İSTANBUL TEKNİK ÜNİVERSİTESİ
ELEKTRİK ELEKTRONİK FAKÜLTESİ

LEON3 MİKROİŞLEMCİSİNİN FPGA ÜZERİNDE GERÇEKLENMESİ

BİTİRME ÖDEVİ

MEHMET MURAT ÇETİN

040090440

Bölümü: Elektronik ve Haberleşme Mühendisliği Bölümü

Programı: Elektronik Mühendisliği

Danışmanı: Doç. Dr. Sıddıka Berna ÖRS YALÇIN

MAYIS 2014

ÖNSÖZ

Bitirme projem boyunca benden yardımlarını, vaktini ve desteğini esirgemeyen proje danışmanım Doç. Dr. Sıddıka Berna ÖRS YALÇIN'a teşekkürlerimi sunarım.

Benden proje boyunca desteklerini esirgemeyen Ahmet Çağrı BAĞBABA, Buse USTAOĞLU ve İnan ERDEM'e ve Gömülü Sistem Tasarım Laboratuvarı çalışanlarına ve arkadaşlarıma teşekkürü borç bilirim.

Ayrıca, hayatım boyunca yanımda olan değerli aileme sevgilerimi sunarım.

Mayıs 2014 Mehmet Murat ÇETİN

İÇİNDEKİLER

KISALTMALAR

FPGA : Field Programmable Gate Array
HDL : Hardware Description Language
SPARC : Scalable Processor Architecture
RISC : Reduced Instruction Set Computing
SoC : System on a Chip
AMBA : Advanced Microcontroller Bus Architecture
AHB : AMBA High-performance Bus
ASIC : Application Specific Integrated Circuit
IP core : Intellectual Property Core
GPL : GNU General Public License
APB : Advanced Peripheral Bus
DSU : Debug Support Unit
UART : Universal Asynchronous Receiver/Transmitter
JTAG : Joint Test Action Group
PCI : Peripheral Component Interconnect
USB : Universal Serial Bus
LUT : Look-up Table
DSP : Digital Signal Processor
ISE : Integrated Synthesis Environment
GUI : Graphical User Interface
GCC : GNU Compiler Collection
LTS : Long-Term Support

ŞEKİL LİSTESİ

LEON3 MİKROİŞLEMCİSİNİN FPGA ÜZERİNDE GERÇEKLENMESİ

ÖZET

Günümüzde kullanılan elektronik cihazların çoğunda mikroişlemci ya da FPGA bulunmaktadır. Bu iki yapının kendilerine özgü faydalarıyla beraber eksik kaldıkları noktalar da vardır. Bu sebeptendir ki, FPGA içerisinde mikroişlemci tasarlamak gün geçtikçe önem kazanmaktadır. Buna en önemli etkiyi işlemcinin yanına bir donanım eklediğinizde kart tasarımı tamamen değişebilmekteyken FPGA içerisindeki işlemcilerde bunun kod yazarak yapılabilmesidir. Bunun yanı sıra FPGA ile daha uğraştırıcı yollarla yapılabilen ve ya hiç yapılamayan bir işlemin işlemci yardımıyla çok daha kolay yapılmasına olanak sağlamaktadır.

FPGA içerisinde kullanılabilecek olan işlemcilerin de birçok çeşidi vardır. Birçok firma işlemcilerin kaynak kodlarına erişime izin vermemektedir. Diğer taraftan bazı organizasyonlar açık kaynak kodlu işlemciler piyasaya sürmüşlerdir. Desteklenmelerinin diğer işlemcilere göre daha az olmasına rağmen üzerinde geliştirme yapmaya tamamen açık olması bu işlemcilerin çok büyük bir avantajıdır. İşte bu işlemcilerden biri olan LEON başlangıçta Avrupa Uzay Araştırma ve Teknoloji Merkezi tarafından oluşturulmuştur. Daha sonra Gaisler firması tarafından geliştirilmiştir.

Bu projede LEON3 işlemcisinin bir FPGA kartına gömülmesinin aşamaları detaylı olarak anlatılmıştır. Projenin daha kolay anlaşılması için işlemcinin kullanılacak olan kısımların mimarisi anlatıldı. Sonrasında işlemcinin gömülmesi için gerekli olan bütün programların nasıl kuruldukları ve nasıl kullanıldıkları detaylı bir şekilde anlatıldı.

Projenin gerçeklenme aşamasında, LEON3 işlemcisi VHDL dosyalarının üretimi için öncelikle yapılandırıldı. Daha sonra yapılandırılan dosyalardan ISE projesi üretildi, sentezlendi ve FPGA'ye gömüldü. İşlemciyle önce JTAG bağlantısı yardımıyla daha sonra ethernet bağlantısı yardımıyla debug programı ile uygulama çalıştırıldı. Son olarak kart üzerindeki anahtarların durumu okundu ve ledler yakıldı.

IMPLEMENTATION OF LEON3 SOFT-CORE ON FPGA

SUMMARY

In today's worlds many electronic devices has microprocessor or FPGA. Even though, these two architectures have specific advantages, there are still some parts that these devices are not sufficient. Just for this reason, designing microprocessors is getting importance in FPGA. In that point the main advantage of FPGA processor is to code instead of putting new equipment on it. Besides, FPGA facilitates such challenging or unattainable issues in a softer and easy way.

There are many various processors which can use in FPGA. Many companies do not let to access source codes. On the other hand some organizations launch open source codes to the market. Even though these embedded processors are less supported by in compare with other processors, to be completely open to develop for these processors is an important advantage. Leon is one of the processors of this system which is designed by European Space Research and Technology Centre (ESTEC). Then, it is developed by Gaisler Company.

In this project, embedding of Leon3 processor in FPGA board is explained in detail. In order to understand clearly, processors' architecture of parts been used is explained. Later on, embedding of processor and the required programs explained clearly in terms of how they are established and how they are operated.

During the process of realization of the project, LEON3 processor is configured primarily for the generated of VHDL files. Then these configured the ISE project of the files were produced, synthesized and was embedded in FPGAs. Then the system is operated first with the help of JTAG connection then the help of Ethernet connection. Finally, the status of the switch on the card has been read and LEDs are turned on.

1. GİRİŞ

Hayatımızda kullandığımız birçok cihazda yazılım ve donanım tabanlı gömülü sistem tasarımları vardır ref. Bu tasarımlardan en efektif kullanılanları yazılım ve donanım tasarımını beraber gerektirir. İşte bu nedenle gömülü sistem uygulamalarında uzun hal FPGA'in içine işlemci gömmenin birçok avantajı bulunmaktadır. Uygulamaya göre çevrebirimleri seçilebilir hatta kullanıcı tanımlı çevrebirimleri kolaylıkla eklenebilir. Bir çeşit hafıza kontrol birimi kullanmak, FPGA gömülü işlemci sisteminin arayüz kullanılabilirliğini geliştirir. Aynı zamanda işlemcisi bulunan FPGA'ler de üretilmektedir. Bu alanda iki büyük şirket olan Xilinx ve Altera FPGA silikonu içinde gömülü işlemci içeren FPGA üretmektedir [1]. HDL dili ve ya netlist ile tarif edilen "soft" işlemcilerin bu "hard" işlemcilere göre değiştirilebilirlik, eskime, gerektiği kadar yapı kullanma ve fiyat kısma, donanım hızlandırıcılar gibi avantajları vardır [1]. Aynı zamanda bu işlemcinin sistem tasarımcısı tarafından geliştirmesinin gerekmesi ve bu işlemci normal bir işlemciyle yapılacak bir iş için pahalı olması gibi kayıpları da bulunmaktadır.

LEON3 işlemcisinin ref diğer işlemcilere, özellikle M~~m~~icroblaze, göre en büyük avantajı açık kaynak kodlu olmasıdır. Geliştirmeye açık olan bu işlemci ilerde bizi kısıtlayan bir durum çıktığında kaynak kodunda değişiklik yaparak bu durumları aşmamızı sağlamaktadır.

Projenin amacı LEON3 yazılımsal işlemcisinin FPGA içerisine gömülmesi ve çalıştığının gösterilmesidir. LEON3 kaynak uzun hal VHDL kaynak kodunun derlenmesi için Xilinx İ~~I~~SE yazılımı kullanıldı. Debug ref yazılımı olarak Grmon ref kullanıldı. Çalışmanın gösterilmesi için LEON3 işlemcisinin kullanılan kart üzerindeki arabirimlere erişimi ile ilgili bir program yazıldı. Program jtag bağlantısı ve ethernet bağlantısı ile test edildi.

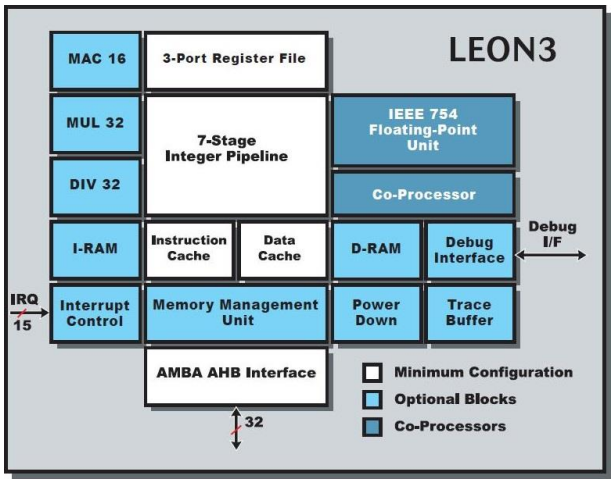
Formatted: Font color: Red

Formatted: Font color: Red

2. ÖNBİLGİLER

2.1. Leon3 Mimarisi

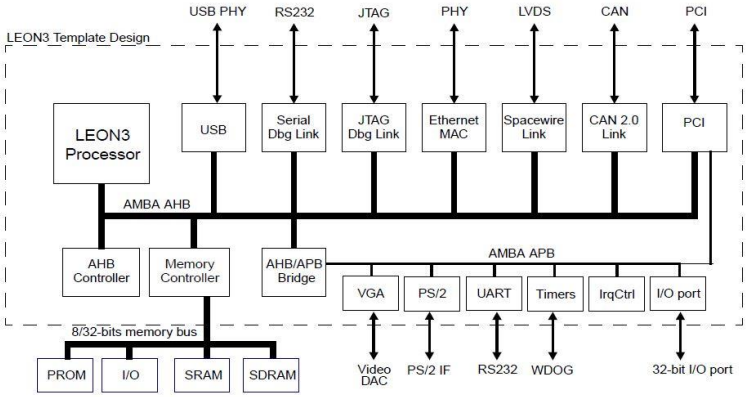
LEON, SPARC-V8 RISC yapısına sahip 32-bit bir mikroişlemci çekirdeğidir [2]. Bu çekirdek Şekil 2.1'de görüldüğü gibi değiştirilebilir bir yapıya sahiptir ve bir çip içinde sistem (system on a chip – SOC) (system-on-a-chip) tasarımlar için oldukça uygundur. Yüksek düzeyde değiştirilebilirlik performans, güç tasarrufu, silikon alanı ve para tasarrufunda tasarımcıya işlemciyi optimize etmede büyük olanaklar sağlar. Çekirdek arayüzü AMBA 2.0 AHB ref veri yoludur, ayrıca Gaisler Research IP kütüphanesi (GRLIB) ref tarafından sağlanan IP core (intellectual property core) ekle-kullan yöntemiyle desteklenmektedir. İşlemci FPGA'ya ve ASIC (Application Specific Integrated Circuit; Uygulamaya Özel Tümleşik Devre)'e etkin bir şekilde uygulanabilir. LEON3 çekirdeğinin uzay ve yüksek güvenilirlik beklenen alanlar için fault-tolerant (hatadan etkilenmez) versiyonu da bulunmaktadır. SPARC yapısının desteklenmesi ve ön değerlendirme ve prototipin kolaylaştırılması için işlemcinin ve bütün bağlantılı IP kütüphanelerin kaynak kodları araştırma, değerlendirme ve eğitim için GNU GPL (General Public License) lisansı altında sunulmaktadır. Ticari kullanıcılar için düşük maliyetli ticari lisans da var [2].



Şekil 2.1 : LEON3 işlemcisinin Değiştirilebilir Yapıları [2]

2.2. Sistem Veriyolu

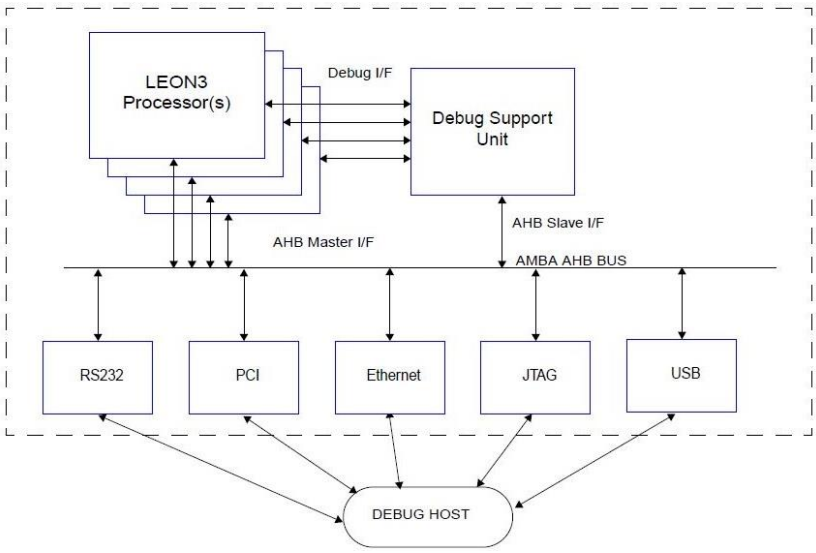
GRLIB veri yolu merkez olarak tasarlanmıştır, yani IP çekirdeklerinin çoğu birbirine şekil 2.2'deki gibi sistem veri yolu üzerinden bağlıdır [3]. AMBA-2.0 AHB/APB veri yolu bu işlemcide en çok kullanılan veri yollarından biri olduğundan dolayı seçilmiştir. Bu kullanım çokluğunu da ARM işlemciler sayesindeki piyasa üstünlüğü, detaylı olarak hazırlanması ve lisans kısıtlaması olmadan ücretsiz olarak kullanılabilmesi sağlamıştır [3].



Şekil 2.2 : Sistem veriyolu [3]

2.3. DSU(Debug Support Unit) (Hata Ayıklama Birimi)

Hedef cihazda hata ayıklamayı kolaylaştırmak için iş hattı (pipeline) boşta iken LEON3 işlemcisi bir hata ayıklama modu sağlar ve işlemci özel bir hata ayıklama ara yüzü tarafından kontrol edilir [4]. Hata ayıklama birimi AHB slave (yavru uçbirim) olarak davranır ve herhangi bir AHB master (ana uçbirim) tarafından erişilebilir. Bu sayede dış bir hata ayıklama anasistemi hata ayıklama birimine Şekil 2.3'deki gibi birçok ara yüz aracılığı ile erişebilir. Bu ara yüz seri UART (RS232), JTAG, PCI, USB, ve ya Ethernet olabilmektedir. Hata ayıklama birimi çoklu hata ayıklamayı destekler ve 16 işlemciye kadar işlem yapabilir [4].



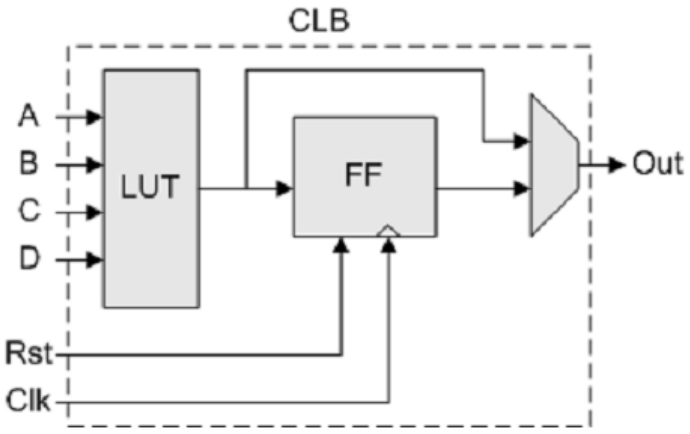
Şekil 2.3 : Sistemin hata ayıklama yapısı [4]

2.4. Sahada Programlanabilir Kapı Dizisi

Sahada Programlanabilir Kapı Dizinleri (field prog – FPGA) herhangi bir sayısal fonksiyonu gerçekleştirebilmek için kullanıcı tarafından programlanabilen tümleşik devrelerdir [5]. FPGA yönetilebilir anahtarların ve programlanabilir mantık hücrelerinin iki boyutlu olarak dizilmesiyle oluşturulur. Mantık hücreleri basit bir fonksiyonu gerçeklemek üzere yapılandırılabildiği gibi programlanabilir anahtarlar ile mantık hücreleri arasında bağlantılar kurulabilir. Bu şekilde mantık hücreleri ve anahtarların programlanmasıyla sayısal donanımlar gerçeklenir. Donanım tanımlama

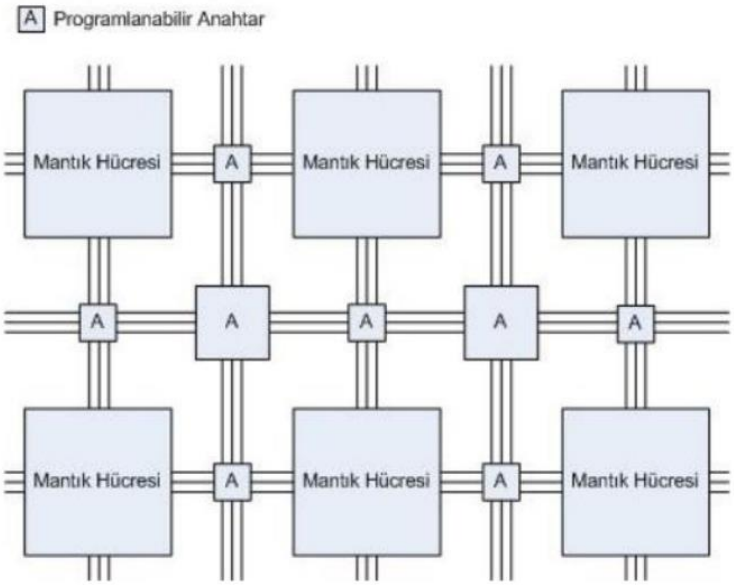
dilleri kullanılarak devrenin tasarımı yapıldıktan ve sentezlenmesinin ardından istenilen lojik hücre ve anahtar yapılandırılmasının yer aldığı veri dizisi kablo yardımıyla FPGA"A"ya gönderilerek devre gerçeklenmiş olur [6].

Mantık hücreleri Şekil 2.4'de görüldüğü gibi programlanabilir kombinezonsal devre ve bir adet D tipi flip-flop içerir.



Şekil 2.4 : Mantık Hücresi [6]

Programlanabilir ara bağlantılardan ve içyapısı Şekil 2.4'de gösterilen mantık hücrelerinden oluşan FPGA'nın genel yapısı Şekil 2.5'de gösterilmektedir.

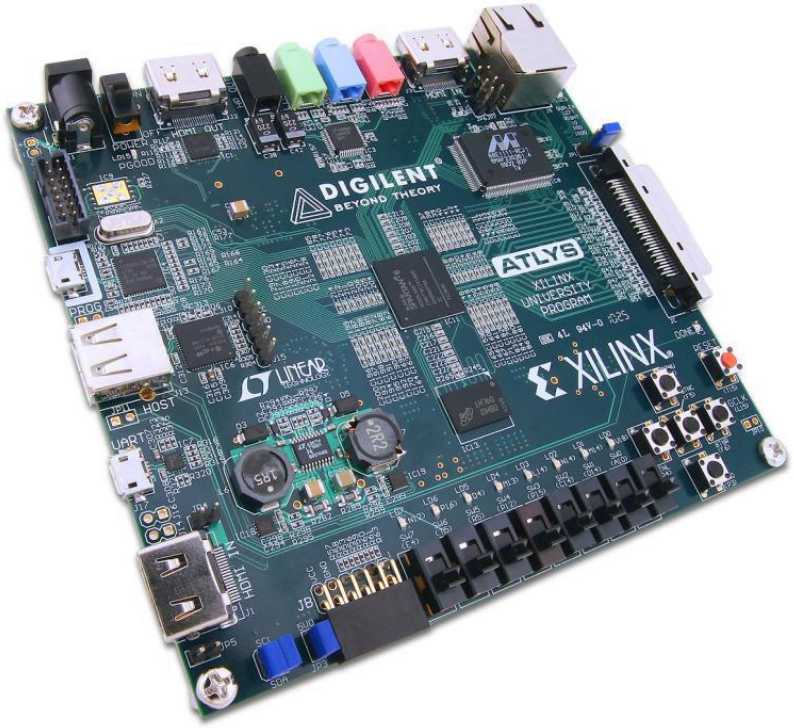


Şekil 2.5 : FPGA'in İçyapısı [6]

Mantık hücrelerindeki LUT (Look-up Table) yapılandırılabilen kombinezonsal devreleri gerçeklemek için kullanılmaktadır. LUT‟lar aslında bir mantık işlemini yerine getiren küçük belleklerdir. N girişli bir LUT 2 N boyutlu bellek elemanına karşılık düşmektedir. Binlerce LUT elemanı yan yana getirilerek daha karmaşık kombinezonsal fonksiyonlar gerçekleştirilmesine imkân tanır [7]. FPGA‟nın paralel işlem yapabilme kapasitesine sahip olması çok daha hızlı sistemlerin tasarlanmasına olanak sağlamaktadır. Bununla birlikte, mikroişlemciler de birer mantık devresi oldukları için FPGA üzerinde kullanılabilirler. Dolayısıyla tek bir tümleşik devre içerisinde kontrol birimi olarak hem işlemci hem de kullanıcıya özgü donanımsal fonksiyonları gerçekleyen fonksiyonlar tanımlamak mümkündür. Tüm sistemin aynı yerde yer alması bağlantılar arası gecikmeler azalacağından FPGA, üzerinde daha da hızlı sistemlerin tasarlanmasına da olanak sağlamaktadır. Bütün bu özelliklerin tasarım sırasında büyük esneklik sağlaması ve ayrıca FPGA’nın paralel işlem yapabilme kapasitesine sahip olduğundan ötürü bu tezin FPGA üzerinde tasarlanması tercih edilmiştir.

2.5. Atlys Spartan-6 FPGA Board

Çalışmada kullanılan kart üzerinde Xilinx firmasının Spartan-6 LX45 FPGA'yi bulunmaktadır. Bu FPGA, her biri 4 adet 6 girişli LUT ve 8 adet flip flop içeren 6822 dilim, 58 DSP dilimi içermektedir. Ayrıca 500 MHz'e kadar saat hızı sunmaktadır [8].

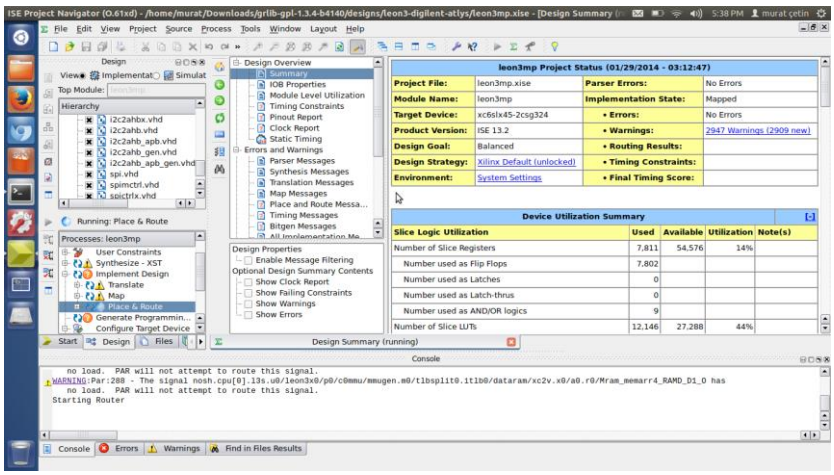


Şekil 2.6 : Kullanılan FPGA Kartı[8]

2.6. Xilinx ISE Ortamı

Tümleşik Yazılım Ortamı (Integrated Software Environment, ISE) FPGA'ları programlamak için kullanılmak üzere Xilinx firmasının geliştirdiği bir ara yüz yazılımıdır ref. ISE ortamı, donanım tanımlama dilleri ya da şematik çizimler sayesinde FPGA'da çalıştırmak üzere sistemler tasarlamaya olanak sağlar. ISE ortamında Verilog, VHDL gibi donanım tanımlama dilleriyle oluşturulan tasarım sentezleme ve gerçekleme aşamalarından geçirerek kablo aracılığıyla FPGA içine yerleştirilebilir. Ayrıca, donanım üzerinde çalıştırılmadan hata ayıklamak ve hatanın

nereden kaynaklandığını görmek üzere ISE ortamında test yapmaya da olanak sağlamaktadır. Ayrıca ISE kullanıcıya tasarladığı sistemlerin ne kadar hat gecikmesine sahip olduğunu, donanım üzerinde ne kadar yer kapladığını ve yapılan tasarımın FPGA'nın hangi bölgesine yerleştirileceğine kadar detaylı bilgiler sunmaktadır.



Şekil 2.7 : ISE Programının Görüntüsü

2.7. LEON3 Yapılandırma Aracı

LEON3 işlemcisinin içereceği özelliklerin ayarlanabileceği grafiksel kullanıcı arayüzüdür. Bu arayüzün çalışabilmesi için tck/tl uygulamasının kurulması gerekmektedir. Aşağıdaki kodlar sayesinde xconfig GUI(Graphical User İnterface) açılacak ve LEON3 şablon tasarımı değiştirilebilecektir.

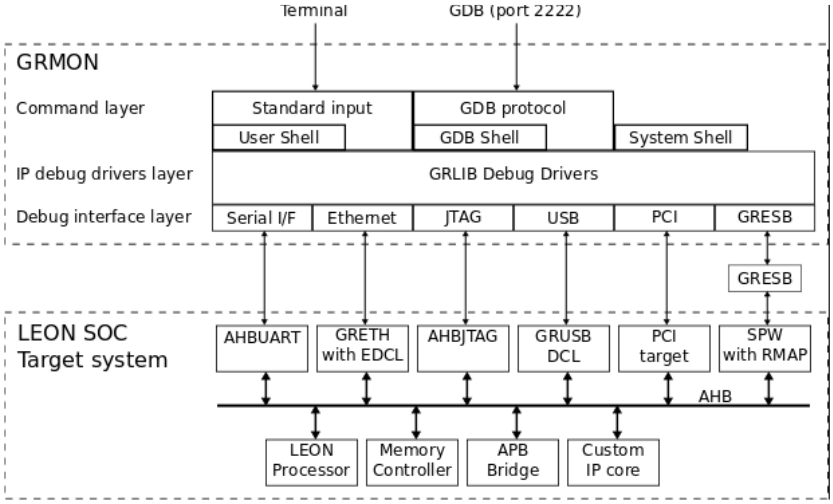
```
$ cd ..../designs/leon3-digilent-atlys
$ make xconfig
```

çevrebirimler düzenlenebilir.

2.8. Grmon

GRMON, LEON işlemcisi ve GRLIB IP Kütüphanesi tabanlı SOC tasarımları için genel bir hata ayıklama ekranıdır. LEON3 ve daha yeni sürümler destekleniyor [9]. GRMON ile bütün sistem hafızasında ve tutucularında okuma/yazma işlemi yapılabilir. LEON programları yüklenebilir ve çalıştırılabilir. Breakpoint ve watchpoint kontrolü yapılabilir. Hata ayıklama ekranı hedef donanım üzerindeki AHB

veriyolunda yazma ve okuma çevrimleri yapabilen özel arayüz ile bağlanır. Arayüz değişik şekillerde olabilir: LEON3/4 işlemcisi UART, 32-bit PCI, JTAG ve Ethernet üzerinden hata ayıklamayı destekler. Bu arayüzler Şekil 2.8'de gösterilmiştir. Hedef sistem üzerindeki donanımsal olarak gömülen hata ayıklama protokolü sayesinde LEON sisteminin hata ayıklaması için hiçbir yazılıma ihtiyaç duyamamaktadır [9].



Şekil 2.8 : Grmon Yapısının Görüntüsü [9]

2.9. BCC - Bare-C Cross-Compiler

BCC LEON3 işlemciler için geliştirilmiş bir çapraz derleyicidir [10]. GNU derleyici araçlarına ve Newlib standart C kütüphanesine dayanır. Çapraz derleyici sistemi görev olan ve olmayan C ve C++ uygulamaları derlemeye izin verir. SPARC V8 çarpma ve bölme yapılarını yanı sıra donanımsal ve yazılımsal kayan noktalı~~(floating point)~~ işlemlerini de destekler. Bu çapraz derleyici GNU GCC C/C++ derleyicisi v3.4.4, v4.4.2, Newlib C-kütüphanesi v1.13.1, LEON3 için düşük level I/O (giriş/çıkış) rutini, LEON3/4 için Mkprom prom-builder gibi paketleri içerisinde barındırır [10].

2.10. Gerekli Yazılımların Kurulumu

2.10.1. Ubuntu

Öncelikle Windows (Cygwin) üzerinden de bu işlemler yapılabilir ama Linux (özellikle Ubuntu) üzerinde yapılması tavsiye edilir ref. Ubuntu 32-bit işletim

sisteminin kurulması sürücülerin rahatlıkla yüklenmesi açısından kolaylık sağlamaktadır. Bu çalışmadaki işlemler Ubuntu 12.04 LTS(Long Term Support) 32-bit işletim sistemi üzerinde yapılmıştır. Ubuntu bilgisayara yüklenirken dikkat edilmesi gereken bir diğer konu hali hazırda bulunan windowsun yanına wubi aracı yardımıyla yüklenen ubuntunun çok sağlıklı çalışmamasıdır, bundan dolayı harddiskte bir alan oluşturulup Ubuntu'nun o bölüme kurulması tavsiye edilir.

2.10.2. Xilinx ISE

Bu kurulum anlatımı tamamıyla Ubuntu için yazılmış olup başka Linux ortamlarında farklılık gösterebilir. Xilinx sitesinden indirilen ISE design tool tar dosyasından çıkarılır ve ardından çıkarılan bu dizine konsol yardımıyla ulaşılıp "sudo ./xsetup" komutu ile yüklenir. Yükleme sırasında webpack sürümü seçilmesi yeterli olmaktadır. ISE yükleme sırasında hiçbir kablo sürücüsü yüklenmez, daha sonra sürücüler ayrıca yüklenecektir.

ISE yüklendikten sonra teminalde;

```
$ cd /opt/Xilinx/<ISE-sürümü>/ISE_DS
$ sudo gedit startise.sh
```

komutları yardımıyla ise başlatma için gerekli dosya oluşturulur. Açılan yazı düzenleme aracına aşağıdaki satırlar eklenir;

```
#!/bin/bash
. /opt/Xilinx/13.2/ISE_DS/settingsXX.sh
ise
```

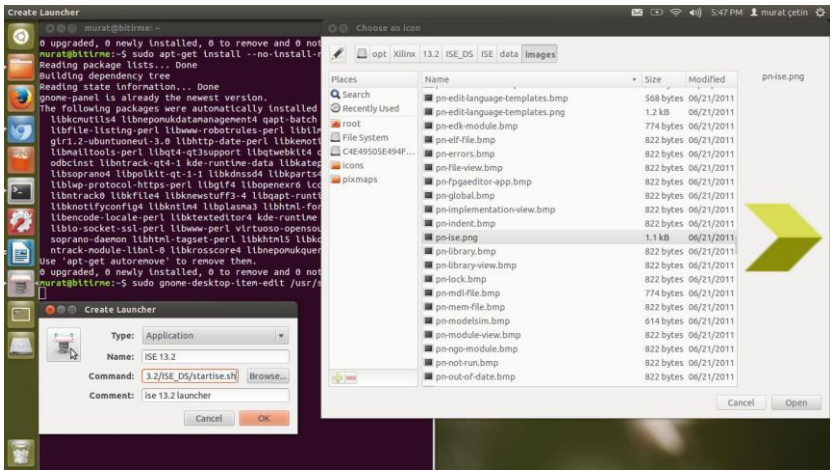
XX yazan yer işletim sisteminin türüne göre değiştirilir (32 bit ise settings32.sh,64 bit ise settings64.sh). Dosya kaydedilip kapatılır, daha sonra aşağıdaki komutlarla dosyanın derlenebilir olması;

```
$ sudo chmod +x startise.sh
```

komutları ile sağlanır ve ardından aşağıdaki komutlar yardımıyla programa daha rahat ulaşmak için bir kısayol oluşturulabilir.

```
$ sudo apt-get install --no-install-recommends gnome-panel
$ sudo gnome-desktop-item-edit /usr/share/applications/ --create-new
```

Yukarıdaki komutlardan sonra çıkan ekran şekil 2.9'da gösterildiği gibi doldurulur. Command alanı için /opt/Xilinx/13.2/ISE_DS dizinindeki oluşturulan startise.sh dosyası seçilir. ISE küçük resmi /opt/Xilinx/13.2/ISE_DS/ISE/data/images/pn-ise.png seçilebilir. Bu işlemlerden sonra ISE Unity içinden çalıştırılabilir veya konsol üzerinden bu işlemleri yapmadan ISE_DS dizininde "./startise.sh" komutuyla çalıtırılabilir.



Şekil 2.9 : Ubuntu İçinde ISE Yüklenme Görüntüsü

2.10.3. Digilent Kablo Sürücüleri

2.10.3.1. Genel USB sürücüleri için libusb

Ubuntu işletim sistemi için aşağıdaki komut ile yüklenebilir:

```
$ sudo apt-get install libusb-dev
```

2.10.3.2. Digilent Adept Runtime and Utilities

Digilent Adept Runtime ve Utilities şekil 2.10'da da görüldüğü gibi Digilent sitesinden, kullanılan işletim sistemine uygun olacak şekilde indirilebilir. İndirilen dosyalar sıkıştırılmış tar.gz formatında olacağından "tar zxf <dosya_ismi>" komutuyla bulunulan dizine çıkartılır.

Şekil 2.10 : Linux için Adept 2 [11]

Çıkarılan digilent.adept.runtime_2.15.3 dizininde "./install.sh" komutu ile varsayılan dizinler değiştirilmeden yüklenir. Eğer "uname –r" komutu ile öğrenilebilen kernel sürümü 3.0'dan büyük ise install.sh dosyasının 209. satırına aşağıdaki komutların eklenmesi gerekmektedir [12].

```
209 cprocUdev=$(ps -e | grep -i -c udevd)
210
211 if [ "${szVmjr}" = "3" ]
212 then
213         if (( $cprocUdev ))
214         then
215                 let fUseUdev=1
216         fi
217 fi
```

Utilities de aynı şekilde ama kernel sürümüne bakılmaksızın yüklenir.

2.10.3.3. Xilinx Design Suite için Digilent Plugin

Digilent Plugin şekil 2.11'de görüldüğü gibi Digilent sitesinden, kullanılan işletim sistemine uygun olacak şekilde indirilebilir. İndirilen dosya sıkıştırılmış tar.gz formatında olacağından "tar zxf <dosya_ismi>" komutuyla bulunulan dizine çıkartılır.

Şekil 2.11 : Digilent Kablo Sürücüsü[13]

libCseDigilent_2.5.2-<platform> dizininde bilgisayarda kurulu olan Xilinx DS sürümünün olduğu dizine girilir. Örneğin bu projede Xilinx DS 13.2 kurulu olduğundan ISE 13x dizinine girilir. Bu dizindeki "Digilent_Plug-in_Xilinx_<versionNo>.pdf" dosyasında tam olarak nasıl yükleneceği anlatılmıştır. Aslında yapılması gereken tek şey libCseDigilent.so ve libCseDigilent.xml dosyalarını;

<Xilinx_DS_Dizini>/ISE/lib/lin64/plugins/Digilent/libCseDigilent/ (64-bit işletim sistemi için)

<Xilinx_DS_Dizini>/ISE/lib/lin/plugins/Digilent/libCseDigilent/ (32-bit işletim sistemi için)

dizinine kopyalamaktır. Eğer <Xilinx_DS_Dizini>/ISE/lib/lin/plugins/ dizini içinde Digilent klasörü yoksa mkdir komutu ile oluşturulması gerekmektedir.

Bu yüklemelerin gerçekleşip gerçekleşmediği ve Adept bileşenlerinin kartı tanıması digilent kartını bilgisayara bağladıktan sonra "djtgcfg enum" komutu ile öğrenilebilir. Bu komuttan sonra bağlantı kurulan kartın ismi terminal ekranında çıktı ve bağlantının kurulduğu anlaşıldı.

2.10.4. Grmon

Grmon'u yüklemek için tar.gz dosyası /opt/ dizinine aşağıdaki komut ile çıkartılır.

```
$ sudo tar zxf grmon-<version>.tar.gz -C /opt
```

Daha sonra "/opt/grmon-<version>/linux/bin" dizinine girilip aşağıdaki komut ile grmon çalışırılır.

```
$ ./grmon -digilent
```

2.10.5. BCC - Bare-C Cross Compiler

LEON3 işlemcisine yüklenecek C/C++ kodlarını derleyebilmek için Gaisler sitesinden BCC indirilip /opt dizinine çıkarıldı.

```
$ sudo tar -C /opt -xjf sparc-elf-<version-number>.tar.bz2
```

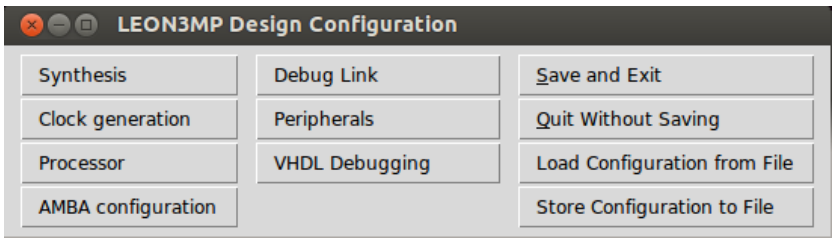
Kurulumdan sonra /opt/sparc-elf-<gcc-version-number>/bin dizini PATH değişkenine eklenmelidir. Bu işlem aşağıdaki satırın ev dizinindeki ".profile" dosyasına eklenmesi ile yapılabilir.

```
export PATH=/opt/sparc-elf-<gcc-version-number>/bin:$PATH
```

3. UYGULAMA

3.1. LEON3 İşlemcisinin Sentezlenmesi

Gaisler internet sitesinden grlib indirilip uygun bir dizine çıkartıldıktan sonra kartlara göre LEON3 kaynak kodlarının bulunduğu designs klasörüne girildi. Bu klasörün içinde kullanılacak olan kartın ismi bulundu, bizim için bu "grlib-gpl-1.3.1-b4135-2/designs/leon3-digilent-atlys/" dizinidir. Kaynak kodlarının derlenmesi için öncelikle işlemcinin özelliklerinin belirlendiği şekil 3.1'deki görünen yapılandırma aracı açılır. Bu işlem "make xconfig" komutu ile rahatlıkla yapılabilir. Eğer terminalde "/bin/sh: 11: wish: not found" hatası gözükürse tcl/tk kurulması gerekmektedir.



Şekil 3.1 : LEON3 Düzenleme Arayüzü

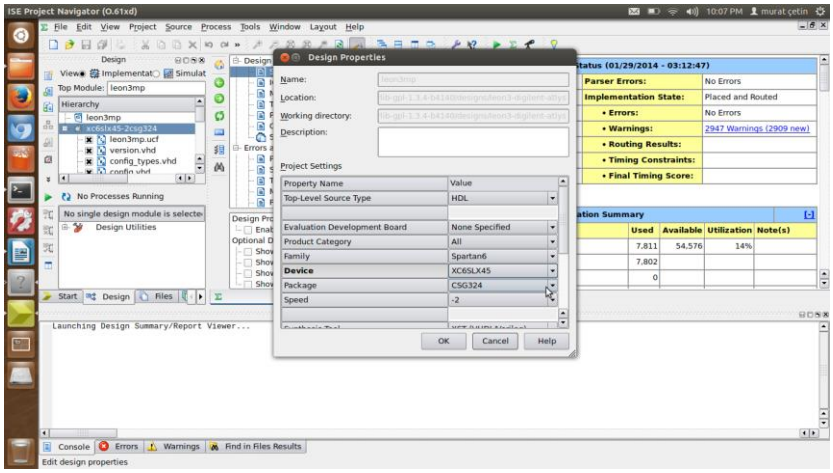
Yapılandırma aracındaki özellikler varsayılan ayarında tutularak "save and exit" yapılarak çıkıldı. Bu işlemin sonucunda üretilen config.vhd dosyası sayesinde tasarım "make vsim" komutu ile derlendi ve sentezlenecek olan VHDL dosyaları üretildi.

Xilinx ISE açılıdı ve "make vsim" komutu ile üretilen proje dosyasının yolu ISE programında proje aç yardımıyla açıldı. Sentezlemek için synthesis üzerine şağ tık yapılarak run seçildi.

3.2. LEON3 İşlemcisinin Translate, Map ve Place&Route Yapılması

Translate işlemi hiçbir ayar gerektirmeden yapıldı. Ancak Map işleminin gerçekleşebilmesi için ISE proje özelliklerinin kullanılan karta göre tekrar düzenlenmesi gerekti. Örneğin şekil 3.2'de de görüldüğü üzere Design Properties kısmındaki Device, Package ve speed parametrelerinin doğruluğu kontrol edilmelidir.

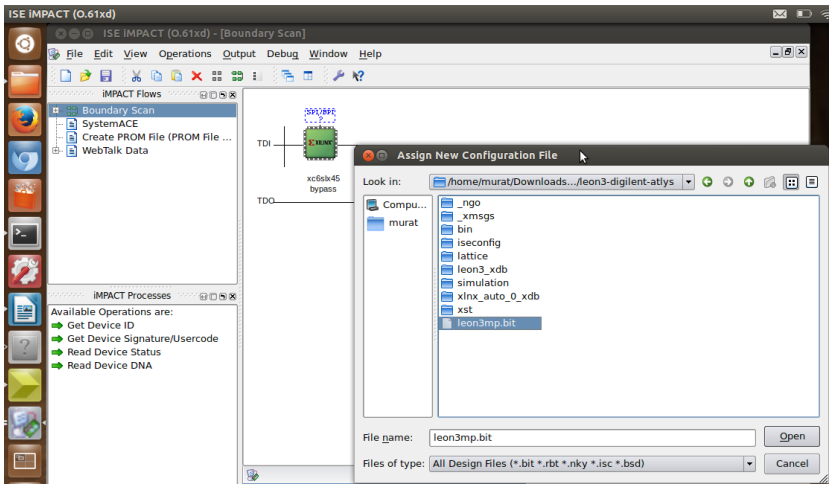
Yani kartın özellikleri ile projenin özelliklerinin karşılaştırılması gerekmekte, karşılaştırma sonucunda farklı olan durum karta göre düzenlendi ve sorunsuz bir şekilde Place&Route işlemi yapıldı.



Şekil 3.2 : Tasarım Özellikleri Ekranı

3.3. Programlama dosyasının üretilmesi ve FPGA'ye gömülmesi

ISE içindeki "Generate Programming File" yardımıyla programlama dosyası oluşturuldu. Yine bu alandaki "Configure Target Device" yardımıyla Impact programlama aracı açıldı. Kart jtag kablosuyla bilgisayara bağlandıktan sonra boundary-scan, ardından initialize chain yapılarak kartın program tarafından otomatik olarak tanınması sağlandı. Kart bulunduktan sonra kutucuğun içine sağ tıklanarak şekil-3.3'deki gibi "Assign New Configuration File" ile az önce üretilen programlama dosyası seçildi.



Şekil 3.3 : ISE IMPACT Ekran Görüntüsü

Sonra yine aynı yere sağ tıklanarak Program işlemi yapılır. LEON3 işlemcisi artık karta gömülmüş oldu ve bunun sonrasında işlemciyle debug(hata ayıklama) programı üzerinden haberleşildi.

3.4. Yazılım Gömülmesi

Bu bölümde yapılacak olan işlemlere başlanmadan önce grlib içinde tasarımı yapılan kartın dizinindeki "README.txt" dosyası iyice okundu. Hata ayıklama programı ile kart jtag kablosuyla bağlı iken "./grmon -digilent" komutu ile karta bağlanıldı. Çıkan konsolda "info sys" komutu ile sistem bilgisi alındı ve bellek biriminin çalışmadığı görüldü. Bellek biriminin yeniden çalışır konuma getirmek için proje dizinindeki bin klasörü içerisindeki program işlemciye yüklendi.

```
grmon2>          load          /home/murat/Downloads/grlib-gpl-1.3.7-
b4144/designs/leon3-digilent-atlys/bin/ddrtune.exe

  A0000000 .text                  896B            [===============>]
100%

  Total size: 896B (7.17Mbit/s)

  Entry point 0xa0000000

  Image    /home/murat/Downloads/grlib-gpl-1.3.7-b4144/designs/leon3-
digilent-atlys/bin/ddrtune.exe loaded
```

```
grmon2> run


DDRTUNE:
...
0000000000000000
0000000000000000
0000000000000001
0011111111111111
1111111111111111
1111111111111111
1111111111111111
1111111111111111
1111111111110000
0000000000000000
0
delay = 0x5f, OK.


  Program exited normally.
```

Bu işlemden sonra grmon hata ayıklama aracından çıkılıp tekrar girildi. Daha sonra bellek birimi düzgün çalışmaya başladı.

3.5. İlk Programın Gömülmesi

İşlemciye deneme amaçlı olarak "hello.c" programı gömüldü. Gaisler firmasının sitesinde BCC örnek kodları arasında da yer alan bu c kodu aşağıda da verilmiştir.

```
#include <stdio.h>
main() {  printf("Hello\n"); }
```

Bu ve bu tarz kodları derlemek için konsolda aşağıdaki BCC derleme komutunun çalıştırılması gerekmektedir.

```
$ sparc-elf-gcc -msoft-float -g -O2 Downloads/examples/hello.c -o
Downloads/examples/hello.exe
```

Kod derlendikten sonra oluşan program dosyası grmon aracılığıyla işlemciye yüklendi ve çalıştırıldı, ancak jtag üzerinden program çıktısı görülemedi. Sadece programın çalıştığı ve normal bir şekilde sonlandığı anlaşıldı:

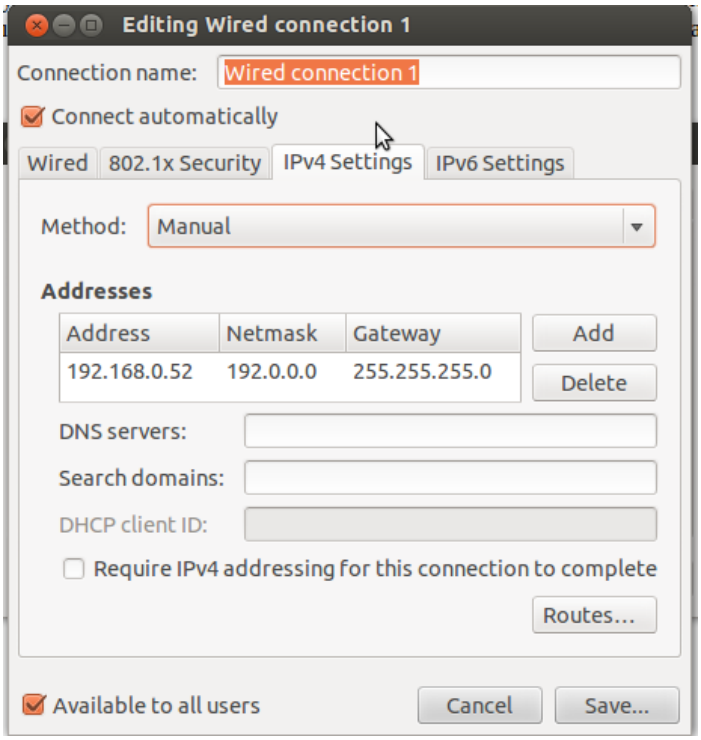
```
grmon2> lo /home/murat/Downloads/examples/hello.exe
  40000000 .text                       23.6kB /  23.6kB   [===============>]
100%
  40005E60 .data                        2.7kB /   2.7kB   [===============>]
100%
  Total size: 26.27kB (469.94kbit/s)
  Entry point 0x40000000
  Image /home/murat/Downloads/examples/hello.exe loaded


grmon2> verify /home/murat/Downloads/examples/hello.exe
  40000000 .text                       23.6kB /  23.6kB   [===============>]
100%
  40005E60 .data                        2.7kB /   2.7kB   [===============>]
100%
  Total size: 26.27kB (467.90kbit/s)
  Entry point 0x40000000
  Image of /home/murat/Downloads/examples/hello.exe verified without errors


grmon2> run
  Program exited normally.
```

3.6. İlk Programın Ethernet Üzerinden Çalıştırılması

Ethernet bağlantısının kurulabilmesi için cihazın ip adresi bilgisayarda statik ip olarak tanımlandı. Cihazın bu Statik IP'si Grlib design dizini içerisinde yer alan README dosyasında da yazdığı üzere 192.168.0.51, bu ayar şekil 3.3'deki gibi yapıldı.



Şekil 3.4 : Bilgisayardaki IP Ayarı

Bu ayar yapıldıktan sonra cihaza bağlanmak için terminal üzerinde root olduktan sonra /opt/grmon-eval-<versiyon>/linux/bin dizininde "./grmon -eth 192.168.0.51 -u -ni" komutu çalıştırıldı ve "hello.exe" dosyası aşağıdaki gibi çalıştırıldı.

```
grmon2> lo /home/murat/Downloads/examples/hello.exe
  40000000 .text                       23.6kB /  23.6kB    [===============>]
100%
  40005E60 .data                        2.7kB /   2.7kB    [===============>]
100%
  Total size: 26.27kB (11.96Mbit/s)
  Entry point 0x40000000
  Image /home/murat/Downloads/examples/hello.exe loaded


grmon2> verify /home/murat/Downloads/examples/hello.exe
  40000000 .text                       23.6kB /  23.6kB    [===============>]
100%
```

```
  40005E60 .data                        2.7kB /   2.7kB    [===============>]
100%
  Total size: 26.27kB (13.45Mbit/s)
  Entry point 0x40000000
  Image of /home/murat/Downloads/examples/hello.exe verified without errors


grmon2> run
Hello



  Program exited normally.
```

4. GELİŞTİRME

4.1. GPIO(General Purpose İnput/Output) Erişimi

GPIO çekirdeği APB(Advanced Peripheral Bus) adres alanındaki tutucular ile kontrol edilmektedir ve bu tutucuların ilk 4'ü veri, ikinci 4'ü çıkış ve üçüncü 4'ü yön tutucusu için ayrılmıştır [3]. Bu tutucular pointer değişkeni ile kontrol edilmesi gerekti. Ayrıca, genel amaç giriş çıkış pinlerine erişebilmek için ram bloğunda hangi bloğun GPIO için ayrıldığına bakıldı. Grmon içerisinde "info sys" komutu ile aşağıdaki satırlar sonucunda GPIO'nun APB veriyolunda "80000a00" adresinden başlandığı görüldü.

```
Gaisler Research  General purpose I/O port (ver 0x2)
             apb: 80000a00 - 80000b00
```

Bu adrese erişebilmek için pointer değişkeni içeren aşağıdaki c kodu yazıldı. Pointer'a GPIO'nun başlangıç adresi atandı.

```
#include <stdio.h>

int main(void)
{
volatile unsigned int *pio = (int *)0x80000A00;

pio[2] = 0x3f; /* ledler çıkış olarak tanımlandı */
pio[1] = 0x3f; /* ledler açık olarak tanımlandı */

printf("Test for LED\n");

printf("Current value of gpio lines: 0x%x\n", *pio);

return 0;
}
```

Kart üzerinde 8 anahtar ve led bulunmasına rağmen son 2'şer tanesi sistem tarafından kullanıldığından ilk 6 tane led yakıldı ve ilk 6 anahtarın durumu gözlendi. Uygulama ilk çalıştırıldığında anahtarların 6'sı da açık durumda iken ikinci durumda sw2 ve sw3 kapatılmıştır. Uygulamanın sonucu aşağıda verildi.

```
grmon2> lo examples/led.exe
  40000000 .text                 52.3kB /  52.3kB   [===============>]
100%
```

```
  4000D160 .data                  2.7kB /   2.7kB   [===============>]
100%
  Total size: 55.04kB (11.87Mbit/s)
  Entry point 0x40000000
  Image /home/murat/examples/led.exe loaded

grmon2> run
Test for LED
Current value of gpio lines: 0x00003f00


  Program exited normally.

grmon2> run
Test for LED
Current value of gpio lines: 0x00003300


  Program exited normally.
```

Uygulama kodları “pio**[**1**]** = 0x33” olarak değiştirilip çalıştırıldıktan sonra cihazdaki ledlerin görüntüsü şekil 4.1’de verilmiştir.

Şekil 4.1 : GPIO Kontrol Programı Sonucu Ledler

5. SONUÇLAR

Çalışmanın başlangıcında işlemcinin gömülmesinin ve uygulamalarının örnekleri incelendi. Örnek çalışmalardan yola çıkarak işlemci yapılandırıldı ve sentezlenerek FPGA içerisine gömülüp çalıştırıldı. İşlemcinin çevre birimlerle olan iletişimi gözlendi.

Öncelikle "make xconfig" komutu çalıştırılırken konsolda hata vermesinin sebebi bu fonksiyona uygun programın yüklü olmamasından olduğu anlaşıldı ve "tcl/tk" yüklendi. Daha sonra ISE sentezlenirken ".ucf" dosyasından alınan hatanın uygun karta göre proje oluşturulmadığından olduğu anlaşıldı ve proje özelliklerinden gerekli değişiklikler yapılınca sentezleme gerektiği gibi gerçekleşti.

İşletim sistemi olarak 64-bit kullanılırken Grmon ve kablo sürücülerinin 32-bit olması uyum sorunları yaşanmasına neden oldu ve 32-bit işletim sistemine geçildi. İşlemciyle ilk haberleşmede ram algılanmama sorunu yaşandı. Yayıncı firma tarafından bu problem için sağlanan programın işlemci içerisinde çalıştırılması ile ram hataları giderildi. Son olarak ise JTAG bağlantısı ile haberleşirken program tarafından gönderilen verilerin terminal üzerinde görünmediği görüldü ve Ethernet haberleşmesi ile geri yönde de bir haberleşme sağlanmış oldu.

İleriki çalışmalarda daha fazla çevre birim ile haberleşilebileceği ve karmaşık uygulamaların çalıştırılabileceği görülmüş oldu.

Kaynaklar

[1] **Fletcher, Bryan H.**, 2005. FPGA Embedded Processors, *FPGA Embedded Processors: Revealing True System Performance*, San Francisco, ETP-367

[2] **GRLIB product Brief,** [Alıntı Tarihi: 25 Nisan 2014], http://www.gaisler.com/doc/Leon3%20Grlib%20folder.pdf

[3] **GRLIB IP Library User's Manual,** Version 1.3.7 - B4144, April 2014 [Alıntı Tarihi: 1 Mayıs 2014], http://www.gaisler.com/products/grlib/grlib.pdf

[4] **GRLIB IP Core User's Manual,** Version 1.3.7 - B4144, April 2014 [Alıntı Tarihi: 7 Mayıs 2014], http://www.gaisler.com/products/grlib/grip.pdf

[5] **Koca, H.,** 2007. "Robot Manipülatör Denetimi", Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

[6] **Chu, Pong P.,** 2008. FPGA Prototyping by VHDL Examples. Wiley-Interscience, New Jersay.

[7] **MicroBlaze Soft Processor Core**, [Alıntı Tarihi: 8 Mayıs 2014], http://www.xilinx.com/tools/microblaze.htm

[8] **Atlys™ Spartan-6 FPGA Development Board,** [Alıntı Tarihi: 8 Mayıs 2014], http://www.digilentinc.com

[9] **GRMON2 User's Manual GRMON2-UM,** Version 2.0.51, April 2014 [Alıntı Tarihi: 8 Mayıs 2014], http://www.gaisler.com/doc/grmon2.pdf

[10] **BCC - Bare-C Cross-Compiler User's Manual,** Version 1.0.43, June 2013 [Alıntı Tarihi: 8 Mayıs 2014], http://www.gaisler.com/doc/bcc.pdf

[11] **Adept 2 for Linux,** [alıntı tarihi: 11 Mayıs 2014] http://www.digilentinc.com/Products/Detail.cfm?Prod=ADEPT2

[12] **Wang S.,** How-to Install Digilent Cable Driver for Xilinx Design Suite on Ubuntu 11.10, [20 Nisan 2014] http://lighttomorrow.wordpress.com/2011/12/18/how-to-install-digilent-cable-driver-for-xilinx-design-suite-on-ubuntu-11-10/

[13] **Digilent Plugin Download**, [alıntı tarihi: 11 Mayıs 2014] http://www.digilentinc.com/Products/Detail.cfm?Prod=DIGILENT-PLUGIN