2021-2022 Eğitim-Öğretim Yılı Güz Yarıyılı/*2021-2022 Academic Year Fall Semester*
Mühendislik ve Mimarlık Fakültesi/*Faculty of Engineering and Architecture*
Bilgisayar Mühendisliği Bölümü/*Computer Engineering Department*
COMP343 SOC Tasarım Yarıyıl İçi Sınavı/*COMP343 SOC Design Midterm Exam*

Öğrenci Adı Soyadı / *Student's Name Surname*:	**Sınav Notu - Paraf /** Exam Score - Initials
Öğrenci Numarası / *Student's Number:*	
Öğrenci İmzası / *Student's Signature:*	

Sınav Tarihi / *Exam Date:*				**Sınav Süresi** / *Exam Duration:*						
Soru Numarası / *Numbers of the Questions*	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Alınan Not / *Scored Points*										
Sınav Kuralları / *Exam Rules:*										

**Her sorunun puan değeri rakamsal olarak yanına belirtilmelidir. / The point for each question must be stated next to the question.*

SORULAR/*QUESTIONS*

1. **(25 Puan)** Aşağıda verilen Verilog kod parçacığı sentezlenince oluşacak donanım şematiğini çiziniz.

```
module prob (A, B, C, E);
        input A, B, E;
        output reg C;

        always @(*)
                if (A)
                        C = B ^ E;
                else
                        C = A;

        end
endmodule
```

Dersin Sorumlusu / Lecturer
Başarılar / Good Luck

2. **(15 Puan)** Aşağıda ZYNQ mimarisi ile ilgili verilen soruları yanıtlayınız.

   a) MIO nedir? Ne amaçla kullanılır?
   b) EMIO nedir? Ne amaçla kullanılır?
   c) ZYNQ mimarisinde bulunan GP, HP ve ACP portları nedir? Aralarındaki fark nedir?

3. **(15 Puan)** Aşağıda AXI veriyolları ile ilgili verilen soruları yanıtlayınız.

   a) AXI Lite ve AXI MM veriyolu üzerinde bulunan 5 kanalın isimleri ve görevleri nelerdir?
   b) AXI arayüzlerindeki ready sinyali ne işe yaramaktadır?
   c) AXI arayüzlerindeki valid sinyali ne işe yaramaktadır?

Dersin Sorumlusu / Lecturer
Başarılar / Good Luck

4. **(45 Puan)** Aşağıda Xilinx'in PS Uart Polled Example kod parçacığı verilmiştir. Bu kod parçacığında 32 adet veri UART hattına iletilip, loopback yapılıp geriye aynı değerlerin alındığı kontrol edilmektedir. Bu kod parçacığını temel alarak, UART hattından 3 adet sayı yakalıp, ilk sayı operasyonun ne olduğunu, ardışık gelen iki sayı ise işleme sokulacak olan sayılar olacak bir uygulama geliştiriniz. Uygulama ilk gelen sayı 0 ise toplama, 1 ise çıkartma, 2 ise çarpma yapıp sonucunu tekrar UART hattından geri döndürmelidir. Aşağıdaki kod'a eklemeler ve silmeler yapılarak, verilen satır numaralarını kullanarak nereye ekleme ve silme yapıldı ise tüm kod tekrar yazılmadan, yeni uygulama açıklanabilir.

```
1.  #include "xparameters.h"
2.  #include "xuartps.h"
3.  #include "xil_printf.h"
4.  #define UART_DEVICE_ID
    XPAR_XUARTPS_0_DEVICE_ID
5.  #define TEST BUFFER SIZE 32

6.  int UartPsPolledExample(u16
    DeviceId);

7.  XUartPs Uart_PS;        /* Instance
    of the UART Device */

8.  static u8
    SendBuffer[TEST_BUFFER_SIZE]; /*
    Buffer for Transmitting Data */
9.  static u8
    RecvBuffer[TEST_BUFFER_SIZE]; /*
    Buffer for Receiving Data */

10. #ifndef TESTAPP_GEN
11. int main(void)
12. {
13. int Status;

14. Status =
    UartPsPolledExample(UART_DEVICE_ID)
    ;
15. if (Status != XST_SUCCESS) {
16. xil_printf("UART Polled Mode
    Example Test Failed\r\n");
17. return XST_FAILURE;
18. }

19. xil_printf("Successfully ran UART
    Polled Mode Example Test\r\n");
20. return XST_SUCCESS;
21. }
22. #endif

23. int UartPsPolledExample(u16
    DeviceId)
24. {
25. int Status;
26. XUartPs_Config *Config;
27. unsigned int SentCount;
28. unsigned int ReceivedCount;
29. u16 Index;
30. u32 LoopCount = 0;

31. Config =
    XUartPs_LookupConfig(DeviceId);
32. if (NULL == Config) {
33. return XST_FAILURE;
34. }

35. Status =
    XUartPs_CfgInitialize(&Uart_PS,
    Config, Config->BaseAddress);
36. if (Status != XST_SUCCESS) {
37. return XST_FAILURE;
38. }

39. Status =
    XUartPs_SelfTest(&Uart_PS);
40. if (Status != XST_SUCCESS) {
41. return XST_FAILURE;
42. }

43. XUartPs_SetOperMode(&Uart_PS,
    XUARTPS_OPER_MODE_LOCAL_LOOP);

44. for (Index = 0; Index <
    TEST_BUFFER_SIZE; Index++) {
45. SendBuffer[Index] = '0' + Index;
46. RecvBuffer[Index] = 0;
47. }

48. SentCount = XUartPs_Send(&Uart_PS,
    SendBuffer, TEST_BUFFER_SIZE);
49. if (SentCount != TEST_BUFFER_SIZE)
    {
50. return XST_FAILURE;
51. }

52. while (XUartPs_IsSending(&Uart_PS))
    {
53. LoopCount++;
54. }

55. ReceivedCount = 0;
56. while (ReceivedCount <
    TEST_BUFFER_SIZE) {
57. ReceivedCount +=
58. XUartPs_Recv(&Uart_PS,
    &RecvBuffer[ReceivedCount],(TEST_BU
    FFER_SIZE - ReceivedCount));
59. }

60. for (Index = 0; Index <
    TEST_BUFFER_SIZE; Index++) {
61. if (SendBuffer[Index] !=
    RecvBuffer[Index]) {
62. return XST_FAILURE;
63. }
64. }

65. XUartPs_SetOperMode(&Uart_PS,
    XUARTPS_OPER_MODE_NORMAL);

66. return XST_SUCCESS;
67. }
```

Dersin Sorumlusu / Lecturer

Başarılar / Good Luck