Fenerbahçe Üniversitesi

BLM 201 – Mantıksal Sistem Tasarımı

FB-CPU RTL Tasarımı

Proje İçeriği

Veriliş Tarihi: 7.12.2020

Teslim Tarihi ve Yeri 15.01.2020, Ders Saatlerinde, Ders Sınıfında ve Elektronik olarak

1. Tanım:

Bu proje kapsamında FB-CPU isminde bir işlemcinin Verilog dili ile RTL tasarımı 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. Kullanılacak Basys3 FPGA geliştirme kartı üzerinde FBCPU demo'su yapılacaktır.

2. Proje Ekibi:

Proje 4 kişilik ekiplerden oluşacaktır. Her bir proje ekibinin bir sorumlusu olacaktır. Öğrenciler 4 kişilik kendi proje ekiplerini ve proje sorumlusunu belirlemelidirler.

Ekiplerin kurulması ve proje sorumlusunun belirlenmesi en geç **18.12.2020** tarihine kadar tamamlanmalıdır. Ekip sorumluları, Teams'te açılmış olan "Proje Ekip Sorumluların Takımlarını Bildirmesi" başlığının altına, ekip üyelerinin isimlerini göndermelidirler.

3. Proje LAB'ı:

Proje'nin bir kısmının gerçeklemesinin nasıl olabileceği LAB esnasında yapılacaktır.

FB-CPU Verilog RTL dili ile tasarımı LAB'ı **18.12.2020** tarihinde yapılacaktır. Bu tarihe kadar FB-CPU'u gerçeklemeye çalışarak sorular biriktirilmelidir.

4. Kullanılacak Araçlar:

Proje kapsamında 2 araç kullanılacaktır.

4.1. FBCPU Simulatörü:

FB-CPU'nun mimarisini görselleştiren, veri akışının gözlemlenebildiği "FBCPU Simulatörü" kullanılacaktır.

Erişim adresi: http://levent.tc/files/courses/tools/vonneumann/

Ekranın sağ tarafında bulunan Komutlar ve Değişkenler belleğin (RAM) içerisinde bulunan sayılardır.

Komutları 0-1'lar halinde yazmak yerine, assembly denen bir dil ile ifade ediyoruz.

Komutlar bölümüne, bu işlemci için geçerli komut seti ile komutlar yazabilirsiniz.

Bu simulator aracılığı ile işlemcinin ihtiyaç duyduğu makine dili uygulamayı gerçekleyip test edebilirsiniz.

4.2. Xilinx Vivado Design Suite

Xilinx Vivado Design Suite, FPGA geliştirme kartları üzerinde çalışmalar yapmak için gerekli olan tasarımı oluşturmak için kullanılmaktadır. Verilog, VHDL vb.. donanım tasarım dillerini alarak, FPGA'e konfigüre edilebilecek (Xilinx firması FPGA'leri için .bit uzantılı dosyalar) tasarım dosyasını oluşturur.

İndirme Adresi: https://www.xilinx.com/support/download.html

Kurulum ve Lisanslama Video'su: https://www.youtube.com/watch?v=yW1bJbXnbRU

5. Tasarım Gereksinimleri

Bu başlık altında tasarlanması istenen FB-CPU'nun gereksinimleri verilmektedir.

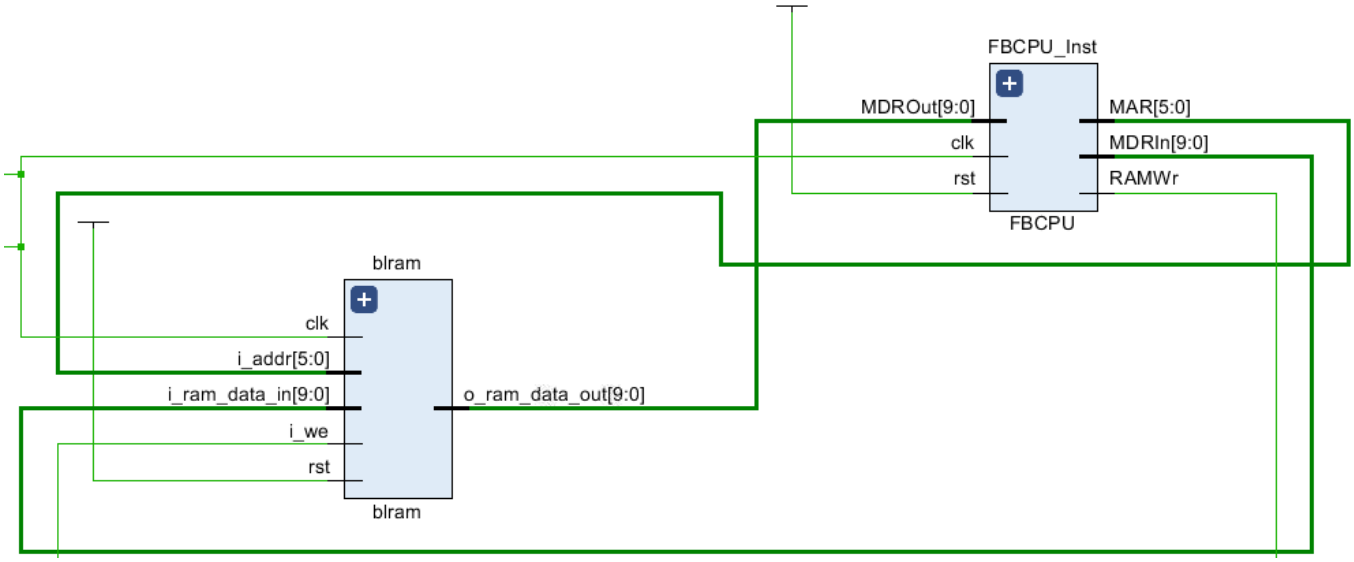
İşlemci tasarımı için Verilog dilinde geliştirilmiş başlangıç tasarımı verilmiştir.

Başlangıç tasarımı: http://www.levent.tc/files/courses/digital_design/project/fbcpu_baslangic.rar

Başlangıç tasarımı rar arşivinin içerisinde 6 adet dosya bulunmaktadır. Bunlar;

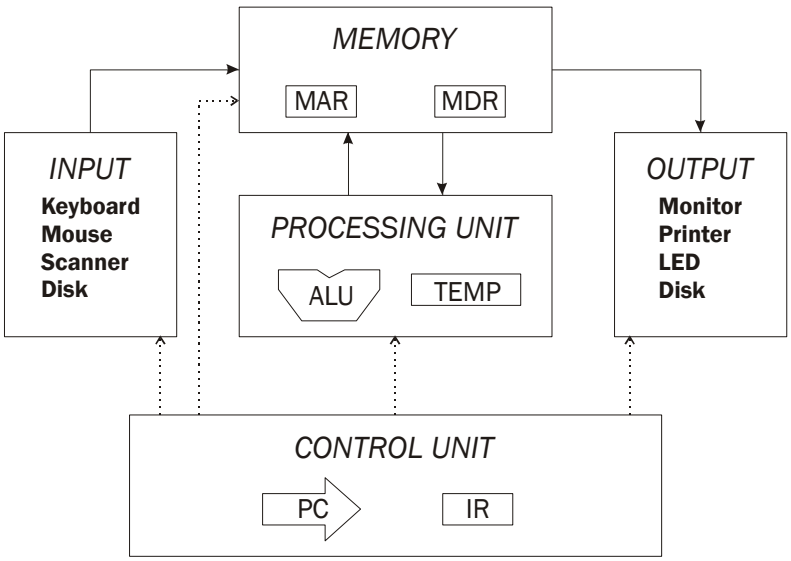
- fbcpu_core.v: İşlemcinin tasarımını barındırır.
- tb_fbcpu.v: İşlemciyi test edecek olan komutları besler ve sonucun doğru olup olmadığını kontrol eden testbench tasarımıdır.
- memory.v: Komutlar ve verilerin tutulduğu, Block RAM olarak tasarlanmış bellektir.
- testCase1.v: Örnek yazılım başlığında verilen 1. Yazılımı içerir.
- testCase2.v: Örnek yazılım başlığında verilen 2. Yazılımı içerir.
- testCase3.v: Örnek yazılım başlığında verilen 3. Yazılımı içerir.

tb_fbcpu.v dosyasında verilen testbench tasarımı, fbcpu_core.v, memory ve testCase1-2-3 dosyalarını alt modül olarak kullanmaktadır. Şekil 1'de testbenchte modüllerin birbirlerine olan bağlantıları verilmektedir.



Şekil 1. Testbench Bağlantılar

Tasarlanması istenen FBCPU RTL tasarımı, Von Neumann mimarisindedir. Şekil 2'de Von Neumann Mimarisi verilmektedir.



Şekil 2. Von Neumann Mimarisi

Temel olarak 4 elemanı vardır.

- Saklayıcılar (Şekil 2'de Processing Unit'in altındaki Temp değişkeni)
- Bellek (RAM)
- İşlem Ünitesi (ALU)
- Kontrol Ünitesi

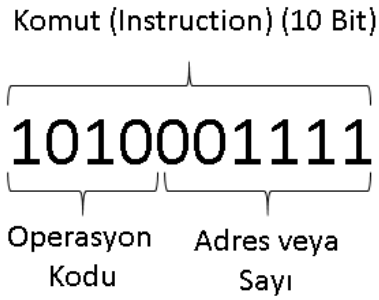
FB-CPU'nun desteklediği operasyonlar Tablo 1'de verilmektedir.

Tablo 1. FB-CPU ISA (Instruction Set Architecture)

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
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

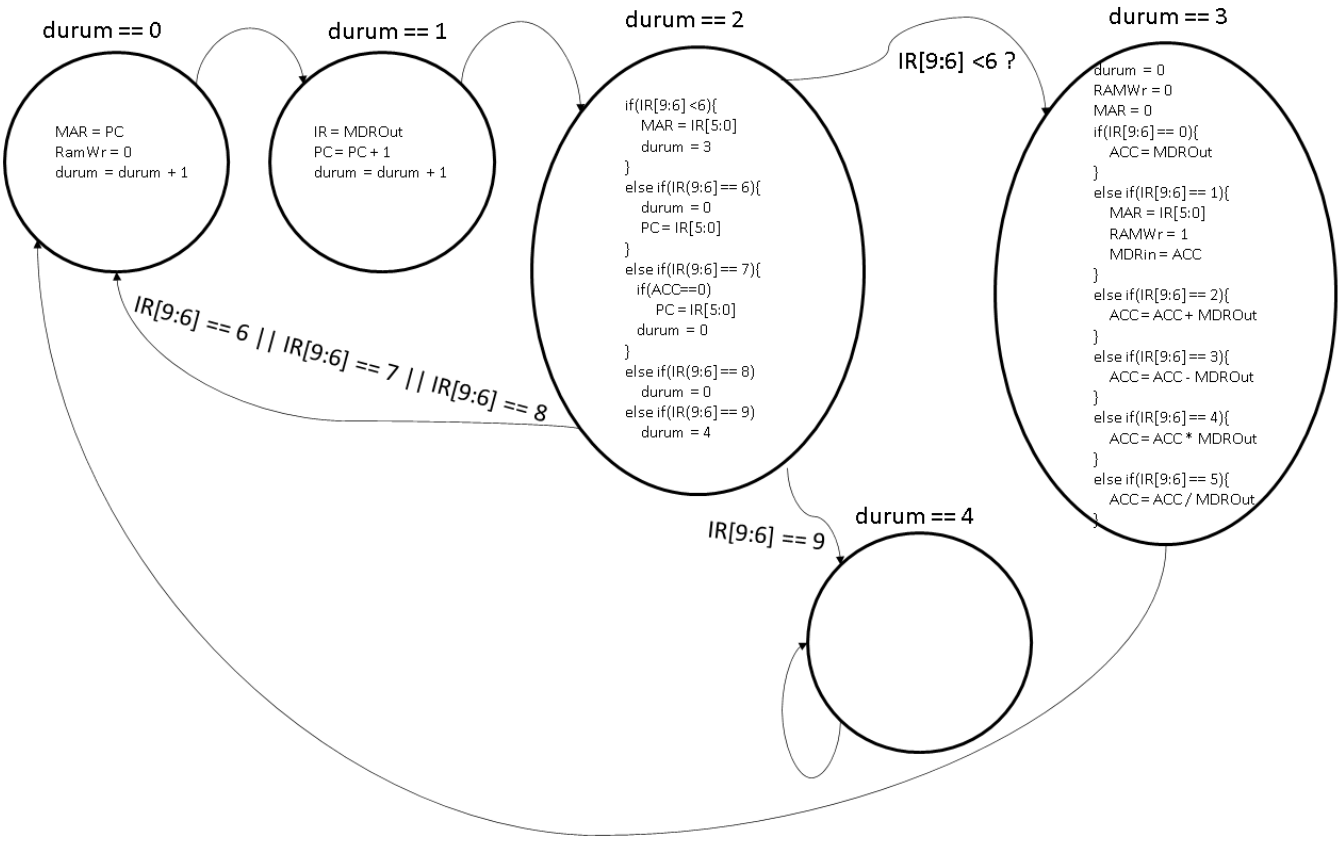
İşlemci 9 adet komutu desteklemektedir.

Şekil 3'te FB-CPU'nun 10 bitlik komutunun, operasyon ve adres için bitlerinin ayrılması gösterilmiştir.



Şekil 3. FB-CPU Örnek Komut Binary Gösterimi

FB-CPU'nun durum diyagramı olarak ifade edilmiş hali Şekil 4'te verilmektedir. İşlemcinin adım adım yapması gereken işler bir arada göstermektedir.



Şekil 5. FB-CPU Durum Makinası Gösterimi

5.1. Saklayıcılar

Başlangıç tasarımında FBCPU'nun gerek duyacağı tüm saklayıcılar tanımlanmıştır. Tasarımda 4 adet saklayıcı bulunmaktadır.

Bunlar;

- durum: Durum makinasında, hangi durumda olduğunu bilgisi tutulur.
- PC: RAM'deki hangi adresteki komutun çalıştığı bilgisi tutulur.
- IR: O anda çalışan komutun kendisi tutulur.
- ACC: Geçici saklama alanı.

Şekil 6'da fbcpu_core.v dosyasında tanımlanmış olan saklayıcılar gösterilmektedir.

```
always@(posedge clk) begin
    durum       <= #1 durumNext;
    PC          <= #1 PCNext;
    IR          <= #1 IRNext;
    ACC         <= #1 ACCNext;
end
```

Şekil 6. FBCPU Saklayıcıları

Yeni saklayıcı eklenmeyecektir. Tüm gerekli saklayıcılar tasarımda bulunmaktadır.

FB-CPU durum makinaları yöntemi ile gerçeklenecektir. Yani bu işlemci durum ismindeki saklayıcının değerine göre 2^3 = 8 farklı durumda çalışan bir tasarımı olacaktır (İşlemcinin desteklemesi istenen işlemlerin tamamı 8 farklı durumda yapılabilmektedir).

Diğer tüm saklayıcılar, durum saklayıcısının değişimine göre çalışacaktı. Yani durum'un değerine göre tüm saklayıcıların giriş sinyalleri değişmektedir. Diğer bir değiş ile, durum saklayıcısının değerine göre saklayıcıların üzerine başka başka sinyaller atanmakta, sistemin ilerlemesi durum sinyaline bağlıdır.

Tasarımda giriş çıkış portlarına bağlı olan bellek sinyalleri aşağıda verilmektedir.

- **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.

5.2. Bellek (RAM, Random Access Memory):

FB-CPU'nun komutları okuyup, hesaplanan değerleri geri yazacağı bir Block RAM mekanizması bulunmaktadır. Bu bellek fbcpu_core.v dosyasında bulunmamaktadır. Testkodunun instantiate ettiği bellek memory.v dosyasında bulunmaktadır. RAM'e bağlı 4 saklayıcı, clock ve reset sinyali bulunmaktadır. RAM'e bağlı saklayıcıların görevleri saklayıcılar bölümünde açıklanmıştır.

5.3. İşlem Ünitesi (ALU, Arithmetic Logic Unit): Aritmetik işlemlerin gerçekleştirildiği bölümdür. FB-CPU'da 3 adet aritmetik işlem vardır. Bunlar toplama, çıkartma ve çarpma, gelen operasyon koduna göre işlemleri gerçekleştirip ACC saklayıcısına yazmaktadır.

5.4. Kontrol Ünitesi: Saklayıcılar, Aritmetik İşlem Ünitesi ve RAM'e verilerin birbirleri arasında transferinden sorumludurlar. İşlemci içi veri akışını yönetir.

İşlemci tasarımı için verilen başlangıç verilog tasarımında, İşlem ve Kontrol Ünitesi eksiktir. Bu eksik yerler, durum makinasında 2 ve 3 durumların içerisindedir.

6. Eksik Ünitelerin Tasarımı

Verilen başlangıç tasarımında durum 0 ve durum 1 için tasarım verilmiştir.

LAB'da durum 2'nin tasarımı yapılacaktır.

İşlemcinin çalışır olması için durum 3'ün tamamlanması gerekmektedir.

7. Örnek Yazılım

FB-CPU 10 bitlik komutunda, ilk 4 biti [9:6] operasyon kodunu, son 6 biti ise [5:0] adresi temsil etmektedir. Tablo 1'de verilmiş olan komutlar ile uygulamalar geliştirilmelidir. İşlemcinin doğru çalışıp çalışmadığı aşağıdaki kod parçacıkları ile test edilebilir. Hangi yazılım ile test yapılmak isteniyorsa, testbench'in en üstünde yer alan

“TEST_CASE = 1” olan satırı, 1-2-3 sayısından birisi ile değiştirilebilir. Yani 2. yazılım ile test yapılmak isteniyorsa, “TEST_CASE = 2” olarak düzenlenmelidir. Testbench bu durumda, belleğe 2. test kodlarını yükleyecektir.

7.1. Test Yazılımı 1

FB-CPU için bellekte 50 ve 51 adresteki iki sayının toplamını 52 no’lu adrese kaydeden uygulamayı geliştiriniz.

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

7.2. Test Yazılımı 2

FB-CPU için bellekte 50 ve 51 adresteki iki sayının çarpımını 52 no’lu adrese kaydeden uygulamayı geliştiriniz.

0: 0000_110010 // LOD 50, (ACC = *50), Hex = 32
1: 0100_110011 // ADD 51, ACC = ACC * (*51), Hex = 133
2: 0001_110100 // STO 52, (*52) = ACC, Hex = 74
3: 1001_000000 // Halt, Hex = 240
50: 0000000101 // Hex = 5
51: 0000001010 // Hex = A

7.3. Test Yazılımı 3

FB-CPU için bellekte 50 ve 51 adresteki iki sayının çarpımını 52 no’lu adrese kaydeden uygulamayı geliştiriniz. Ancak çarpma operasyonunu kullanmayınız. Çarpma işlemi için 50’deki sayıyı 51’deki sayı defa toplayıp 52 no’lu adrese yazınız. Gerekli değişkenler için istediğiniz adresleri kullanabilirsiniz.

0: 0000_110011 // LOD 51, ACC = *51, Hex = 33
1: 0011_110001 // SUB 49, ACC = ACC - *49, Hex = F1
2: 0111_001010 // JMZ 10, döngü bittiyse, döngüden çıkartacaktır (ACC-49 == 0), 10. Satır, Hex = 1CA
3: 0000_110000 // LOD 48, temp değerini yükle, başlangıçta 0, Hex = 30
4: 0010_110010 // ADD 50, ikinci sayıyı ACC’nin üstüne ekle, Hex = B2
5: 0001_110000 // STO 48, ACC’nin değerini temp’e ata, Hex = 70
6: 0000_110001 // LOD 49, ACC = i, Hex = 31
7: 0010_101110 // ADD 46, ACC = i + 1, Hex = AE
8: 0001_110001 // STO 49, i = i + 1, Hex = 71
9: 0110_000000 // JMP 0, döngünün başına dön 0. satır, Hex = 180
10: 0000_110000 // LOD 48, ACC = temp, Hex = 30
11: 0001_110100 // STO 52, *52 = ACC, Hex = 74
10: 1001_000000// HLT, bitirme, Hex = 240

46: 1 // 1 sayısı
48: 0 // Hex = 0, temp

49: 0 // Hex = 0, i index'i için
50: 0000000101 // Hex = 5
51: 0000001010 // Hex = A

8. Notlandırma ve Proje Teslimi:

Bu başlık FB-CPU'nun proje teslimi ve notlandırılması hakkında bilgiler içermektedir.

8.1. Notlandırma:

Projenin **iki** ana değerlendirme kriteri vardır. Her iki kriter 50 şer puandır.

İlk kriter FB-CPU'nun komutlarının (instructions) **doğru çalıştırılmasıdır**. FB-CPU'nun 9 komutu (Instruction)'u vardır. Her bir komut'un doğru sonuç üretip üretememesine göre değerlendirme yapılacaktır.

İkinci kriter ise **Proje Teslim Dokümanı ve Sunumdur**.

- **Proje Teslim Dokümanı:**

Öğrenciler, proje raporlarını verilen "Proje Teslim Dokümanı" 'nın içerisini doldurarak yapacaklardır.

Proje Teslim Dokümanı: http://www.levent.tc/files/courses/digital_design/project/BLM201_proje_teslim_dokumani.docx

Proje teslim dokümanında, sarı işaretlenmiş olan yerleri silerek, ilgili içerikleri yazınız. Proje teslim dokümanı en az 3, en fazla 5 sayfa olmalıdır.

- **Proje Sunumu:**

Powerpoint üzerinde ortalama 5 dakika (4-6 dakika arası) sürecek bir sunum hazırlayarak kayıt etmelidirler. Kayıt işlemi, cep telefonu veya bilgisayar ekran kayıt yazılımları (Screen-Recorder, Bandicam vb…) ile yapılabilir.

Sunum, ekip üyeleri içinden biri tarafından, projenin nasıl yapıldığı, işlemcinin nasıl çalıştığı vb.. konularının powerpoint slaytları üzerinden anlatılırken kaydedilmesi ile olmalıdır. Sunum video'sunda powerpoint slaytları okunabilir ve konuşmacının sesinin anlaşılır olması gerekmektedir. Powerpoint slayt görünüm tasarımı istenildiği gibi yapılabilir.

Proje ekibinin tamamı, notlarını bu değerlendirmeye göre alırlar.

8.2. Teslim:

Projenin teslimi için aşağıdaki adımların gerçekleştirilmesi gerekmektedir. İstenen dosyaları sadece proje ekip sorumlusunun getirmesi, Teams ve Github (Çok yaygın bir açık kaynak kod paylaşım platformudur)'a yüklemelidir.

Proje ekip sorumlusunun Teams açılmış olan "Proje Teslim" sayfasına aşağıdaki dosyaların yüklenmesi gerekmektedir.

- Verilog RTL Tasarım (Tamamlanmış fbcpu_core.v dosyası)
- Hazırlanan powerpoint sunum dosyası (.ppt uzantılı dosya)
- Proje Teslim Dokümanı (Word formatında yüklenmelidir)
    - Dokümanın alt başlıkları doldurulmalıdır
    - Kaydedilen powerpoint sunum video'su youtube'a yüklenip, adresi, dokümanın sonuçlar bölümündeki açılmış yere link'i yazılmalıdır (Video'nun herkes'e görünür olmamasını istiyorsanız, youtube'a yükledikten sonra liste dışı seçeneğini seçerek, sadece link'e sahip olan kişilerin görmesini sağlayabilirsiniz).
    - Teams'e yüklenen tüm dosyalar (Tamamlanmış fbcpu_core.v dosyası, ppt uzantılı sunum dosyası ve Proje Teslim Dokümanını (PDF formatında)), github.com sitesine üye olup, yüklenip, dokümanın sonuçlar bölümündeki yere link'i yazılmalıdır.