12]=0;
        end
        else if(buyruk[31]==1)begin
            immDuzenleme[31:12]=1048575;
        end
        immDuzenleme[11:0]=buyruk[31:20];
        ps<=yazmac_obegi[buyruk[17:15]]+immDuzenleme[31:0];
    end
    ////////////////////////////////////////////////////
    else if(buyruk[6:0]==111)begin//jal //opcode 111 ise jal işlemi uygulanıyor. Anlık işaretle
//genişletiliyor(en değerli bitine göre). PS+4ü  yazmac_obegi[buyruk[9:7]]'e veriyoruz.
        if(buyruk[31]==0)begin
            immDuzenleme[31:21]=0;
        end
        else if(buyruk[31]==1)begin
            immDuzenleme[31:21]=2047;
        end
        immDuzenleme[20]=buyruk[31];
        immDuzenleme[10:1]=buyruk[30:21];
        immDuzenleme[11]=buyruk[20];
        immDuzenleme[19:12]=buyruk[19:12];
        immDuzenleme[0]=0;
        yazmac_obegi[buyruk[9:7]]<=ps+4;
        yazmac_obegi[0]=0;
        ps<=ps+immDuzenleme[31:0]; ps anlıkla toplanıyor atlıyor.
    end
    ////////////////////////////////////////////////////
    else if(buyruk[6:0]==99)begin
        if(buyruk[14:12]==0)begin//beq //iki koşul da geçerliyse beq işlemi oluyor iki yazmaç değeri
//eşitse ps'na anlığı veriyoruz değilse ps 4 arttırıyoruz. Anlığı düzenleyip immDüzenlemeye atıyoruz.
            if(yazmac_obegi[buyruk[17:15]]==yazmac_obegi[buyruk[22:20]])begin
```

```
      if(buyruk[31]==0)begin
          immDuzenleme[31:13]=0;
      end
      else if(buyruk[31]==1)begin
          immDuzenleme[31:13]=524287;
      end
      immDuzenleme[31:13]=0;
      immDuzenleme[12]=buyruk[31];
      immDuzenleme[10:5]=buyruk[30:25];
      immDuzenleme[11]=buyruk[7];
      immDuzenleme[4:1]=buyruk[11:8];
      immDuzenleme[0]=0;
      ps<=immDuzenleme[31:0];
    end
    else begin
       ps<=ps+4;
    end
end
//////////////////////////////////////////
else if(buyruk[14:12]==1)begin//bne // bu durum yukarıdakinin tam tersi.
    if(yazmac_obegi[buyruk[17:15]]==yazmac_obegi[buyruk[22:20]])begin
      ps<=ps+4;
    end
    else begin
      if(buyruk[31]==0)begin
       immDuzenleme[31:13]=0;
      end
      else if(buyruk[31]==1)begin
       immDuzenleme[31:13]=524287;
      end
      immDuzenleme[12]=buyruk[31];
```

```
            immDuzenleme[10:5]=buyruk[30:25];
            immDuzenleme[11]=buyruk[7];
            immDuzenleme[4:1]=buyruk[11:8];
            immDuzenleme[0]=0;
            ps<=immDuzenleme[31:0];
          end
        end
        ////////////////////////////////////////
        else if(buyruk[14:12]==5)begin//blt
          if(yazmac_obegi[buyruk[17:15]]<yazmac_obegi[buyruk[22:20]])begin //küçük eşitse anlık
//yapıyoruz yoksa ps 4 arttırıyoruz. Anlık yine düzenlenip immDüzenlemeye atılıyor.
            if(buyruk[31]==0)begin
              immDuzenleme[31:13]=0;
             end
            else if(buyruk[31]==1)begin
              immDuzenleme[31:13]=524287;
            end
            immDuzenleme[31:13]=0;
            immDuzenleme[12]=buyruk[31];
            immDuzenleme[10:5]=buyruk[30:25];
            immDuzenleme[11]=buyruk[7];
            immDuzenleme[4:1]=buyruk[11:8];
            immDuzenleme[0]=0;
            ps<=immDuzenleme[31:0];
          end
          else begin
            ps<=ps+4;
          end
        end
      end
      ////////////////////////////////////////////
```

```
      else if(buyruk[6:0]==55)begin//lui //opcode 55 ise immDüzenleniyor. İlk 12 bit 0 oluyor.
         immDuzenleme[31:12]=buyruk[31:12];
         immDuzenleme[11:0]=0;
         yazmac_obegi[buyruk[9:7]]<=immDuzenleme[31:0]; //anlığın genişletilmiş halini yazmaca
//atıyoruz.
         yazmac_obegi[0]=0;
         ps<=ps+4;
      end
      ////////////////////////////////////////////////
      else if(buyruk[6:0]==23)begin//auipc
         immDuzenleme[31:12]=buyruk[31:12];
         immDuzenleme[11:0]=0;
         yazmac_obegi[buyruk[9:7]]=immDuzenleme[31:0]+ps; //ps ile anlığın genişletilmiş halini
//toplayıp yazmaca atıyoruz.
         yazmac_obegi[0]=0;
         ps<=ps+4;
      end
      ////////////////////////////////////////////////////
      else if(buyruk[6:0]==3)begin//lw // anlığı genişletip kaynak yazmacıyla toplayıp adres buluyoruz.
//Sonra belleğe adresi verip gelen veriyi hedef yazmacına yazıyoruz.
         if(buyruk[31]==0)begin
            immDuzenleme[31:12]=0;
         end
         else if(buyruk[31]==1)begin
            immDuzenleme[31:12]=1048575;
         end
         immDuzenleme[11:0]=buyruk[31:20];

         if(immDuzenleme[31:0]+yazmac_obegi[buyruk[17:15]]%4==0) begin//4e bölünmüyorsa adresi
//göstermiyor demektir.

yazmac_obegi[buyruk[9:7]]<=veri_bellek[(immDuzenleme[31:0]+yazmac_obegi[buyruk[17:15]])>>2];
//2 kaydırmamızın nedeni adresler 4 erli şekilde sıralı gidiyor 0. Adresten sonra 4 geliyor.
```

```
        end
        yazmac_obegi[0]=0;
        ps<=ps+4;
    end
    ////////////////////////////////////////////////
    else if(buyruk[6:0]==35)begin//sw //anlığın genişletilmiş hali ile kaynak yazmacını toplayıp veri
//belleğindeki adresi buluyoruz. Diğer kaynakla da bulduğuımuz adrese veri yolluyoruz.
        if(buyruk[31]==0)begin
           immDuzenleme[31:12]=0;
        end
        else if(buyruk[31]==1)begin
           immDuzenleme[31:12]=1048575;
        end
        immDuzenleme[11:5]=buyruk[31:25];
        immDuzenleme[4:0]=buyruk[11:7];
        if((immDuzenleme[31:0]+yazmac_obegi[buyruk[17:15]])%4==0) begin
           //4e bölünmüyorsa adresi //göstermiyor demektir.
veri_bellek[(immDuzenleme[31:0]+yazmac_obegi[buyruk[17:15]])>>2]<=yazmac_obegi[buyruk[22:20
]];
//2 kaydırmamızın nedeni adresler 4 erli şekilde sıralı gidiyor 0. Adresten sonra 4 geliyor.

        end
        yazmac_obegi[0]=0;
        ps<=ps+4;
    end

  end

  end
```

```
endmodule
```

//

B şıkkında sadece buyruk belleğini yaptım.

```
`timescale 1ns / 1ps

module buyruk_bellegi(
   input clk,       //inputlar ve output soruda verildiği şekilde tasarlandı.
   input rst,
   input [31:0] adres,
   output reg [31:0] veri
   );

   reg [31:0] buyruk_bellek[31:0]; // buyruk bellek soruda verildiği şekilde tasarlandı.




   islemci uut(.clk(clk),.rst(rst),.ps(adres),.buyruk(veri)); //buyruk belleğine işlemciden gelen modülü
//bağlıyoruz.



   always@(posedge clk or posedge rst) begin
   if(rst)begin
     ////
   end
   else begin
```

```
      veri<=buyruk_bellek[adres];  // adresi ps'na bağladık ve buyruk belleğine verdik veri geldi ve onu
//da çağırdığımız modülde buyruğa bağladık.
   end
   end


endmodule
```