

# CORDIC Tabanlı HW/SW CoDesign

İrem Kalkanlı, Özlem Çalı, Deniz Uzun, Ceyda Uymaz, İrem Bozkurt

Fenerbahçe Üniversitesi  
Bilgisayar Mühendisliği  
İstanbul, Türkiye

e-mail: { irem.kalkanli , ozlem.cali, deniz.uzun, ceyda.uymaz,irem.bozkurt }@stu.fbu.edu.tr,

**Özetçe**— Donanım hızlandırıcı olarak Xilinx'in CORDIC (COordinate Rotation DIgital Computer) IP'si kullanıldığı ve işlemcinin hesap yükünü donanım hızlandırıcıya aktardığı bir proje yapılacaktır.

**Anahtar Kelimeler** — FPGA, CPU

**Abstract**— Created a project where processor transfer the computational load to the designed hardware accelerator using Xilinx's CORDIC (COordinate Rotation DIgital Computer) IP.

**Keywords** — FPGA, CPU.

## I. Giriş

Donanım hızlandırıcı olarak Xilinx'in CORDIC (COordinate Rotation DIgital Computer) IP'si kullanıldığı ve işlemcinin hesap yükünü donanım hızlandırıcıya aktardığı bir proje yapılacaktır.

## II. SİSTEM MİMARİSİ

Proje kapsamında 1 araç kullanılacaktır.

1)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. Vivado Tasarım Aracı, Xilinx'in 7 ve daha yeni jenerasyon FPGA'leri için kullanılabilen bir geliştirme ortamıdır. Bu ortam Xilinx'in sunduğu çeşitli geliştirme ve doğrulama araçlarını barındırır.

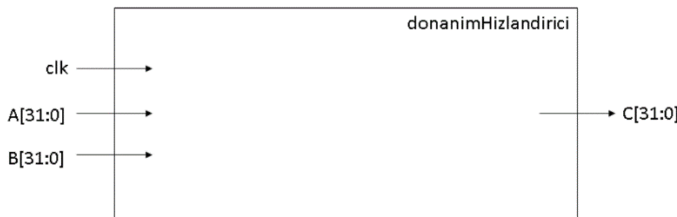
Vivado:

- Verilog
- System Verilog
- VHDL Dillerini desteklemektedir. Projede Verilog dili ile tasarımlar yapılacaktır.

## III. KULLANILAN YAZILIM

ZYNQ mimarisine sahip olan PYNQ geliştirme kartı üzerinde proje geliştirilecektir. ZYNQ'in PS (Processor) bölümü, tasarlanacak özel bir modüle verileri besleyip, sonucunu alacak şekilde tasarladık. Modülüzümün giriş ve çıkışları aşağıda gösterilmiştir.

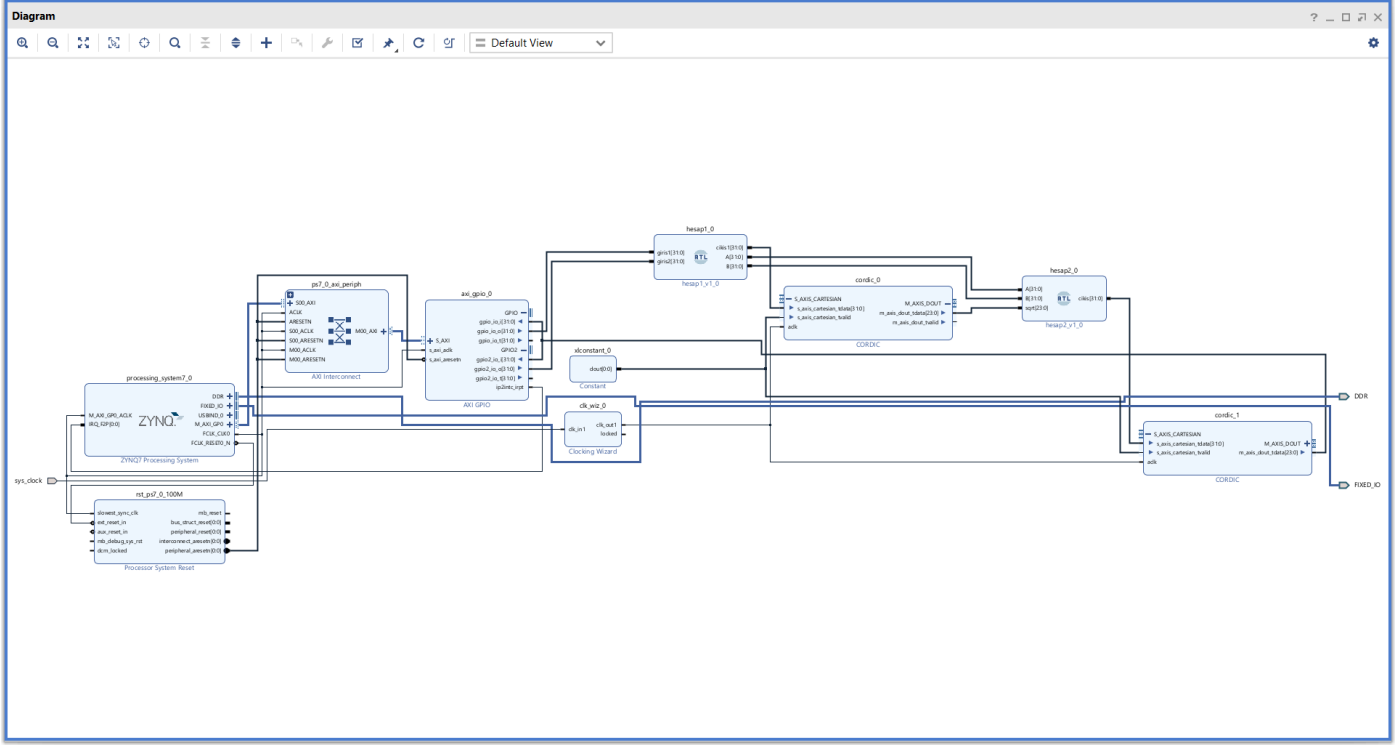
- clk, referans clock sinyali
- A[31:0] ve B[31:0] giriş sinyalleri
- C[31:0] çıkış sinyalleri



Bu modülümüzün içerisinde aşağıda verilen aritmetik işlemi yapan donanım bulunmaktadır.

$$C = \sqrt{\sqrt{A^2 + B^2} + A*B}$$

Karekök alma işlemi için CORDIC IP'sini kullandık. CORDIC IP'sinde karekök alma fonksiyonu için unsigned integer'ı seçtik. AXI GPIO IP'si ile tasarlanan modülün giriş ve çıkışlarına bağladık. PS tarafında A ve B sayıları örnek olarak 10 ve 20 olarak ayarlayıp , sonuca doğru bir şekilde ulaştık.



Öncelikle Zynq mimarisi eklenir. Sonra axi interconnect ve processor system reset de eklenir. Daha sonra AXI GPIO eklenir.

```

Status = XGpio_Initialize(InstancePtr, DeviceId);
if (Status != XST_SUCCESS) {
    return XST_FAILURE;
}

Status = GpioSetupIntrSystem(IntcInstancePtr, InstancePtr, DeviceId,
                             IntrId, IntrMask);
if (Status != XST_SUCCESS) {
    return XST_FAILURE;
}

flag = 0;
delay = 0;

while(!flag && (delay < INTR_DELAY)) {
    int *gpioData_1;
    int *gpioData_2;

    gpioData_1 = (int *) 0x41200000;
    gpioData_2 = (int *) 0x41200008;

    *gpioData_1 = 10;
    *gpioData_2 = 20;

    sayi1 = *gpioData_1;
    sayi2 = *gpioData_2;

    delay++;
}

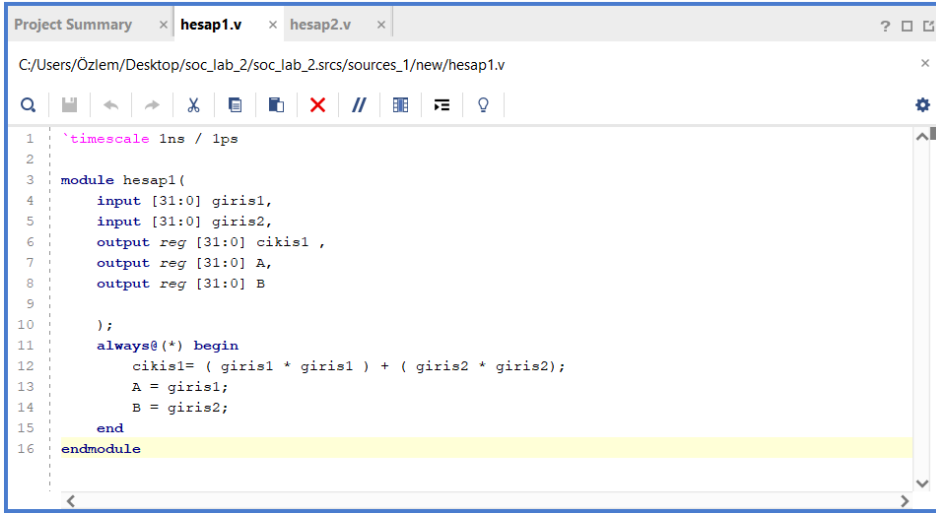
GpioDisableIntr(IntcInstancePtr, InstancePtr, IntrId, IntrMask);

*DataRead = flag;

return Status;

```

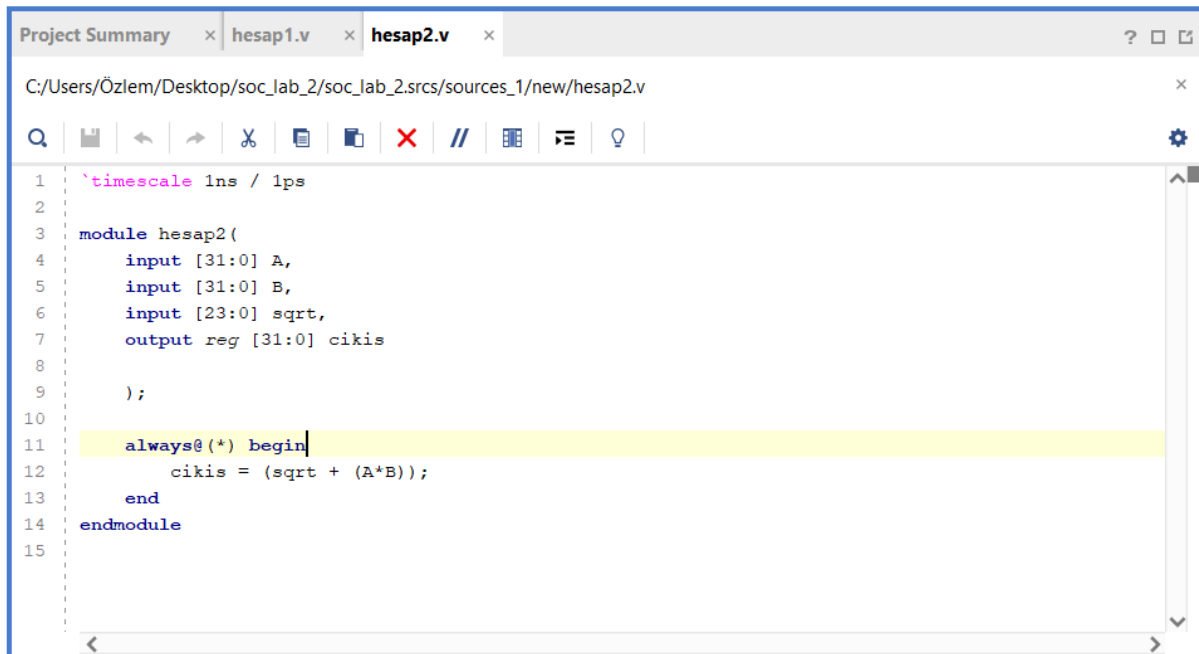
AXI GPIO 'u Vitis kısmında tanımlıyoruz 0.adrese gpioData\_1'i 8.adrese gpioData\_2'i atıyoruz gpioData\_1'e 10 gpioData\_2'e 20 veriyoruz. Ve bu dataları output olarak hesaplama modülüne yolluyoruz.



```
Project Summary x hesap1.v x hesap2.v x
C:/Users/Özlem/Desktop/soc_lab_2/soc_lab_2.srscs/sources_1/new/hesap1.v

1 `timescale 1ns / 1ps
2
3 module hesap1(
4     input [31:0] giris1,
5     input [31:0] giris2,
6     output reg [31:0] cikis1 ,
7     output reg [31:0] A,
8     output reg [31:0] B
9
10 );
11 always@(*) begin
12     cikis1= ( giris1 * giris1 ) + ( giris2 * giris2);
13     A = giris1;
14     B = giris2;
15 end
16 endmodule
```

Hesap1.v dosyasının içine ürettiğimiz dataları 32 bitlik input giriş1 ve input giriş2 olarak alıyoz. Çıkış1 outputuna  $giris1^2 + giris2^2$ 'i atıyoruz. Sonra giriş1'i A değişkenine, giriş2'yi B değişkenine atıyoruz. Daha sonra tekrar hiyerarşiye döndüğümüzde bu hesap1de ürettiğimiz A ve B'yi hesap2'ye atıyoruz. Sonra bu hesap1de üretilen cikis1 değişkenini cordic'e atıyoruz. Daha sonra cordicte bunun karekökü hesaplanır ve sqrt değişkenine atanır. Bu sqrt değişkeni ni de hesap2'ye gönderiyoruz.



```
Project Summary x hesap1.v x hesap2.v x
C:/Users/Özlem/Desktop/soc_lab_2/soc_lab_2.srscs/sources_1/new/hesap2.v

1 `timescale 1ns / 1ps
2
3 module hesap2(
4     input [31:0] A,
5     input [31:0] B,
6     input [23:0] sqrt,
7     output reg [31:0] cikis
8
9 );
10
11 always@(*) begin
12     cikis = (sqrt + (A*B));
13 end
14 endmodule
15
```

Hesap2 input olarak hesap1den A ve B değişkeni ve cordicden gelen sqrt değişkenini alır. Output olarak cikis isminde 32 bitlik bir çıktı verir. Bu çıktı =  $((A*B) + \text{sqrt})$ . Daha sonra bu çıkış 2. Bir cordice atılır. Cordicte kare kökü alınıp axi\_gpio'ya geri gönderilir. AXI GPIO IP'sinin girişi değişeceği için interrupt üretir. PS'de otomatik olarak interrupt fonksiyonuna dallandı ve sayı buradan okunabilir hale geldi.

```

void GpioHandler(void *CallbackRef)
{
    XGpio *GpioPtr = (XGpio *)CallbackRef;

    printf(" Hesaplama Sonucu: %d", sayi1);

    flag = 1;

    /* Clear the Interrupt */
    XGpio_InterruptClear(GpioPtr, GlobalIntrMask);
}

```

Az önce de söylediğimiz dallanma ile Xgpio'nun içine giriyor ve hesaplanan sayı bastırılıyor.

```

COM4 - PuTTY
Press button to Generate Interrupt
Successfully ran Gpio Interrupt Tapp Example
Hesaplama Sonucu: 14

```

$$C = \sqrt{\sqrt{A^2 + B^2} + A*B}$$

$$14,911 = \sqrt{\sqrt{10^2 + 20^2} + 10*20}$$

#### IV. SONUÇLAR

ZYNQ mimarisine sahip olan PYNQ geliştirme kartı üzerinde proje geliştirilmiştir. ZYNQ'in PS (Processor) bölümü, tasarlanacak özel bir modüle verileri besleyip, sonucunu alacak şekilde tasarlanmıştır. iki sayı girişi verilip, beklenen çıktının aynı sonucu elde edilip edilmediği kontrol edilmiştir. 10 ve 20 sayıları verilmiştir. Hesaplamanın sonunda beklenen sonuç olan 14'e ulaşılmıştır.

## PROJE EKİBİ

İREM KALKANLI (PROJE EKİP SORUMLUSU):

OKUL NUMARASI:190301007

DOĞUM TARİHİ:15.01.2000

DOĞUM YERİ: İSTANBUL

MEZUN OLDUĞU LİSE: ATAŞEHİR 3 DOĞA KOLEJİ

DENİZ UZUN:

OKUL NUMARASI:190301015

DOĞUM TARİHİ:08.04.2001

DOĞUM YERİ: İSTANBUL

MEZUN OLDUĞU LİSE: KAVACIK UĞUR ANADOLU LİSESİ

ÖZLEM ÇALI:

OKUL NUMARASI:190301002

DOĞUM TARİHİ:19.05.2000

DOĞUM YERİ: HATAY

MEZUN OLDUĞU LİSE: NECMİ ASFUROĞLU ANADOLU LİSESİ

İREM BOZKURT:

OKUL NUMARASI:190302010

DOĞUM TARİHİ:04.11.1998

DOĞUM YERİ: ADIYAMAN

MEZUN OLDUĞU LİSE: ÖZEL BİL KOLEJİ FEN LİSESİ

CEYDA UYMAZ:

OKUL NUMARASI:200301503

DOĞUM TARİHİ:26.08.2000

DOĞUM YERİ: İSTANBUL

MEZUN OLDUĞU LİSE: CELAL ARAS ANADOLU LİSESİ

## REFERANS DOSYALAR

<https://github.com/ozlemcali/CORDIC-Tabanl-HW-SW-CoDesign>  
<https://www.youtube.com/watch?v=onF9bH80pq0>

#### KAYNAKLAR

- [1] Levent, Vecdi Emre (2020) “ZNYQ Mimarisi”, *System on Chip (SOC) Design -Ders Notları*.
- [2] Levent, Vecdi Emre (2020) “PL/PS CoProcessing”, *System on Chip (SOC) Design -Ders Notları*.
- [3] Levent, Vecdi Emre (2020) “Donanım Hızlandırıcı Projesi”, *System on Chip (SOC) Design-Ders Notları*.
- [4] Levent, Vecdi Emre (2020) “Interrupt’lar”, *System on Chip (SOC) Design-Ders Notları*.
- [5] Levent, Vecdi Emre (2020) “Interfaces II”, *System on Chip (SOC) Design-Ders Notları*.