	TOBB Ekonomi ve Teknoloji Üniversitesi **Bilgisayar Mühendisliği Bölümü** **Elektrik ve Elektronik Mühendisliği Bölümü**	11 Aralık 2024 BİL 265/264/264L – Mantıksal Devre Tasarımı ve Laboratuvarı 2024 – 2025 Öğretim Yılı Güz Dönemi Final Sınavı

AÇIKLAMALAR:

1. Sınavı çözmeye başlamadan önce tüm açıklamaları ve soruları okuyun. Sınavda toplam 6 sayfa (3 adet A4 kağıdı), 2 soru var ve soruların toplam değeri **100** puandır. Bütün soruların değeri köşeli ayraç ile belirtilmiştir. Sınav süresi 150 dakikadır.
2. Sınav sırasında soru kabul edilmeyecektir.
3. Sınav esnasında internet ve tarayıcı kullanımı yasaktır. Bilgisayarda Xilinx Vivado programı dışında hiçbir program **KESİNLİKLE** açık olamaz.
4. İnternete bağlı olduğu veya herhangi bir tarayıcısı açık olduğu görülen kişilerin sınavları geçersiz sayılacak ve kopya olarak değerlendirilip gerekli işlemler yapılacaktır.
5. Sınav boyunca **sadece** bir yüzü dolu A4 sayfadan faydalanabilirsiniz. Bunun dışında her türlü araç/gereç ve kaynak kullanımı yasaktır.(hesap makinesi, akıllı saat, telefon, pdf dosyaları vb.)
6. Sonucu yanlış olan yanıtlar puan alamayabilir. Gidiş yolunun ayrıntılı gösterilmesi sorudan puan alınması için gereklidir ancak yeterli değildir. Açıklamasız kod yazmamaya özen göstermeniz alacağınız puanı artıracaktır.
7. <Projenizin bulunduğu dizin>\<Proje ismi>\<Proje ismi>.srcs\sources_1\new → dizininde yazdığınız ".v" uzantılı dosyaları bulabilirsiniz. Simülasyon dosyalarını ise aynı uzantıda .srcs'den sonra \sim_1 klasöründe bulabilirsiniz.
8. **9. talimata uyulmaması ve dosya isimlerinin yanlış yazılması durumlarında toplam puanınız üzerinden 20 puan kırılacaktır.**
9. **Dosya gönderimi için sorularda belirtilen ".v" uzantılı dosyalarınızı "isim_soyisim_numara_final" isimli bir klasöre attıktan sonra klasörü sıkıştırınız ve sınav sırasında gözetmenin getireceği USB veya Uzak'a yüklemeye hazır olacak şekilde bekleyiniz.**

   Sınavda Göndermeniz Gereken .v dosyaları: (Gönderim yapmak istemediğiniz soruları eklemek zorunda değilsiniz.)

   - catch.v
   - decoder.v
   - verici.v
   - verici_decoder.v

1. [30 Puan] Catch (Verilog Davranışsal Modelleme)

Oluşturacağınız modüle "catch"(oluşacak dosya "catch.v") ismini verin.
İstenen modülü Verilog dilinde **davranışsal modelleme** ile gerçekleştirin.

Devrenin girişleri:

clk: (1 bit) saat girişi
rst: (1 bit) reset girişi (sinyal 1 olunca saat ile senkron şekilde resetler)
yon: (4 bit) Oyuncuların hareketlerini gösterir.yon[3:2] İlk oyuncunun hareketini, yon[2:1] ikinci oyuncunun hareketini temsil eder. 0: yukarı, 1:aşağı, 2:sol, 3:sağ

Devrenin çıkışları:

durum: (1 bit): 1.oyuncunun 2.oyuncuyu yakaladığını gösteren çıkış. 1.oyuncu ile 2.oyuncu aynı karedeler ise bu çıkış bir sonraki çevrimde 1 olmalı. Değiller ise 0 olmalı.
yakalama_sayisi (4 bit): Toplam yakalama sayısını gösteren çıkıştır.

Tasarlayacağınız modül 2.oyunculu "catch" oyununu gerçekleştirmelidir. Oyunda 4x4'lük bir harita vardır. İlk oyuncu sol üst(3,3), ikinci oyuncu ise sağ alt(0,0) köşeden başlamaktadır. Her saat darbesinde her iki oyuncu da birer kare(yukarı,aşağı,sol,sağ) gidebilmektedir. Harita dışına çıkmak mümkün değildir. Eğer bir oyuncu haritanın sınırındaysa ve harita dışına çıkmasını gerektirecek bir hareket konumu gelmekte ise, o hareket gerçekleştirilememektedir. 1. oyuncu 2.oyuncuyu yakaladığı zaman bir sonraki çevrimde durum çıkışı 1, diğer durumlarda ise 0 olmalıdır. Yakalama sonrası oyuncular tekrar hareket ettiği için, durum çıkışı her saat darbesinde tekrar hesaplanmalıdır. 1.oyuncu 2.oyuncuyu 15 kez yakaladıktan sonra, oyun en baştan tekrar başlamalıdır(oyuncuların ilk konumları en baştaki gibi olmalıdır).

	3	2	1	0
3	1			
2				
1				
0				2

Örnek: Yandaki durumda, 2 numaralı oyuncu sadece sola veya yukarı, 1 nolu oyuncu ise sadece sağa veya aşağı gidebilmektedir. Aksi bir yönde hareket komutu gelirse, bu hareket gerçekleştirilmekte ve oyuncu aynı karede kalmaktadır. Bu durumda yon = 4'b1100 gelirse, ilk oyuncunun yönü 3 yani sağ, ikinci oyuncunun yönü ise 0 yani yukarı olacaktır. Bir sonraki çevrimde 1. oyuncu (3,2), 2. oyuncu ise (1,0) karesinde olacaktır

2. [70 puan] Boru Hattı (Verilog Davranışsal Modelleme)

Not: Parametrik yazamayanlar N=30 için yazabilir. Parametrik yazmayanlardan puan kırılacaktır !!

A) [30 puan] Oluşturacağınız modüle "decoder"(oluşacak dosya "decoder.v") ismini verin. İstenen modülü Verilog dilinde **davranışsal modelleme** ile gerçekleştirin.

Devrenin parametre girişi:

N: Gelen şifrelenmiş ve çözülmüş olarak çıkış olarak verilecek verinin bit uzunluğunu belirleyen parametre. (varsayılan değer: 30) ($3 \leq N \leq 60$) ($N \% 3 = 0$)

Devrenin girişleri:

clk: (1 bit) saat girişi

rst: (1 bit) reset girişi (sinyal 1 olunca saat ile senkron şekilde resetler)

basla: (1 bit) verinin gelmeye başladığı ilk çevrim 1 olan giriş.

mod: (1 bit) gelen verinin (mod = 1) çok çevrimde seri olarak veya (mod = 0) tek çevrimde paralel olarak geleceğini belirten giriş.

gelen_veri: (N bit) Dinamik sezar şifreleme ile şifrelenmiş verinin seri veya paralel olarak alındığı veri girişi.

Devrenin çıkışları:

cikan_veri: (N bit) Şifreli gelen N bitlik verinin çözülmüş halini dışarıya veren veri çıkışı.

bitti: (1 bit) cikan_veri'nin hazır olduğu çevrim 1 olan sinyal çıkışı.

İsterler şu şekildedir:

- ***rst*** sinyali geldiğinde, saat ile senkron şekilde, bütün devrenin durumunu sıfırlamanız gerekmektedir.
- ***basla*** sinyalinin 1 olduğu çevrim veri gelmeye başlar.
- decoder modülünün hangi modda veriyi alacağı **basla** sinyalinin 1 olduğu çevrimdeki **mod** girişindeki değere göre belirlenir.
- **N** bitlik şifrelenmiş veri 3'er bitlik karakterlerden oluşmak üzere **N/3** adet karakterden oluşmaktadır.
- **basla** sinyalinin 1 olduğu çevrim **mod** girişi 0 ise aynı çevrimde **gelen_veri** girişinden **N** bit uzunluğundaki şifrelenmiş veri tek çevrimde alınır. Bir sonraki çevrim çözmeye başlanır.
- **basla** sinyalinin 1 olduğu çevrim **mod** girişi 1 ise şifrelenmiş **N** bit uzunluğundaki veri en anlamlı karakterden başlayarak **gelen_veri** girişinin **en anlamsız** 3 bitinden **N/3** çevrim boyunca karakter karakter alınır (gelen_veri[2:0] bitleri üzerinden her çevrim karakter okunur). Son karakterin alındığı çevrimden sonraki çevrim çözmeye başlanır.

- Dinamik Sezar şifre ile şifrelenmiş **N** bitlik veri alındıktan sonra her çevrim bir adet 3 bitlik karakter çözülmek üzere **N/3** çevrim boyunca gelen şifrelenmiş veri çözülür.
- Son karakterin çözüldüğü çevrim **bitti** sinyali 1 yapılarak **N** bitlik **cikan_veri** çıkışından çözülmüş veri dışarıya verilir ve bir sonraki çevrim modül yeni veri alabilir duruma döner.
- Sezar şifrelemede alfabe üzerinde sağa doğru **K** sabit sayısına göre yapılan dairesel karakter kaydırma sayesinde şifreleme yapılır.
- Dinamik sezar şifrelemede ise **K** değeri ilk şifrelenecek karakter için 1 olarak başlar ve sonraki şifrelenecek karakterler için **K** değeri **bir önceki karakterin şifrelenmiş halinin** bit karşılığındaki büyüklüğüne göre belirlenir. Şifreleme en anlamlı karakterden başlar.
- Bu sorudaki kullanılacak alfabe her biri 3 bitle ifade edilen 8 adet karakterden oluşmaktadır. Alfabenin tanımı, örnek şifreleme ve örnek bir modül giriş, çıkışları aşağıdaki gibidir.

Alfabe:

Harf:	Ö	Ü	T	E	K	N	R	G
Bit:	3`b000	3`b001	3`b010	3`b011	3`b100	3`b101	3`b110	3`b111

Örnek: Veri: "TÜRK" = 12`b010001110100 Şifrelenmiş Veri: "EKTR" = 12`b011100010110

N=12	1. Harf	2. Harf	3. Harf	4. Harf
Harf:	T	Ü	R	K
Bit:	3`b010	3`b001	3`b110	3`b100
K değeri:	1	3	4	2
Şifrelenmiş Harf:	E	K	T	R
Şifrelenmiş Bit:	3`b011	3`b100	3`b010	3`b110
Yeni K değeri:	3	4	2	6

Örnek: "x" önemsiz anlamına gelmektedir.

N = 6	1.çevrim	2.çevrim	3.çevrim	4.çevrim	5.çevrim	6.çevrim
rst:	0	0	0	0	0	0
basla:	0	1	0	0	0	0
mod:	x	0	x	x	x	x
gelen_veri:	6`bxxxxxx	6`b011100	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
cikan_veri:	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`b010001	6`bxxxxxx	6`bxxxxxx
bitti:	0	0	0	1	0	0
Açıklama	Boşta	Veri alma	Çözme	Çözme + Sonuç	Boşta	Boşta

Örnek: "x" önemsiz anlamına gelmektedir.

N = 6	1.çevrim	2.çevrim	3.çevrim	4.çevrim	5.çevrim	6.çevrim
rst:	0	0	0	0	0	0
basla:	0	1	0	0	0	0
mod:	x	1	x	x	x	x
gelen_veri:	6`bxxxxxx	6`bxxx011	6`bxxx100	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
cikan_veri:	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`b010001	6`bxxxxxx
bitti:	0	0	0	0	1	0
Açıklama	Boşta	Veri alma	Veri alma	Çözme	Çözme + Sonuç	Boşta

B) [15 puan] Oluşturacağınız modüle "verici"(oluşacak dosya "verici.v") ismini verin. İstenen modülü Verilog dilinde **davranışsal modelleme** ile gerçekleştirin.

Devrenin parametre girişi:

N: Gelen verinin bit uzunluğunu belirleyen parametre. (varsayılan değer: 30)
(3 ≤ N≤ 60) (N % 3 = 0)

Devrenin girişleri:

clk: (1 bit) saat girişi.
rst: (1 bit) reset girişi (sinyal 1 olunca saat ile senkron şekilde resetler).
basla: (1 bit) verinin geldiğini belirten sinyal girişi.
mod: (1 bit) alınan verinin (mod = 1) çok çevrimde seri olarak veya (mod = 0) tek çevrimde paralel olarak çıkışa verileceğini belirten giriş.
gelen_veri: (N bit) veri girişi.

Devrenin çıkışları:

cikan_veri: (N bit) Gelen N bitlik veriyi seri olarak veya paralel olarak dışarıya veren veri çıkışı.
bitti: (1 bit) Verinin dışarayı verilmesinin bittiği çevrim 1 olan sinyal çıkışı.
İsterler şu şekildedir:

- ***rst*** sinyali geldiğinde, saat ile senkron şekilde, bütün devrenin durumunu sıfırlamanız gerekmektedir.
- ***basla*** sinyalinin 1 olduğu çevrim N bitlik veri gelir.
- verici modülünün hangi modda veriyi dışarıya vereceğini **basla** sinyalinin 1 olduğu çevrimdeki **mod** girişindeki değere göre belirlenir.
- **N** bitlik gelen veri 3'er bitlik karakterlerden oluşmak üzere **N/3** adet karakterden oluşmaktadır.
- **basla** sinyalinin 1 olduğu çevrim **mod** girişi 0 ise aynı çevrimde **gelen_veri** girişinden alınan **N** bit uzunluğundaki veri bir sonraki çevrim dışarıya cikan_veri çıkışından N bit olarak verilir ve **bitti** sinyali 1 yapılır.
- **basla** sinyalinin 1 olduğu çevrim **mod** girişi 1 ise aynı çevrimde **gelen_veri** girişinden alınan **N** bit uzunluğundaki veri bir sonraki çevrim başlamak üzere en anlamlı karakterden başlayarak her çevrim **cikan_veri** çıkışının **en anlamsız 3 bitinden karakter karakter** dışarıya verilir. Son karakterin dışarıya verildiği çevrim **bitti** sinyali 1 yapılır. (cikan_veri[2:0] bitleri üzerinden her çevrim karakter dışarıya verilir)
- Bitti sinyalinin 1 olduğu çevrimden sonraki çevrim modül tekrardan veri alabilir durumuna geri döner.
- Aşağıda iki mod için örnek girdi çıktılar verilmiştir.

Örnek: "x" önemsiz anlamına gelmektedir.

N = 6	1.çevrim	2.çevrim	3.çevrim	4.çevrim	5.çevrim	6.çevrim
rst:	0	0	0	0	0	0
basla:	0	1	0	0	0	0
mod:	x	0	x	x	x	x
gelen_veri:	6`bxxxxxx	6`b111011	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
cikan_veri:	6`bxxxxxx	6`bxxxxxx	6`b111011	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
bitti:	0	0	1	0	0	0
Açıklama	Boşta	Veri alma	Veri verme	Boşta	Boşta	Boşta

Örnek: "x" önemsiz anlamına gelmektedir.

N = 6	1.çevrim	2.çevrim	3.çevrim	4.çevrim	5.çevrim	6.çevrim
rst:	0	0	0	0	0	0
basla:	0	1	0	0	0	0
mod:	x	1	x	x	x	x
gelen_veri:	6`bxxxxxx	6`b111011	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
cikan_veri:	6`bxxxxxx	6`bxxxxxx	6`bxxx111	6`bxxx011	6`bxxxxxx	6`bxxxxxx
bitti:	0	0	0	1	0	0
Açıklama	Boşta	Veri alma	Veri verme	Veri verme	Boşta	Boşta

C) [25 puan] Oluşturacağınız modüle "verici_decoder"(oluşacak dosya "verici_decoder.v") ismini verin. İstenen modülü Verilog dilinde **davranışsal modelleme** ile gerçekleştirin.

Devrenin parametre girişi:

N: Gelen verinin bit uzunluğunu belirleyen parametre. (varsayılan değer: 30)
$(3 \leq N \leq 60)\ (N \% 3 = 0)$

Devrenin girişleri:

clk: (1 bit) saat girişi.
rst: (1 bit) reset girişi (sinyal 1 olunca saat ile senkron şekilde resetler).
basla: (1 bit) verinin geldiğini belirten sinyal girişi.
mod1: (1 bit) gelen verinin (mod1 = 1) çok çevrimde seri olarak veya (mod1 = 0) tek çevrimde paralel olarak geleceğini belirten giriş.
mod2: (1 bit) gelen verinin (mod2 = 1) çok çevrimde seri olarak veya (mod2 = 0) tek çevrimde paralel olarak çıkışa verileceğini belirten giriş.
gelen_veri: (N bit) dinamik sezar şifreleme ile şifrelenmiş verinin seri veya paralel olarak alındığı veri girişi.

Devrenin çıkışları:

cikan_veri: (N bit) çözülmüş N bitlik veriyi seri olarak veya paralel olarak dışarıya veren veri çıkışı.
bitti: (1 bit) çözülmüş verinin dışarayı verilmesinin bittiği çevrim 1 olan sinyal çıkışı.
İsterler şu şekildedir:

- ***rst*** sinyali geldiğinde, saat ile senkron şekilde, bütün devrenin ve alt modüllerin durumunu sıfırlamanız gerekmektedir.
- ***basla*** sinyalinin 1 olduğu çevrim veri gelmeye başlar.
- mod1 ve mod2 basla sinyalinin 1 olduğu çevrim belirlenir.
- Gelen verinin mod1'e göre giriş olarak alınma şekli ve çözülme şekli A şıkkındaki decoder modülünde olduğu gibidir.
- Çıkan verinin mod2'ye göre çıkış olarak verilme şekli B şıkkında verici modülünde olduğu gibidir.
- A ve B şıklarında yazdığınız **verici** ve **decoder** modüllerini kullanarak veriyi 2 farklı modda alarak dinamik sezar şifrelemeye göre çözen ve çözülen veriyi 2 farklı modda dışarıya verebilen bir **boru hattı** oluşturarak **verici_decoder** modülünü tasarlamanız istenmektedir.
- Boru hattındaki **decoder'ın** işlemini bitirdiği çevrimden sonraki çevrim **verici_decoder** modülü yeni giriş alabilmelidir. Yeni bir işlem almak için verici modülünün işlemini bitirmesi beklen**meme**lidir.
- **decoder'ın** işlemini bitirdiği çevrimin sonraki çevreminde **verici_decoder** modülü belirlenen mod2'ye göre veriyi dışarıya vermeye başlamalıdır.
- Aşağıdaki verilen örnekteki gibi hangi çevrimde hangi işlemlerin yapıldığına dikkat edilmelidir.

Örnek: "x" önemsiz anlamına gelmektedir.

N=6	1.çevrim	2.çevrim	3.çevrim	4.çevrim	5.çevrim	6.çevrim	7.çevrim	8.çevrim
rst:	0	0	0	0	0	0	0	0
basla:	0	1	0	0	1	0	0	0
mod1:	x	0	x	x	0	x	x	x
mod2:	x	1	x	x	0	x	x	x
gelen_veri:	6`bxxxxxx	6`b011100	6`bxxxxxx	6`bxxxxxx	6`b011100	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx
cikan_veri:	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxxxxx	6`bxxx010	6`bxxx001	6`bxxxxxx	6`b010001
bitti:	0	0	0	0	0	1	0	1