

Petalinux Tabanlı CORDIC 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. Bu donanım tasarımı PL tarafında hazırlandıktan sonra Petalinux ortamından veriler aktarılacaktır.

Anahtar Kelimeler — FPGA, CPU, PYNQ, CORDIC

Abstract— The project where Xilinx, as the hardware developer, transferred the budget hardware accelerator project of the CORD (Coordinate Rotation DIgital Computer) IP training and hardware. Once this hardware design is prepared for PL, Petalinux will be handed over.

Keywords — FPGA, CPU, PYNQ, CORDIC

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. Bu donanım tasarımı PL tarafında hazırlandıktan sonra Petalinux ortamından veriler aktarılacaktır.

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.

2) Putty

Putty ağ üzerindeki Linux işletim sistemli sunucunuza/makinanıza terminal üzerinden bağlantı sağlayan boyut olarak oldukça küçük ama güçlü ücretsiz ve açık kaynak kodlu bir yazılımdır. Putty ile bağlantı yaptığınız sunucunuzu terminal üzerinden çeşitli komutlar ile yönetebilirsiniz.

3) Ubuntu

Ubuntu, Linux tabanlı özgür ve ücretsiz bir işletim sistemidir. Bilgisayarlar, sunucular ve akıllı telefonlara yönelik olarak geliştirilmektedir. Ubuntu projesi Linux ve özgür yazılımın, bilgisayar kullanıcılarının günlük yaşamının bir parçası haline gelmesi amacıyla başlatılmıştır.

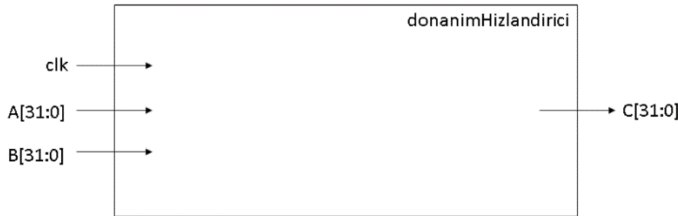
4)WinSCP

Özellikle güvenlik bakımından en gelişmiş uygulamalardan olan WinSCP içerisinde, SCP Protokollerini SSH desteklenmesini sağlar, FTP ile WebDav Protokollerini de içerisinde barındırır. Görsel kullanıcı ara yüzü gelişmiş olup, Windows, entegre bir şekilde çalışmaktadır. Dosya erişimi kolay ve rahattır. Ayrıca toplu dosya komutları, komut dosya satır ara yüzleri, .NET Desteklidir. Dizin ve PUTTY ile senkronizasyonları mevcut olup Dahili text editörünü de içerisinde barındırmaktadır. Güvenlik bakımından Kayıtlı şifreleri genel bir şifreyle güvenliğini üst seviyeye çıkartmakla beraberinde Public key, Kerberos gibi şifreleme yöntemlerini de bu uygulamada bulunmaktadır.

III. KULLANILAN YAZILIM

ZYNQ mimarisine sahip olan PYNQ geliştirme kartı üzerinde proje geliştirilecektir. ZYNQ’ın PS (Processor) bölümü, tasarlanacak özel bir modüle verileri besleyip, sonucunu alacak şekilde tasarlanacaktır. Özel modülün giriş ve çıkışları aşağıda verilmektedir.

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

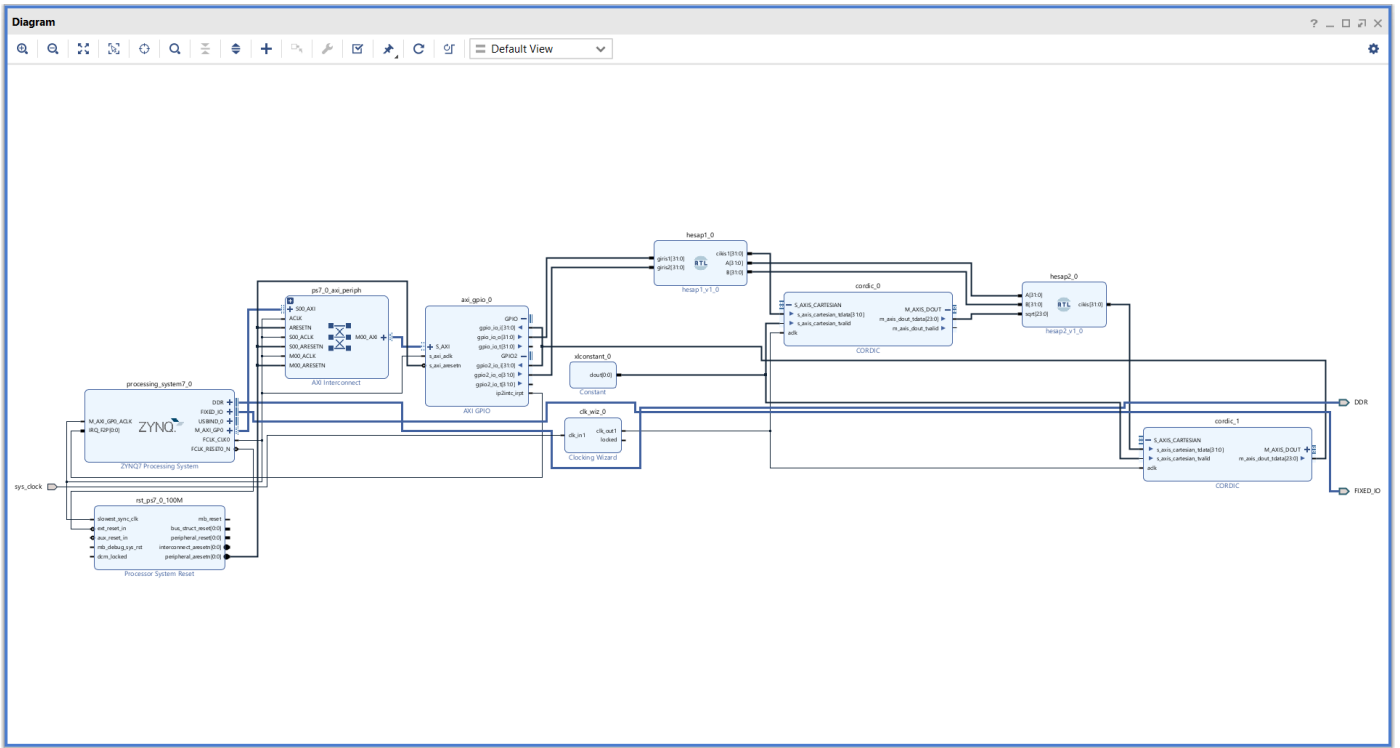


Bu modül içerisinde aşağıda verilen aritmetik işlemi yapan donanımı içermelidir.

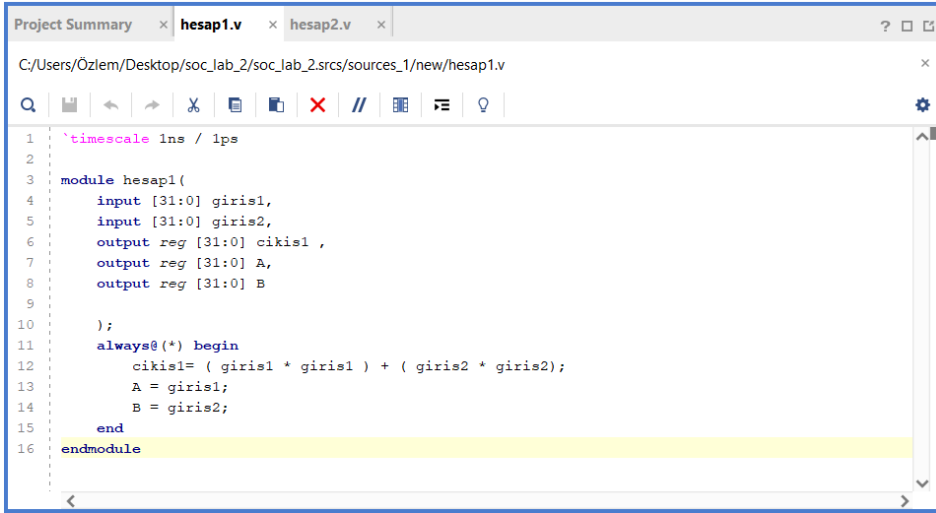
$$C = \text{SQRT}(\text{SQRT}(a^2 + B^2) + A * B)$$

SQRT işlemi için CORDIC IP'si kullanılabilir. CORDIC IP'sinde bulunan SQRT fonksiyonu için unsigned integer seçeneği seçilebilir.

AXI GPIO IP'si ile tasarlanan modülün giriş ve çıkışlarına bağlanmalıdır. PS tarafında A ve B sayıları örnek olarak 10 ve 20 olarak ayarlanıp giriş verilip, sonuç doğru üretildiğinde geriye değer alınmalıdır.

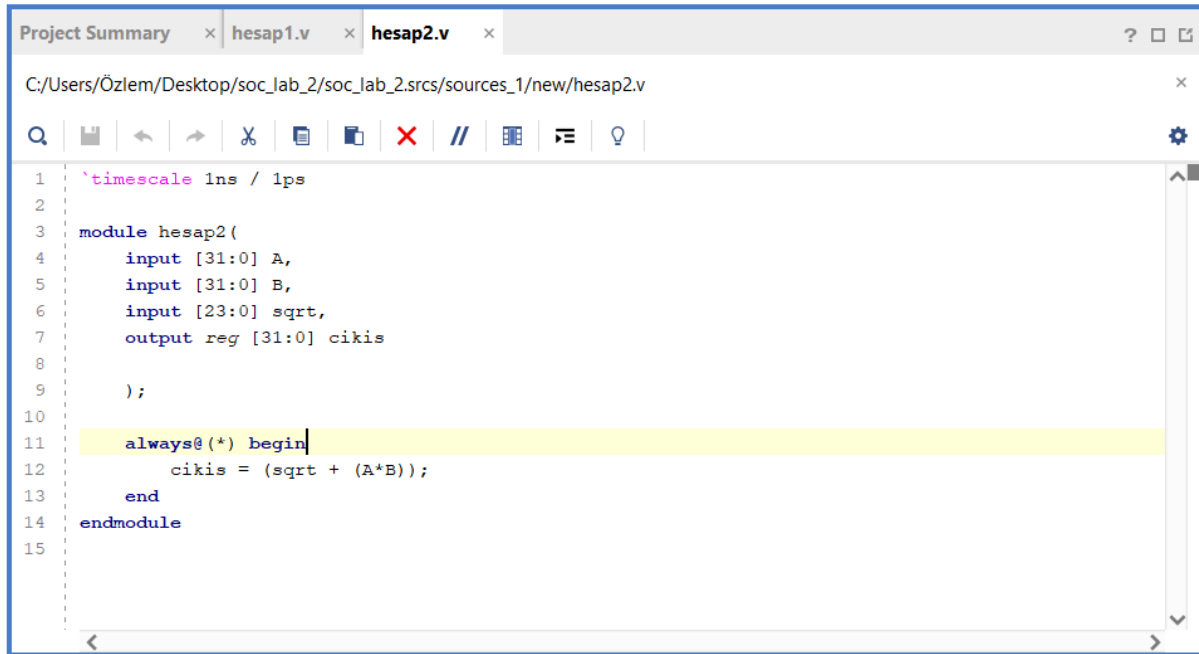


Öncelikle Zynq mimarisi eklenir. Sonra axi interconnect ve processor system reset de eklenir. Daha sonra AXI GPIO eklenir. AXI GPIO 'u Vitis kısmında tanımlıyoruz 0.adrese gpioData_1'i 8.adrese gpioData_2'yi atıyoruz gpioData_1'e 10 gpioData_2'e 20 veriyoruz. Ve bu dataları output olarak hesaplama modülüne yolluyoruz.



```
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 alıyouz. 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.



```
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.

Oluşturulan blok tasarımın ardından derlendiğinde ortaya ortaya çıkan .xsa uzantılı dosya linux ortamında yapıştırılır. Linux ortamında terminal ekranı açılarak Petalinux Tool'ları çalıştırılır.

```
ubuntu@ubuntu: ~/Desktop/Embedded Systems/proje
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje$ source /home/ubuntu/Desktop/petalinux/settings.sh
Petalinux environment set to '/home/ubuntu/Desktop/petalinux'
WARNING: /bin/sh is not bash!
bash is Petalinux recommended shell. Please set your default shell to bash.
WARNING: This is not a supported OS
INFO: Checking free disk space
INFO: Checking installed tools
INFO: Checking installed development libraries
INFO: Checking network and other services
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje$ echo $PETALINUX
/home/ubuntu/Desktop/petalinux
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje$
```

Petalinux çalıştırıldıktan sonra düzgün çalıştırılıp çalıştırılmadığını control etmek için echo komutu kullanıldı.

```
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje/proje$ petalinux-create -t project
-n proje --template zynq
INFO: Create project: proje
find: '/home/ubuntu/Desktop/Embedded': No such file or directory
find: 'Systems/proje/proje/proje': No such file or directory
environment: line 712: pushd: too many arguments
environment: line 717: popd: directory stack empty
INFO: New project successfully created in /home/ubuntu/Desktop/Embedded Systems/p
roje/proje/proje
```

Proje taslağı oluşturuldu.

```

ubuntu@ubuntu: ~/Desktop/Embedded Systems/proje/proje/proje$ petalinux-config --get-hw-descript
ion ../../
[INFO] Sourcing buildtools
INFO: Getting hardware description...
INFO: Renaming design_1_wrapper.xsa to system.xsa
[INFO] Generating Kconfig for project
[INFO] Menuconfig project

*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] Sourcing buildtools extended
[INFO] Extracting yocto SDK to components/yocto. This may take time!
[INFO] Sourcing build environment
[INFO] Generating kconfig for Rootfs
[INFO] Silentconfig rootfs
[INFO] Generating plnxtool conf
[INFO] Adding user layers
[INFO] Generating workspace directory
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje/proje/proje$

```

PL kısmındaki donanıma göre Petalinuxun kendini ayarlaması.

```

ubuntu@ubuntu:~/Desktop/Embedded Systems/proje/proje/proje$ petalinux-config -c rootfs
[INFO] Sourcing buildtools
[INFO] Silentconfig project
[INFO] Generating kconfig for Rootfs
[INFO] Menuconfig rootfs

*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[INFO] Generating plnxtool conf
[INFO] Successfully configured rootfs

```

Komut -> **petalinux-config -c rootfs** Linuxu oluştururken root file system'in build ayarlarını yaptığımız pencereyi açar.

```

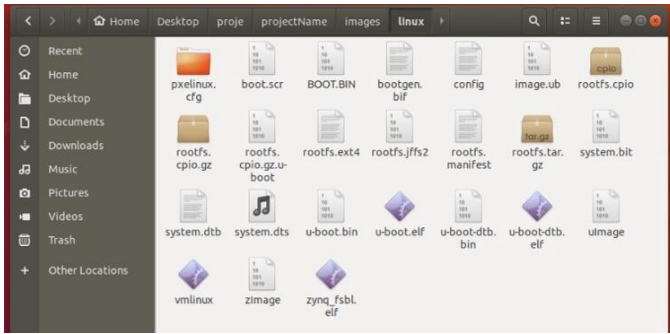
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje/proje/proje$ petalinux-config -c kernel
[INFO] Sourcing buildtools
[INFO] Silentconfig project
[INFO] Sourcing buildtools extended
[INFO] Sourcing build environment
[INFO] Generating kconfig for Rootfs
[INFO] Silentconfig rootfs
[INFO] Generating plnxtool conf
[INFO] Generating workspace directory
[INFO] Configuring: kernel
[INFO] bitbake virtual/kernel -c cleansstate
NOTE: Started PRServer with DBfile: /home/ubuntu/Desktop/Embedded_Systems/proje/proje/build/ca
che/prserv.sqlite3, IP: 127.0.0.1, PORT: 35937, PID: 12446
WARNING: Host distribution "ubuntu-22.04" has not been validated with this version of the bui
ld system; you may possibly experience unexpected failures. It is recommended that you use a
tested distribution.
Loading cache: 100% | ETA: --:--:--
Loaded 0 entries from dependency cache.
Parsing recipes: 100% |#####| Time: 0:04:42
Parsing of 3476 .bb files complete (0 cached, 3476 parsed). 5133 targets, 268 skipped, 0 mask
ed, 0 errors.
NOTE: Resolving any missing task queue dependencies

```

Komut -> **petalinux-config -c kernel** Petalinux'u build ederken kernel ayarlarını bu pencereden yapmamızı sağlar.

```
ubuntu@ubuntu:~/Desktop/Embedded Systems/proje/proje/proje$ petalinux-build
[INFO] Sourcing buildtools
[INFO] Building project
[INFO] Sourcing buildtools extended
[INFO] Sourcing build environment
[INFO] Generating workspace directory
INFO: bitbake petalinux-image-minimal
NOTE: Started PRServer with DBfile: /home/ubuntu/Desktop/Embedded_Systems/proje/proje/build/cache/prserv.sqlite3, IP: 127.0.0.1, PORT: 39627, PID: 228565
WARNING: Host distribution "ubuntu-22.04" has not been validated with this version of the build system; you may possibly experience unexpected failures. It is recommended that you use a tested distribution.
Loading cache: 100% |#####| Time: 0:00:02
Loaded 5128 entries from dependency cache.
Parsing recipes: 100% |#####| Time: 0:00:01
Parsing of 3476 .bb files complete (3471 cached, 5 parsed). 5133 targets, 268 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies
NOTE: Fetching uninative binary shin file:///home/ubuntu/Desktop/Embedded_Systems/proje/proje/components/yocto/downloads/uninative/5ec5a9276046e7eceeac749a18b175667384e1f445cd4526300a41404d985a5b/x86_64-nativesdk-libc.tar.xz;sha256sum=5ec5a9276046e7eceeac749a18b175667384e1f445cd4526300a41404d985a5b (will check PREMIRRORS first)
WARNING: Your host glibc version (2.35) is newer than that in uninative (2.33). Disabling uninative so that sstate is not corrupted.
Initialising tasks: 100% |#####| Time: 0:00:10
```

Petalinux build işlemi.



1. partiton'a koymak için boot.bin ve image.ub dosyasını ekledik ve 2. Patiton'da bir linux dağıtımı olan Yocto'nın kaynak dosyalarını ekledik (rootfs.tar.gz). Ayrıca gcc derleyicisi de koda eklendi.

```
COM7 - PuTTY
method.
Fri Mar 9 14:34:47 UTC 2018
hwclock: Cannot access the Hardware Clock via any known method.
hwclock: Use the --verbose option to see the details of our search for an access
method.
INIT: Entering runlevel: 5
Configuring network interfaces... done.
Starting system message bus: dbus.
Starting haveged: haveged: command socket is listening at fd 3
haveged: haveged starting up

Starting Dropbear SSH server: dropbear.
hwclock: Cannot access the Hardware Clock via any known method.
hwclock: Use the --verbose option to see the details of our search for an access
method.
Starting internet superserver: inetd.
Starting syslogd/klogd: done
Starting tcf-agent: OK

PetaLinux 2021.2 projectName ttyPS0

root@projectName:~#
```

Com7 hattından petalinuxu ayağa kaldırdık.

IP ayarlarını düzenle

El ile girilen

IPv4

☒ Açık

IP adresi

192.168.2.9

Alt ağ maskesi

255.255.255.0

Ağ geçidi

Tercih edilen DNS

HTTPS üzerinden DNS

Kapalı

Alternatif DNS

Kaydet

İptal

Fpga ve bilgisayar arasında iletişim olabilmesi için ip atamaları yapıldı ve iletişimde bilgisayarın ipsini static olarak yukarıdaki değere atadık.


```
COM7 - PuTTY
root@projectName:~# root
-sh: root: command not found
root@projectName:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:0a:35:00:1e:53 brd ff:ff:ff:ff:ff:ff
    inet 192.168.2.8/24 scope global eth0
        valid_lft forever preferred_lft forever
    inet6 fe80::20a:35ff:fe00:1e53/64 scope link
        valid_lft forever preferred_lft forever
3: sit0@NONE: <NOARP> mtu 1480 qdisc noop state DOWN group default qlen 1000
    link/sit 0.0.0.0 brd 0.0.0.0
root@projectName:~#
```

Proje kapsamında bilgisayar ve çip arasındaki haberleşmenin oluşturduğu engel durumunu kaldırmak için Ethernet ara yüzü kullanılmıştır. Bu yüzden çipe “ip.a” kodu ile statik ip adresi atadık.
vi /etc/network/interfaces -> düzenlenmesi için bu komut kullanılmıştır.

```
COM7 - PuTTY
/etc/network/interfaces -- configuration file for ifup(8), ifdown(8)

# The loopback interface
auto lo
iface lo inet loopback

# Wireless interfaces
iface wlan0 inet dhcp
    wireless_mode managed
    wireless_essid any
    wpa-driver wext
    wpa-conf /etc/wpa_supplicant.conf

iface atml0 inet dhcp

# Wired or wireless interfaces
auto eth0
iface eth0 inet static
    address 192.168.2.8
    netmask 255.255.255.0
    network 192.168.2.0
    gateway 192.168.2.1
iface eth1 inet dhcp
- /etc/network/interfaces 1/35 2%
```

Bu işlemleri yaptıktan sonra bilgisayar ile petalinux arasında ssh ile iletişim kurmak için gerekli yollar oluşturuldu.

```
#define TERMINAL "/dev/ttyPS0"
#include <errno.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <termios.h>
#include <unistd.h>
#include <sys/mman.h>
#include <math.h>
#include <time.h>

int num1[255];
int num2[255];
int i;

int main()
{
    unsigned int gpio_size = 0x8000;
    off_t gpio_pbase = 0x41200000;
    long long *gpio64_vptr;
    long long *bram64_vptr;
    int fd2;

    char *portname = TERMINAL;
    int fd;
    int wlen;
    char *xstr = "Hesaplanacak degerleri giriniz: \n";
    int xlen = strlen(xstr);
    fd = open(portname, O_RDWR | O_NOCTTY | O_SYNC);
    if (fd < 0)
    {
        printf("Error opening %s: %s\n", portname, strerror(errno));
        return -1;
    }
    wlen = write(fd, xstr, xlen);
    if (wlen != xlen)
    {
        printf("Error from write: %d, %d\n", wlen, errno);
    }

    do
    {
        unsigned char buf[80];
        int rdlen;
```

```

rdlen = read(fd, buf, sizeof(buf) - 1);
if (rdlen > 0)
{
    buf[rdlen] = 0;
    printf("%s \n", buf);

    int flag = 0;
    for (i = 0; i < strlen(buf); i++)
    {
        if (buf[i] == ' ')
        {
            flag = 1;
        }
        else
        {
            if (flag == 0)
            {
                num1[i] = buf[i] - '0';
            }
            else
            {
                num2[i] = buf[i] - '0';
            }
        }
    }
    int val1 = num1[1] + num1[0] * 10;
    int val2 = num2[4] + num2[3] * 10;

    if ((fd2 = open("/dev/mem", O_RDWR | O_SYNC)) != -1)
    {
        gpio64_vptr = (long long *)mmap(NULL, gpio_size, PROT_READ | PROT_WRITE,
MAP_SHARED, fd2, gpio_pbase);
        *(gpio64_vptr) = val1;
        *(gpio64_vptr + 1) = val2;

        int pres = *(gpio64_vptr + 1);
        printf("%d \n:", pres);

        char res[20];
        sprintf(res, "%d", pres);
        int reslen = strlen(res);
        write(fd, res, reslen);

        close(fd2);
    }
}

```

```

    }
}
else if (rdlen < 0)
{
    printf("Error from read: %d: %s\n", rdlen, strerror(errno));
}
else
{
    /* rdlen == 0 */
    printf("Timeout from read\n");
}

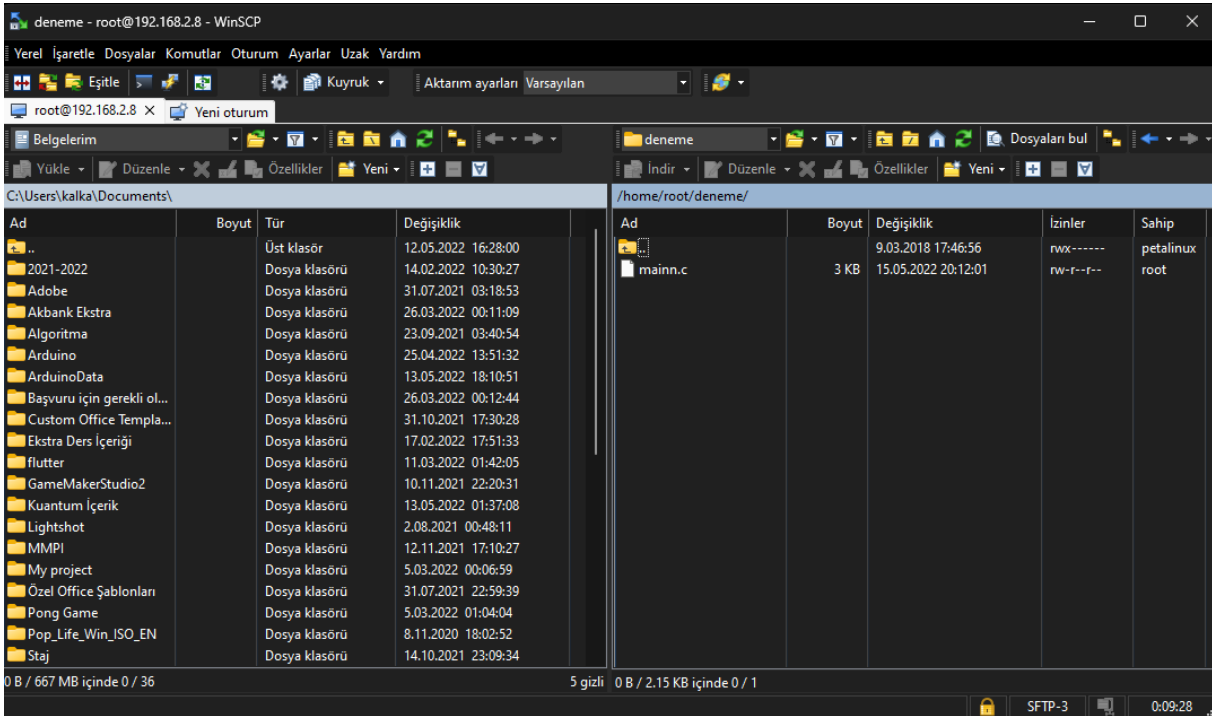
} while (1);
}

```

Amacı Petalinux ile bilgisayarın iletişimi için gerekli olan protokol c kodu ile yazıldı.

Kodu yazarken sanal adresleme kullanılır.

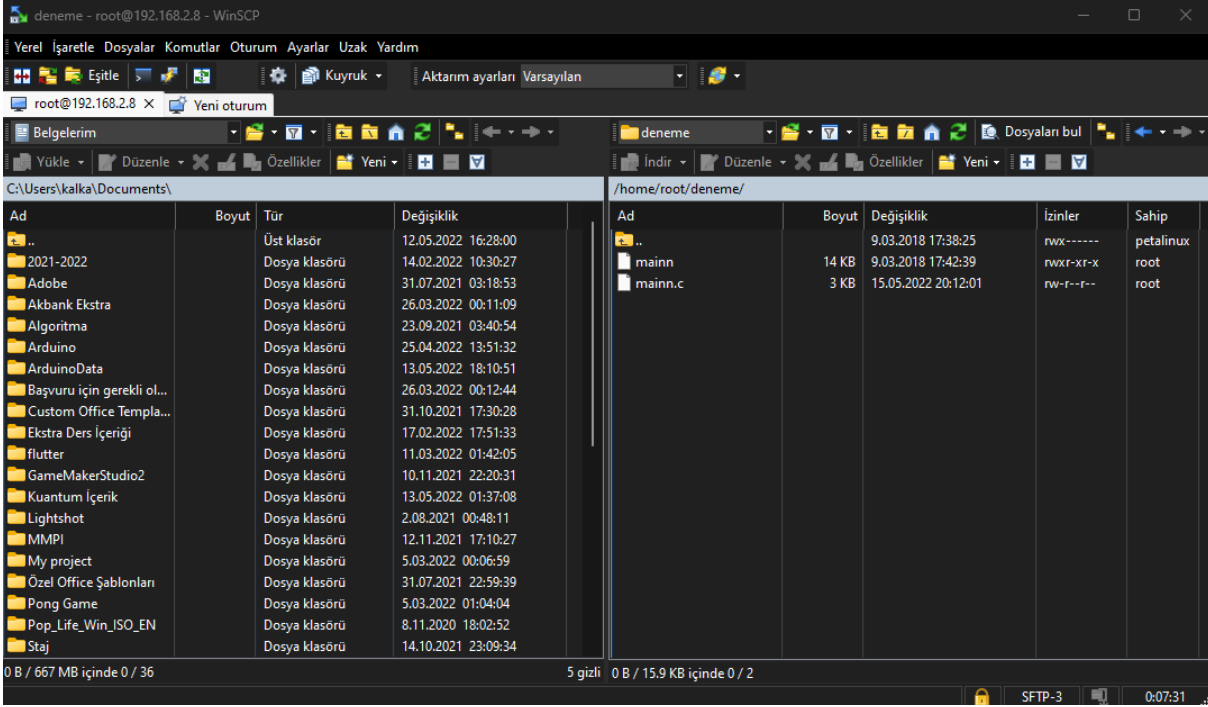
İlk olarak gpio'nun fiziksel adresini koda ekledik daha sonra 32 bitlik sayı okumak istediğimizi ekledik. Pynq'ten bilgisayara bilgilendirme mesajı bastırdık. Bunu Ethernet arayüzünü kullanarak yaptık. Pointer ile mmap sanal adresine değer atanır. Sonra kullanıcıdan gelen değer (bilgisayar üzerinden C# ile) buffer array'e yazılır. Buffer arrayde sayıların arasında tire olacak şekilde verileri alır. Char olarak gelen veriler Gpio IP integer ile çalıştığı için integer değere dönüştürülür. val1 ve val2 gpio ip'sinin girişlerine gönderilir. Petalinux fiziksel adreslerle değil sanal adreslerle çalışan bir işletim sistemidir. Bu yüzden mmap komutuna Gpio Ip'sinin fiziksel adresini verilir ve sanal adresi elde edilir. Gpio'nun sanal giriş ve çıkışlarına veri yazılmış olur. Gpio'nun 0. adresine kullanıcıdan alınan birinci değer, 8. Adresine de ikinci değeri gönderildi. İkinci adrese ulaşmak için "+1" yazıldı. Hesaplama sonucu Gpio'nun 8. adresinden PL tarafından hesaplanan değeri geri aldık. Yani 10 ve 20 gönderdiğimiz değerlerin adresini okuduğumuzda sonuçta 14'ü elde ederiz.



WinSCP aracı sayesinde petalinux ile bilgisayar arası okuma yazma yapan c kodunu kolaylıkla ekledik.

```
192.168.2.8 - PuTTY
root@projectName:~# cd deneme/
root@projectName:~/deneme# ls
mainn.c
root@projectName:~/deneme# gcc mainn.c -o mainn
root@projectName:~/deneme#
```

Putty uygulaması ile de eklediğimiz c kodunu derledik.

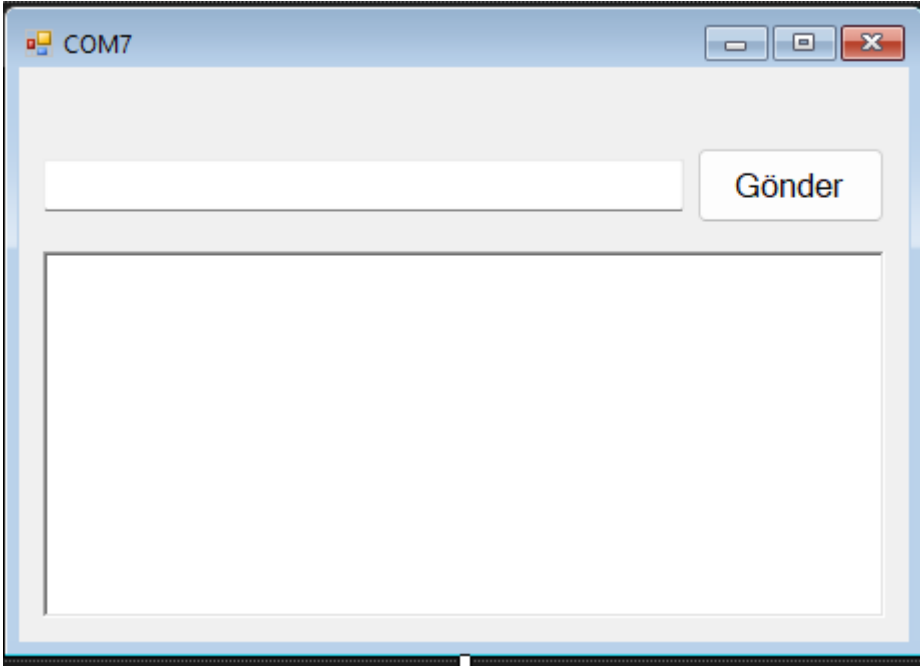


The WinSCP interface shows a local file system on the left and a remote file system on the right. The local file system is located at C:\Users\kalka\Documents\ and contains various folders and files. The remote file system is located at /home/root/deneme/ and contains a folder named 'deneme' and a file named 'mainn.c'.

Ad	Boyut	Tür	Değişiklik
..		Üst klasör	12.05.2022 16:28:00
2021-2022		Dosya klasörü	14.02.2022 10:30:27
Adobe		Dosya klasörü	31.07.2021 03:18:53
Akbank Ekstra		Dosya klasörü	26.03.2022 00:11:09
Algoritma		Dosya klasörü	23.09.2021 03:40:54
Arduino		Dosya klasörü	25.04.2022 13:51:32
ArduinoData		Dosya klasörü	13.05.2022 18:10:51
Başvuru için gerekli ol...		Dosya klasörü	26.03.2022 00:12:44
Custom Office Templa...		Dosya klasörü	31.10.2021 17:30:28
Ekstra Ders İçeriği		Dosya klasörü	17.02.2022 17:51:33
Flutter		Dosya klasörü	11.03.2022 01:42:05
GameMakerStudio2		Dosya klasörü	10.11.2021 22:20:31
Kuantum İçerik		Dosya klasörü	13.05.2022 01:37:08
Lightshot		Dosya klasörü	2.08.2021 00:48:11
MMPi		Dosya klasörü	12.11.2021 17:10:27
My project		Dosya klasörü	5.03.2022 00:06:59
Özel Office Şablonları		Dosya klasörü	31.07.2021 22:59:39
Pong Game		Dosya klasörü	5.03.2022 01:04:04
Pop_Life_Win_ISO_EN		Dosya klasörü	8.11.2020 18:02:52
Staj		Dosya klasörü	14.10.2021 23:09:34

Ad	Boyut	Değişiklik	İzinler	Sahip
..		9.03.2018 17:38:25	rwX-----	petalinux
mainn	14 KB	9.03.2018 17:42:39	rwXr-Xr-X	root
mainn.c	3 KB	15.05.2022 20:12:01	rw-r--r--	root

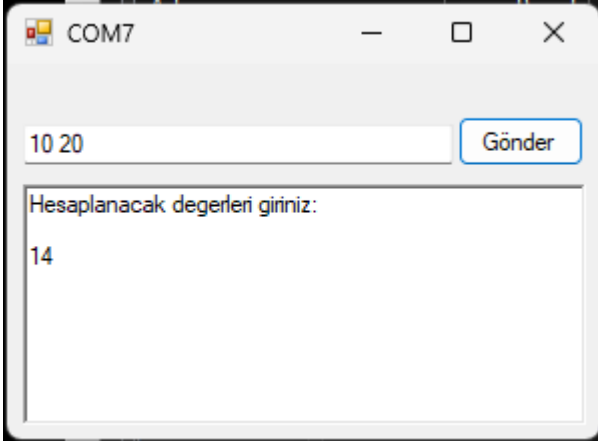
Derlendikten sonraki halinin WinSCP aracı ile görünümü



Girdi ve çıktıları gözlemlemek için C# programı gerçekleştirdi

```
root@projectName:~# devmem 0x41200000 32 0xA
root@projectName:~# devmem 0x41200008 32 20
root@projectName:~# devmem 0x41200008 32
0x0000000E
root@projectName:~#
```

Yukardaki çıktı ile fiziksel adres kullanılarak 10(0XA) ve 20 değerlerini atadık. PL kısmından istenen 14 (E) sonucu aldık (sayılar on altılık tabanda yazılmış.)



Putty uygulaması ile petalinuxe attığımız c kodunu derleyip çalıştırdıktan sonra C# kodunu çalıştırıp göndermek istediğimiz iki değeri yollayarak doğru sonuca ulaşırız.

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

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

IV. SONUÇLAR

Donanım hızlandırıcı olarak Xilinx'in CORDIC (COordinate Rotation DIgital Computer) IP'si kullanıldığı ve işlemcinin hesap yükünü donanım hızlandırıcıya aktardığı bir proje yapılmıştır. Bu donanım tasarımı PL tarafında hazırlandıktan sonra Petalinux ortamından veriler aktarılmıştır ve doğru çalıştığı gözlemlenmiştir.

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://www.youtube.com/watch?v=s1VmbWmU9RM>

<https://github.com/iremalkanli/Petalinux-Tabanli-Cordic-HW-SW-CoDesign>

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*.