İÇİNDEKİLER **Sayfa**

SİMGE LİSTESİ

Bit	İkili düzendeki her bir rakam (0 ve 1)
Bayt	8 bitten oluşan ikili sayı kümesi
Hz	Frekans birimi

KISALTMA LİSTESİ

ASCII	American Standard Code for Information Interchange
CLB	Configurable Logic Block
FPGA	Field Programmable Gate Array
FPL	Field Programmable Logic
FPN	Floating Point Number
IEEE	Institute of Electrical and Electronics Engineers
LCD	Liquid Crystal Display
LSB	Least Significant Bit
LUT	Look Up Table
MLP	Multi Layer Perceptron
MSB	Most Significant Bit
PC	Personel Computer
RAM	Random Access Memory
ROM	Read Only Memory
UART	Universal Asynchronous Receiver/Transmitter
UCF	User Constraints File
XOR	Exclusive Or
VHDL	*VHSIC* Hardware Descpription Language
VLSI	Very Large Scale Integration
YSA	Yapay Sinir Ağı

ŞEKİL LİSTESİ **Sayfa**

ÇİZELGE LİSTESİ **Sayfa**

ÖNSÖZ

Yapay sinir ağları, olayların örneklerine bakıp, bu örneklerin dağılımlarını öğrenip hakkında genellemeler yapan, bilgiler toplayan ve daha sonra hiç görmediği örnekler ile karışılışınca öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilen yapılardır.

Bu projede çok katmanlı algılayıcı tipi yapay sinir ağları incelenmiş ve FPGA üzerinde VHDL programlama dili kullanılarak tasarımı gerçekleştirilmiştir. Ağın sınıflandırma performansının donanım üzerinde gözlemlenebilmesi için proje kapsamında kullanılan Spartan-3E Starter kiti üzerinde yer alan seri kanal ve LCD modüller kullanılmıştır. Seri kanal ağa uygulanacak test girişlerini aktarmakta LCD ise ağın sınıflandırma performansının gözlemlenebilmesini sağlamaktadır.

Tez çalışmalarım sırasındaki özverili yardımlarından dolayı danışman hocam Yrd. Doç. Dr. Burcu ERKMEN'e, her türlü yardım ve fikirlerini esirgemeyen hocalarım Araş. Gör. Evren CESUR, Araş. Gör. Nerhun YILDIZ ve arkadaşım Melike ATAY'a teşekkürlerimi sunarım. Ayrıca bu yoğun dönemimde manevi desteklerini esirgemeyen ailem ve ev arkadaşlarıma şükran borçluyum.

Projeme verdikleri maddi desteklerinden dolayı Elektrik Mühendisleri Odası İstanbul Şubesine teşekkür ederim.

Hilal GÜNEREN

Mayıs 2010

ÖZET

Yapay Sinir Ağları(YSA), insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirebilmek amacı ile geliştirilen bilgisayar sistemleridir. Yapay sinir ağları, olayların örneklerine bakmakta, onlardan ilgili olay hakkında genellemeler yapmakta, bilgiler toplamakta ve daha sonra hiç görmediği örnekler ile karışılışınca öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilmektedirler. YSA doğasında paralel bilgi akışına sahiptir. Bu sebeple gerçek başarımlarını ancak paralel çalışan mimariler üzerinde gösterebilirler.

Bitirme tezi kapsamında yapay sinir ağı algoritmalarından biri olan Çok Katmanlı Algılayıcılar(MLP) hakkında bilgi edinilmiş ve paralel matematiksel yapısı donanımsal olarak FPGA üzerinde VHDL programlama dili kullanılarak gerçekleştirilmiş, ağın sınıflandırma performansı gözlenmiştir.

Tasarım aşamasında Xilinx firmasının Spartan-3E adlı kiti ve üzerinde bulunan xc3s500e kodlu FPGA kullanılmıştır. Program Xilinx firmasının ISE 11.1 Project Navigator kullanılarak derlenmiştir. Gerekli simülasyonlar ModelSIM kullanılarak gerçekleştirilmiştir.

Ağ tasarımında aktivasyon fonksiyonu olarak sigmoid fonsiyonu kullanılmıştır. Ağ algoritmasında yer alan matematiksel blokların tasarımında sayı gösterimindeki dinamiklik ve hassas işlem yapma kapasitesi sebebiyle kayan noktalı sayı(floating point number) sistemi tercih edilmiştir. Öncelikle kayan nokta sayı sistemi formatında toplama çarpma ve sigmoid fonksiyonu blokları tasarlanmış ve daha sonra bu bloklardan genel ağ yapısı oluşturulmuştur. Tasarlanan ağ yapısının sınıflandırma performansı simülasyonlarla test edildikten sonra kart test aşamasına geçilmiştir. Performansın kit üzerinde gözlemlenebilmesi için kit üzerinde yer alan seri kanal ve LCD kullanılmıştır. Sınıflandırma test verileri, C# ile hazırlanan ara yüz programı yardımıyla PC üzerinden seri kanal ile ağ girişine uygulanmıştır. Ağın sınıflandırma performansı kit üzerinde yer alan LCD' de gözlemlenmiştir.

Anahtar Kelimeler: Çok katmanlı yapay sinir ağı, kayan noktalı sayılar, toplama, çarpma, sigmoid fonksiyonu, seri kanal, LCD, FPGA, VHDL,

ABSTRACT

Artificial neural network is a designed computer system which aims to realize some abilities such as creating and discovering new ideas without any help through the way of learning which is one of the most important abilities of the brain. Artificial neural network looks at the samples of the events, makes some generalizations about the event which is related to them and then whenever it encounters with new samples, it can give decisions about them by using learned information. Artificial neural network has parallel information flow in its nature. For this reason, it can show its real ability only on the architectures which are working parallelly.

As a thesis, some information are gotten about Multi Layer Perceptron(MLP) which is one of the artificial neural network algorithms, parallel mathematical structure is made on FPGA as hardware by using VHDL as a programming language, the classification performances of the network is observed.

At the stage of design, the kit of Xilinx firm which is named Spartan-3E and a FPGA with code xc3s500e.The program is organized by using ISE 11.1 Project Navigator of the firm Xilinx. The necessary simulations are provided by using ModelSIM.

At the stage of neural design, sigmoid function is used as activation function. At the stage of mathematical blocks in network algorithm, floating point system is preferred because it has dynamic number system and the capacity to make accurate calculations. Firstly the blocks of addition, multiplication, sigmoid function are designed and then general network structure is made by these block. After the classification performance of designed network structure is tested by simulations, the stage of card test is started. In order to be observed the performance on the kit, serial channel on the kit is used. The data of classification test is implemented to the input of network through PC and serial channel with the help of interface program. The classification performance of network is observed on the LCD which takes part on the kit.

Key Words: Multi Layer Artificial Neural Network , floating point numbers, addition, multiplication, sigmoid function, serial channel, LCD, FPGA, VHDL.

1. GİRİŞ

Akıllı sistemlerde kullanılan Yapay Sinir Ağları(YSA), insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri herhangi bir yardım almadan otomatik olarak gerçekleştirmek amacı ile geliştirilen bilgisayar sistemleridir. Yapay sinir ağları, modelleme, kontrol, görüntü işleme ve tıbbi teşhis gibi alanlarda oldukça zor problemleri çözebilecek yeteneğe sahiptir. Biyolojik yapıdan esinlenilmiş YSA doğasında paralel bilgi akışına sahiptir. Bu sebeple, gerçek zamanlı ve çok yüksek hızlı uygulamalar YSA'lar ile gerçeklenmeye çalışılır. Bu çalışmalar gerçek başarımlarını ancak paralel çalışan mimariler üzerinde gösterebilirler. YSA'nın gerçeklenmesi, yapay sinir hücresi (nöron) modelinin donanıma etkin bir şekilde haritalanması işlemidir.

FPGA (Field Programmable Gate Array - Alanda Programlanabilir Kapı Dizileri), programlanabilir mantık blokları ve bu bloklar arasındaki ara bağlantılardan oluşan ve geniş uygulama alanlarına sahip olan sayısal tümleşik devrelerdir. Tekrar düzenlenebilir FPGA programlanması ile özel amaçlı hızlı donanımlar çok geniş uygulamalar için kullanılabilir. FPGA'ler, geleneksel işlemcilerin sahip olmadığı hız, güvenlik ve paralel işlem yapabilme yeteneğine ve ayrıca VLSI teknolojisinin sahip olmadığı tekrar düzenlenebilirlik kabiliyetlerine sahip olmaları sebebiyle gittikçe kullanımı artan yapılar olmuşlardır.

Bu projede YSA algoritmalarından Çok Katmanlı Algılayıcıların(MLP) FPGA üzerinde gerçeklenmesi ve bir veri kümesinin sınıflandırma performansının gözlemlenmesi amaçlanmıştır.

Paralel aritmetik işlemleri gerçeklemedeki yeteneği sayesinde, yapay sinir ağı'nın modellenmesinde yüksek performans sağlayacağı için projenin FPGA ile gerçeklenmesi uygun bulunmuştur. Diğer yandan Bu çalışmada kullanılan YSA algoritmasının matematiksel yapısının gerçeklenmesinde sayı gösterimindeki dinamiklik ve hassas işlem yapma kapasitesi sebebiyle kayan noktalı sayı(floating point number) sistemi kullanılması tercih edilmiştir. Ağ yapısında aktivasyon fonksiyonu olarak sigmoid(tanh(x)) aktivasyon fonksiyonu alınmıştır. Ağın eğitim işlemi donanımın dışında bir bilgisayar yardımıyla yapılmıştır. Tasarlanan ağın performansının test edilmesi için uygulanacak test veri girişleri bir RS232 alıcı blok tasarlanarak PC üzerinden seri kanal ile aktarılmıştır. Sınıflandırma sonuçları uygulama kartında bulunan LCD üzerinde gösterilmiştir. Tasarım kodları VHDL donanım tanımlama dili ile yazılmış, Xilinx firmasının geliştirildiği ISE programı ile sentezlemiştir. Simülasyonlar ModelSIM üzerinde gözlemlenmiştir.

2. YAPAY SİNİR AĞLARINA GENEL BİR BAKIŞ

Bu bölümde yapay sinir ağları hakkında genel bir bilgi verilmeye çalışılmış, yapay sinir ağlarının özelliklerinden ve genel olarak kullanım alanlarından bahsedilmiştir.

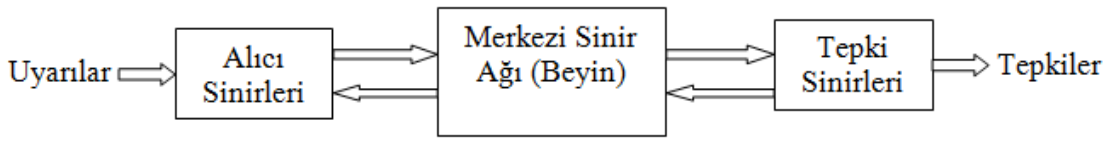
2.1 Genel Tanım

Yapay Sinir Ağları (YSA), insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirebilmek amacı ile geliştirilen bilgisayar sistemleridir. Yapay sinir ağları, olayların örneklerine bakmakta, onlardan ilgili olay hakkında genellemeler yapmakta, bilgiler toplamakta ve daha sonra hiç görmediği örnekler ile karışılışınca öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilmektedir.

Yapay sinir ağları; insan beyninden esinlenerek, öğrenme sürecinin matematiksel olarak modellenmesi uğraşı sonucu ortaya çıkmıştır. Bu nedenledir ki, bu konu üzerindeki çalışmalar ilk olarak beyni oluşturan biyolojik üniteler olan nöronların modellenmesi ve bilgisayar sistemlerinde uygulanması ile başlamış, daha sonraları bilgisayar sistemlerinin gelişimine de paralel olarak birçok alanda kullanılır hale gelmiştir.

Yapay Sinir Ağları, basit biyolojik sinir sisteminin çalışma şeklini simüle etmek için tasarlanan programlardır. Simüle edilen sinir hücreleri (nöronlar) içerirler ve bu nöronlar çeşitli şekillerde birbirlerine bağlanarak ağı oluştururlar. Bu ağlar öğrenme, hafızaya alma ve veriler arasındaki ilişkiyi ortaya çıkarma kapasitesine sahiptirler. Diğer bir ifadeyle, YSA'lar, normalde bir insanın düşünme ve gözlemlemeye yönelik doğal yeteneklerini gerektiren problemlere çözüm üretmektedir. Bir insanın, düşünme ve gözlemleme yeteneklerini gerektiren problemlere yönelik çözümler üretebilmesinin temel sebebi ise insan beyninin ve dolayısıyla insanın sahip olduğu yaşayarak veya deneyerek öğrenme yeteneğidir.

Biyolojik sistemlerde öğrenme, nöronlar arasındaki sinaptik (synaptic) bağlantıların ayarlanması ile olur. Yani, insanlar doğumlarından itibaren bir yaşayarak öğrenme süreci içerisine girerler. Bu süreç içinde beyin sürekli bir gelişme göstermektedir. Yaşayıp tecrübe ettikçe sinaptik bağlantılar ayarlanır ve hatta yeni bağlantılar oluşur. Bu sayede öğrenme gerçekleşir. Bu durum YSA için de geçerlidir. Öğrenme, eğitme yoluyla örnekler kullanarak olur; başka bir deyişle, gerçekleşme girdi/çıktı verilerinin işlenmesiyle, yani eğitme algoritmasının bu verileri kullanarak bağlantı ağırlıklarını (weights of the synapses) bir yakınsama sağlanana kadar, tekrar tekrar ayarlamasıyla olur.[1]



Şekil 2.1 Biyolojik sinir sisteminin blok diyagramı

YSA'lar, ağırlıklandırılmış şekilde birbirlerine bağlanmış birçok işlem elemanlarından (nöronlar) oluşan matematiksel sistemlerdir. Bir işlem elemanı, aslında sık sık transfer fonksiyonu olarak anılan bir denklemdir. Bu işlem elemanı, diğer nöronlardan sinyalleri alır; bunları birleştirir, dönüştürür ve sayısal bir sonuç ortaya çıkartır. Genelde, işlem elemanları kabaca gerçek nöronlara karşılık gelirler ve bir ağ içinde birbirlerine bağlanırlar; bu yapı da sinir ağlarını oluşturmaktadır.

Sinirsel (neural) hesaplamanın merkezinde dağıtılmış, adaptif ve doğrusal olmayan işlem kavramları vardır. YSA'lar, geleneksel işlemcilerden farklı şekilde işlem yapmaktadırlar. Geleneksel işlemcilerde, tek bir merkezi işlem elemanı her hareketi sırasıyla gerçekleştirir. YSA'lar ise her biri büyük bir problemin bir parçası ile ilgilenen, çok sayıda basit işlem elemanlarından oluşmaktadır. En basit şekilde, bir işlem elemanı, bir girdiyi bir ağırlık kümesi ile ağırlıklandırır, doğrusal olmayan bir şekilde dönüşümünü sağlar ve bir çıktı değeri oluşturur. İlk bakışta, işlem elemanlarının çalışma şekli yanıltıcı şekilde basittir. Sinirsel hesaplamanın gücü, toplam işlem yükünü paylaşan işlem elemanlarının birbirleri arasındaki yoğun bağlantı yapısından gelmektedir.

Çoğu YSA'da, benzer karakteristiğe sahip nöronlar tabakalar halinde yapılandırılırlar ve transfer fonksiyonları eş zamanlı olarak çalıştırılırlar. Hemen hemen tüm ağlar, veri alan nöronlara ve çıktı üreten nöronlara sahiptirler.

YSA'nın ana öğesi olan matematiksel fonksiyon, ağın mimarisi tarafından şekillendirilir. Daha açık bir şekilde ifade etmek gerekirse, fonksiyonun temel yapısını ağırlıkların büyüklüğü ve işlem elemanlarının işlem şekli belirler. YSA'ların davranışları, yani girdi veriyi çıktı veriye nasıl ilişkilendirdikleri, ilk olarak nöronların transfer fonksiyonlarından, nasıl birbirlerine bağlandıklarından ve bu bağlantıların ağırlıklarından etkilenir.[2]

2.2 YSA'ların Genel Özellikleri

YSA'lar, uygulanan ağ modeline göre değişik karakteristik özellikler göstermelerine karşın temel birkaç ortak özelliğe sahiptirler.

- YSA'larda sistemin paralelliği, toplamsal işlevin yapısal olarak dağılmışlığıdır. YSA'lar birçok nörondan meydana gelir ve bu nöronlar eş zamanlı olarak çalışarak karmaşık işlevleri yerine getirir. Diğer bir değişle karmaşık işlevler birçok nöronun eş zamanlı çalışması ile meydana getirilir. Süreç içerisinde bu nöronlardan her hangi biri işlevini yitirse dahi sistem güven sınırları içerisinde çalışmasına devam edebilir.
- Genelleme yeteneği, diğer bir değişle ağ yapısının, eğitim esnasında kullanılan nümerik bilgilerden eşleştirmeyi betimleyen kaba özellikleri çıkarsaması ve böylelikle eğitim sırasında kullanılmayan girdiler için de, anlamlı yanıtlar üretebilmesidir.
- Ağ fonksiyonları non-lineer olabilmektedir. Yapı üzerinde dağılmış belli tipteki non-lineer alt birimler özellikle, istenen eşleştirmenin denetim ya da tanımlama işlemlerinde olduğu gibi non-lineer olması durumunda işlevin doğru biçimde yerine getirilebilmesini matematiksel olarak olası kılarlar.
- Sayısal ortamda tasarım, YSA'ların donanımsal gerçekleştirilebilirliklerinin göstergesidir. Bu özellik beklide YSA'ların günlük hayatta daha da fazla yaşamımızın içine gireceğinin (girebileceğinin) göstergesidir.[2]

2.3 Yapay Sinir Ağı Türleri

- Özdüzenleyici Haritalar(Self-Organizing Maps(SOM))
- Çok Katmanlı Algılayıcı (Multi Layer Perceptron - MLP)
- Radial Basis Function(RBF) Network
- İleri Beslemeli Sinir Ağları(Feedforward Neural Networks(FFNN))
- Yinelenen Sinir Ağları(Recurrent Neural Networks(RNN))

Geleneksel Algoritmalar	**Yapay Sinir Ağları**
• Çıkışlar, koyulan kurallara girişlerin uygulanması ile elde edilir.	• Öğrenme esnasında giriş çıkış bilgileri verilerek, kurallar koyulur.
• Hesaplama; merkezi, eş zamanlı ve ardışıldır.	• Hesaplama; toplu, eş zamansız ve öğrenmeden sonra paraleldir.
• Bellek paketlenmiş ve hazır bilgi depolanmıştır.	• Bellek ayrılmış ve ağa yayılmıştır. Dahilidir.
• Hata toleransı yoktur.	• Hata toleransı vardır.
• Nispeten hızlıdır.	• Yavaş ve donanıma bağımlıdır.
• Bilgiler ve algoritmalar kesindir.	• Deneyimden yararlanır.

Tablo 2.1 Geleneksel algoritmalarla yapay sinir ağlarının karşılaştırılması[2]

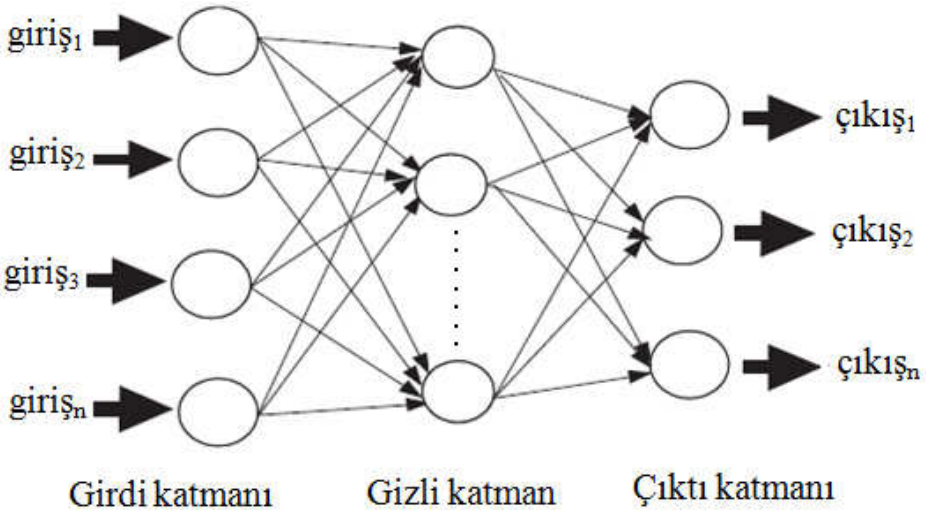
2.4 YSA'ların Kullanıldığı Alanlar

Yapay sinir ağları başlıca; *Sınıflandırma, Modelleme ve Tahmin* uygulamaları olmak üzere, pek çok alanda kullanılmaktadır. Başarılı uygulamalar incelendiğinde, YSA'ların çok boyutlu, gürültülü, karmaşık, kesin olmayan, eksik, kusurlu, hata olasılığı yüksek sensor verilerinin olması ve problemi çözmek için matematiksel modelin ve algoritmaların bulunmadığı, sadece örneklerin var olduğu durumlarda yaygın olarak kullanıldıkları görülmektedir. Bu amaçla geliştirilmiş ağlar genellikle şu fonksiyonları gerçekleştirmektedirler;

- Muhtemel fonksiyon kestirimleri
- Sınıflandırma
- İlişkilendirme veya örüntü eşleştirme
- Zaman serileri analizleri
- Sinyal filtreleme
- Veri sıkıştırma
- Örüntü tanıma
- Doğrusal olmayan sinyal işleme
- Doğrusal olmayan sistem modelleme
- Optimizasyon
- Kontrol [2]

3. ÇOK KATMANLI ALGILAYICI(MLP) YAPISI

Tezde kullanılan YSA, MLP tipi ağ yapısındadır. MLP yapısı birçok birimin oluşturduğu bir kümedir. Bu algılayıcı birimler bir araya gelerek ağdaki katmanları oluşturur. Bu katmanlar da bir araya gelerek ağı oluşturur. MLP'de üç temel katman vardır. Bunlar giriş katmanı, gizli katman ve çıkış katmanıdır. Giriş ve çıkış katmanı dışındaki tüm katmanlar gizli katman olarak adlandırılır.



Şekil 3.1 Çok katmanlı yapay sinir ağı mimarisi

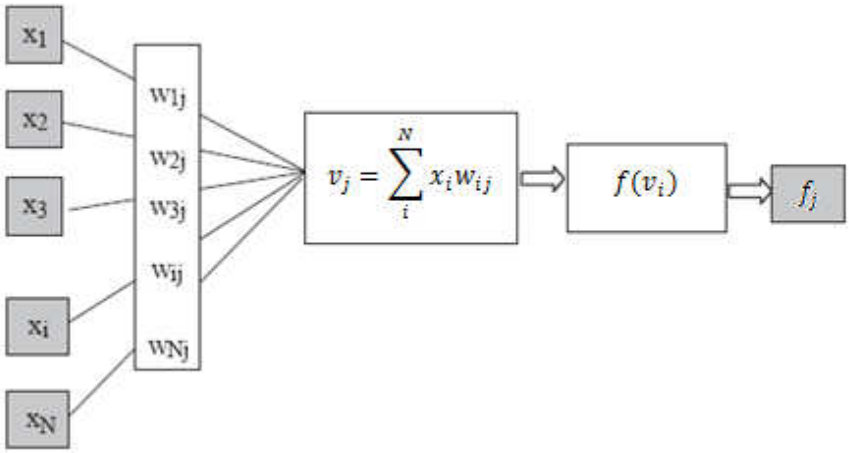
Giriş Katmanı: Dış dünyadan gelen girdileri alarak ara katmana gönderir. Bu katmanda bilgi işleme olmaz. Gelen her bilgi geldiği gibi bir sonraki katmana gider. Her işlem elemanın sadece bir tane girdisi ve bir tane çıktısı vardır. Yani, girdi katmanındaki her işlem elemanı bir sonraki katmanda bulunan işlem elemanlarının hepsine bağlanır.

Gizli Katman: Ara katmanlar girdi katmanından gelen bilgileri işleyerek bir sonraki katmana gönderir. Çok katmanlı bir ağda birden fazla ara katman ve her katmanda birden fazla işlem elemanı bulunabilir.

Çıkış Katmanı: Ara katmandan gelen bilgileri işleyerek ağa girdi katmanından verilen girdilere karşılık ağın ürettiği çıkışları belirleyerek dış dünyaya gönderir.

Bir çıktı katmanında birden fazla işlem elemanı olabilir. Her işlem elemanı bir önceki katmanda bulunan bütün proses elemanlarına bağlıdır. Her işlem elemanının bir çıktısı vardır.[3]

3.1 Ağın Matematiksel Altyapısı



Şekil 3.2 Çok katmanlı yapay sinir ağı nöron yapısı

Bu çalımada $f(v_i)$ aktivasyon Fonksiyonu olarak Sigmoid (tanh) aktivasyon fonksiyonu seçilmiştir.

$$v_j = \sum_{i}^{N} x_i w_{ij}$$

$$u_j = f(v_j)$$

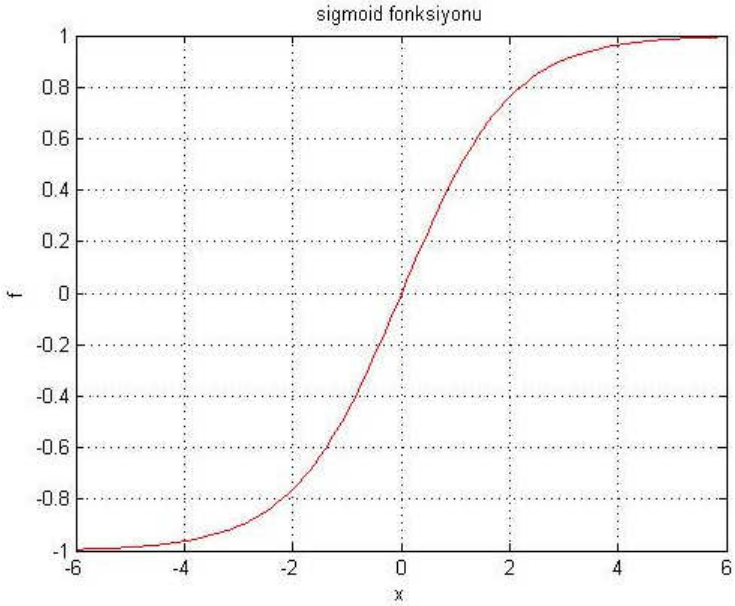
$$f_j = \frac{2}{1 + e^{-au_j}} - 1$$

3.1.1 Sigmoid Aktivasyon Fonksiyonu

Sigmoid (tanh) fonksiyonu sürekli ve doğrusal olmayan bir fonksiyon olması nedeniyle doğrusal olmayan problemlerin çözümünde kullanılan YSA'larda tercih edilir.

Sigmoid fonksiyonu;

$$f = \frac{2}{1 + e^{-ax}} - 1$$



Şekil 3.3 Sigmoid fonksiyonu

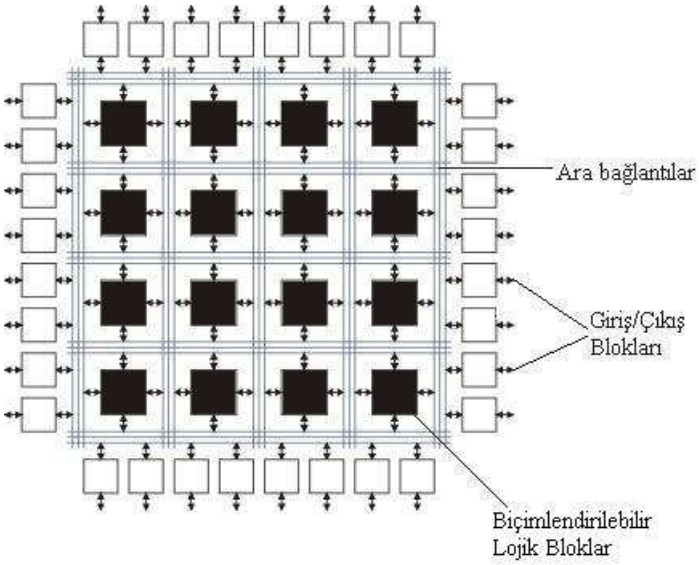
4. FPGA

Bu bölümde FPGA mimarisine, programlanmasına ve FPGA üzerinde devre tasarım süreçlerine değinilmiştir.

4.1 Genel FPGA Mimarisi

FPGA'ler sahada programlanabilen lojik (FPL) cihazlar ailesinin bir üyesidir. FPL'ler birbirini tekrar eden küçük lojik bloklar içeren, programlanabilir cihazlar olarak tanımlanırlar. FPGA, programlanabilir mantık blokları ve bu bloklar arasındaki programlanabilir ara bağlantılardan oluşan ve geniş uygulama alanlarına sahip olan sayısal tümleşik devrelerdir. Tasarımcının ihtiyaç duyduğu mantık fonksiyonlarını gerçekleştirme amacına yönelik olarak üretilmiştir. Dolayısıyla her bir mantık bloğunun fonksiyonu kullanıcı tarafından düzenlenebilmektedir. FPGA ile temel mantık kapılarının ve yapısı daha karmaşık olan devre elemanlarının işlevselliği artırılmaktadır. Alanda programlanabilir ismi verilmesinin nedeni, mantık bloklarının ve ara bağlantıların imalat sürecinden sonra programlanabilmesidir.

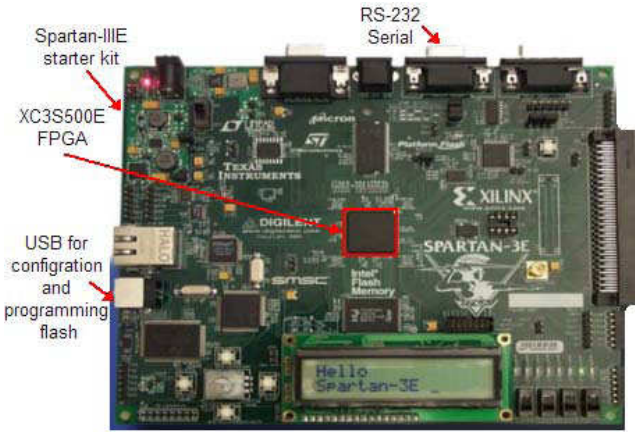
Bir FPGA'i oluşturan temel yapılar Şekil 4.1'de görüldüğü gibi, biçimlendirilebilir lojik bloklar (Configurable Logic Blocks, CLB), giriş/çıkış blokları (Input/Output Blocks, IOB) ve ara bağlantılardır.



Şekil 4.1 FPGA mimarisi

Biçimlendirilebilir lojik bloklar genel olarak doğruluk tabloları (Look-up Table, LUT) ve flip-flop'lardan oluşmuştur. CLB'ler ara bağlantılarla birbirlerine bağlanabilir ve karmaşık fonksiyonlar gerçeklenebilir. IOB'ler FPGA içindeki sinyallerin dış ortamla olan bağlantısını sağlar. Ara bağlantılar ise uygun şekilde programlanarak CLB'ler ve IOB'leri birbirine bağlar. [4]

MLP Tipi YSA yapısının FPGA üzerinde tasarlanmasında Xilinx firmasının Spartan 3E starter kiti kullanılmıştır.



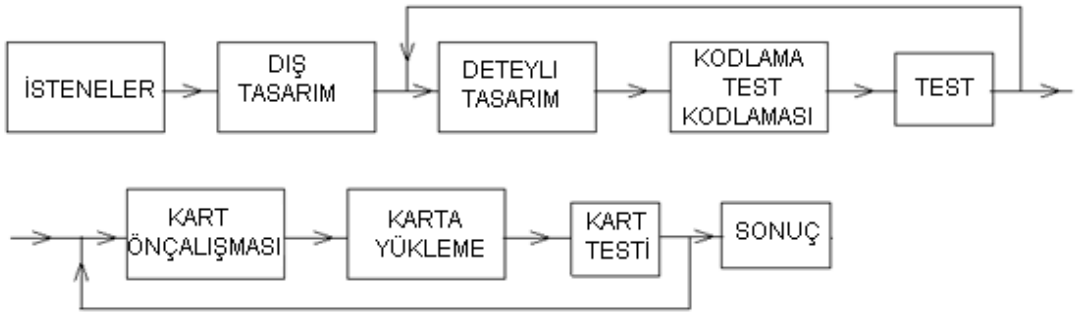
Şekil 4.2 Spartan-3E Starter Kit, XC3S500E FPGA chip

Bu kit XC3S500E-4F modelinde ve üzerinde FG320 kılıfında FPGA, 50MHZ osilatör, İkincil osilatör giriş soketi, 8 switch, 4 push-button, 8 renk destekli VGA portu, RS232 portu, PS2 portu, XCF02S 2Mbit yükleyici PROM, JTAG programlama konnektörü, 3x40 genişleme slotu bulunduran geliştirme kitidir. 5V, 1.6A DA"lık adaptör ile beslenmektedir. 5V sayesinde 3.3V, 2.5V ve 1.2V gerilimler elde edilmektedir. GND (pin1), 5V(pin2) ve 3.3V(pin3) her genişleme slotunun ilk üç pinidir. Böylece dış ek donanımlara da gerilim kit üzerinden verilebilinmektedir. 5V çıkışları direkt adaptörden alınmaktadır.

4.2 FPGA'in Programlanması

FPGA'yi programlamak için gerekli ilk adım donanım tanımlama dilleri ile (Hardware Description Language, HDL) veya şematik yardımıyla, FPGA'in davranışını tanımlamaktır. Daha sonra üretici tarafından sağlanan yazılım ile devreye ait bağlantı listesi (netlist) oluşturulur. Bu aşamada devrenin doğru bir şekilde çalışıp çalışmadığı bilgisayar ortamında simülasyon yapılarak test edildikten sonra varsa gerekli değişiklikler yapılır ve lojik sentezleme aşamasına geçilir. Lojik sentezleme esnasında devre fonksiyonları indirgenerek FPGA içindeki CLB'lerle eşleştirilir ve kapı seviyesinde bir bağlantı listesi oluşturulur. Bir sonraki aşama olan yerleştirme ve yollandırma (place and route) aşamasında da devre fonksiyonları ile eşleştirilmiş CLB'ler, FPGA içerisinde uygun yerlere yerleştirilir ve bu bloklar arasındaki bağlantılar oluşturulur. Gerçeğe çok yakın olan bu modelin benzetim yoluyla çalışması sınandıktan sonra FPGA üreticisi tarafından sağlanan yazılım ile FPGA'i programlamak için gerekli kod oluşturulur. Son olarak uygun donanımlar yardımıyla FPGA bir önceki aşamada oluşturulan kodlarla programlanır ve işlem tamamlanır.

4.3 FPGA Kullanılarak Gerçeklenen Devrelerin Tasarım Süreçleri



Şekil 4.3 FPGA ile devre tasarım süreçleri

İstenenler:	Çalışmada yapılması istenen ifadelerdir.
Dış Tasarım:	Çalışmanın giriş çıkış bilgilerini ve genel olarak proje hakkında bilgi içeren adımdır.
Detaylı Tasarım:	Detaylı tasarımda çalışmanın adım adım ne yapacağını ve işlemlerini göstermelidir. Örneğin sistemin saatin yükselen kenarın da mı düşen kenarın da mı çalışacağına ya da resetin aktif '0' mı ya da aktif '1' mi olacağı kararlarının alındığı, process içinde yapılan işlemler hakkında bilgi veren adımdır. Kısacası çalışma hakkında bilgisi olmayan birinin bile detaylı tasarıma baktığında projenin nasıl çalıştığı hakkında bilgi edinebilmelidir.
Kodlama Ve Test Kodlaması:	Hazırlanan detaylı tasarıma göre VHDL kodunun ve Test Kodlarının (Test Bench) yazılması.
Test:	Kodun simülasyonunun yapılması.

Detaylı tasarımın ne kadar iyi yapılmasına bağlı olarak test sonucunda başa, detaylı tasarıma geri dönme sıklığı değişir.

Testin doğruluğu sonrasında kart ön çalışmasına geçilir.

Kart Ön Çalışması:	Kartta bulunan saat girişinin, kullanıcı tarafından tanımlanan butonların ya da

anahtarların, ledlerin ve benzeri şekilde kullanıcı tarafından kullanılan yapıların hangi pinlerde tanımlandığı öğrenilir ve kod ile (UCF kodu) tanıtılır. Karta yüklenebilecek hale getirilir.

Karta Yükleme: Karta yükleme ISE üzerinden usb bağlantısı ile yapılır.

Kart Testi: Kodun çalışması kartta test edilir.

5.VHDL
(Very high speed integrated circuit Hardware Description Language)
Hiyerarşik düzen:

- VHDL'de sayısal bir sistem uygun alt bloklara ayrıştırılarak tasarlanır.
- Hiyerarşide yer alan her bir alt blok iki farklı boyutta tanımlanır.
  - Diğer bloklarla olan ara yüzü
  - Kendi lojik içyapısı

Alt bloklar:

- Entity/Varlık

VHDL'de her blok kendi başına bir devre olarak düşünülür ve entity olarak adlandırılır.

Entity verilen bir lojik fonksiyon için bütün giriş ve çıkışları tanımlar. Yani lojik fonksiyonun dış dünyayla bağlantısını tanımlar. Her VHDL tasarım mutlaka en az bir entity içerir.

- Architecture/Mimari

Lojik fonksiyonun işlevi architecture tarafından belirlenir. Her entity için bir architecture tanımlanır.

- Process/Usul, Yordam

Tasarlanan Entity'nin davranışını tanımlar.

Donanım tanımlama biçimleri:

- Yapısal Tanımlama (Structural Description)

Sayısal bir modülün iç lojik yapısını oluşturan elementer lojik kapıların ve/veya çeşitli lojik işlevi yerine getiren alt blokların birbirleri ile bağlantılarını dikkate alarak tanımlama.

- Davranışsal Tanımlama (Behavioral Description)

Sistemden istenilen davranışı, lojik yapıdaki karşılığını düşünerek programlama dillerini andıran tarzda tanımlama.

- Veri Akışı Tanımlama(Data Flow Description)

Veri akışı tanımlamada temel blokların (örneğin *and* kapısı) girişlerinin ve çıkışlarının devre içinde nasıl bağlanacağı tanımlanır. İşaretlerin devre içinde nasıl akacağı tanımlanır.

Blok içinde yer alan her bölüm eş zamanlı olarak işlem yapar. Bilgiler sıralı olarak değerlendirilmez. Tasarımda buna dikkat edilmelidir.[5]

5.1 Entity
Verilen bir lojik fonksiyon için bütün giriş ve çıkışları tanımlar. Yani lojik fonksiyonun dış dünyayla bağlantısını tanımlar. Her VHDL tasarım mutlaka en az bir entity içerir.
Port: Giriş ve çıkış işaretidir. Port ifadesi, her işaret için, port tanımlayıcı, port yönü ve port veri tipini belirlemelidir. Port da üç yön kullanılır. Giriş için in, çıkış için out, çift yönlü portlar için inout kullanılır. Buffer ise çıkış için karşılaştırılabilir, sinyal değerleri okunabilir tek sürücülü bir moddadır. Pek çok değişik veri tipi kullanılabilir.

```
entity Ornek is

port (A,B: in std_logic;

      Z: out std_logic_vector(3 downto 0));

end Ornek;
```

• İlk satır yeni tanımlanacak devrenin ismini bildirir: Ornek

• Son satır tanımlamanın bittiğini gösterir.

• Aradaki satırlar devrenin giriş çıkışlarını tanımlar.

• Port tanımında her satır bir giriş-çıkış listesi, modunu ve veri tipini gösterir.

• Son satırın dışındaki her satır " ; " ile bitirilir. Port tanımının tamamı da " ; " ile bitirilir.[5]

5.2 Architecture

Lojik fonksiyonun işlevi architecture tarafından belirlenir. Her entity için bir architecture tanımlanır.

```
entity AND_Gate is

      port(A, B: in bit; X:out bit);

end entity AND_Gate;

Architecture dataflow of AND_Gate is

begin

      X <= A and B;

end dataflow;
```

• İlk satır dataflow isimli mimarinin AND_Gate isimli devreye ait olduğunu gösteriyor.

• begin ve end arasındaki satırlar AND_Gate in nasıl bir işlem gerçekleyeceğini belirtir.

5.3 Process

Process içine yazılan kod sürekli olarak çalışır. Sürekli denetlenmesi istene koşullu ifadeler process içine yazılır. Ya da clock gibi sürekli değişen yapılar process kullanılarak bir kerede yazılır.

```
process
   begin
      clock <= '1'; wait for 5 ns;
      clock <= '0';
      wait for 5 ns;
end process;
```

Process iki formda kullanılır. Sensivity list ile kullanımmı ve wait statement ile kullanımı.

Sensivity List:

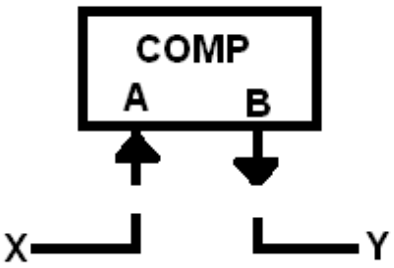
ÖRNEK: process(SEL, A, B) ; process işlemine SEL, A, B sinyallerinin gelmesi ile başlar.

Wait Statement ta üsteki clock örneği gibi kullanılır.

NOT: Eğer process ne sensivity list nede wait statement kullanılmadan tasarlanırsa process sonsuz döngüye girer.

5.4 Component Ve Port Map

Tasarımdaki temel bloklar entity ve karşılık gelen architecture kullanılarak tanımlandıktan sonra diğer tasarımlarda kullanılmak üzere birleştirilebilir.
Bu işlem yapılırken alt bloklar üst bloğun içinde component olarak yer alır. Ardaki bu bağlantı port map kullanılarak tanımlanır.




```
component COMP
port (A: in std_logic;
      B: out std_logic);
end component;
.....
signal X, Y: std_logic
.....
C1: COMP port map(x,y);
```

Şekil 5.1 Component gösterimi ve VHDL kod ifadesi

5.5 Veri Nesneleri

- Sinyaller (Signals) : devredeki lojik sinyalleri tanımlar.
- Sabitler (Constants) : başlangıçta değer atanır ve sonradan değiştirilemezler.
- Değişkenler (Variables) : geçici bilgi saklar. “Process” ve “subprogram”da ifade edilirler.

5.6 Ardışıl Koşul Deyimleri

5.6.1 If deyimi

```
IF <koşul> then
        <durum>;

Elsif <koşul> then

          <durum>

Else

          <durum>;

End if;
```

5.6.2 Case deyimi

```
Case <koşul> is
        When <sabit_değer>=>

                <durum>;

         When <sabit_değer>=>

                  <durum>;

End case;
```

5.6.3 Loop deyimi

[loop_label:]

FOR <değişken_adı> IN range

LOOP

<durum>;

{statement;}

END LOOP [loop_label];

[loop_label:]

WHILE <boolean_koşul>

LOOP

<durum>;

{statement ;}

END LOOP [loop_label];

5.7 IEEE Kütüphanesi

STD_LOGIC_1164

Types: Std_logic ve Std_logic_vector

Operators: and, nand, or, nor, xor, not

Implicit operators: =, /=, >, >=, <, <=

STD_LOGIC_UNSIGNED/SIGNED

Types: Std_logic ve Std_logic_vector

Operators: and, nand, or, nor, xor, not

Implicit operators: =, /=, >, >=, <, <=,+ ,-

(vektörlerde toplama çıkarma işlemi yapılmasına izin verir.)

NUMERIC_STD

Types: Integer, Signed, Unsigned

Operators: and, nand, or, nor, xor, not

Implicit operators: =, /=, >, >=, <, <=, +, -

(Tipler arasında dönüşüme izin verir.)

5.8 STD Kütüphanesi

STD.STANDART

Types: Integer, Boolean, Bit, Time, String...

5.9 Dönüşüm Fonksiyonları

Tipler arası dönüşüm yapılırken her iki tipin de yer aldığı kütüphane ve paket koda eklenmelidir. Kullanılan paketlerin dönüşüm ifadeleri bilinmelidir.

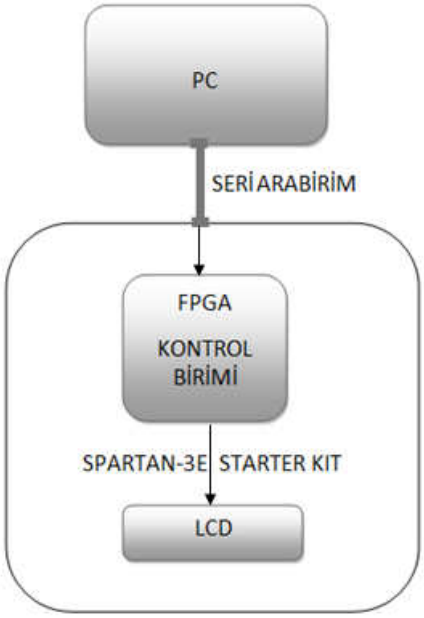
```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
use IEEE.NUMERIC_STD.all;
use IEEE.STD_LOGIC_UNSIGNED.all;
- - - - - - - -
signal U :UNSIGNED(7 downto 0);
signal I : INTEGER;
.....
I <= TO_INTEGER(U);- - - -numeric_std kullanılır.
U <= TO_UNSIGNED(I,8); - - - - numeric_std kullanılır.
- - - - - - - -
signal V,W: STD_LOGIC_VECTOR(7 downto 0);
signal I,J: SIGNED(7 downto 0);
.....
I <= TO_INTEGER(UNSIGNED(V));- - - - numeric_std kullanılır.
J <= CONV_INTEGER(W); - - - - std_logic_unsigned kullanılır.
WS<= STD_LOGIC_VECTOR(TO_UNSIGNED(I, 8)); - - - - numeric_std kullanılır.
```

(I, 8) terimindeki ilk parametre işaret edilen sinyali ikincisi ise işaret edilen sinyalden dönecek olan bit sayısını belirtir.[4]

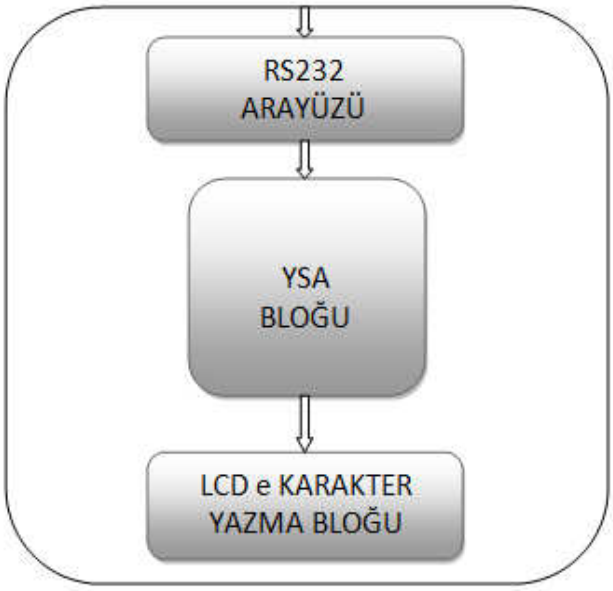
6.FPGA İLE YSA TABANLI AKILLI SİSTEM TASARIMI

6.1 Sistemin Genel Yapısı

Bu bölümde tasarlanan sistemin genel yapısı ve alt blokları ayrıntılı şekilde açıklanmıştır.



Şekil 6.1 Sistemin genel yapısı



Şekil 6.2 Sistemin Kontrol Birimi - FPGA

Proje kapsamında FPGA üzerinde temel olarak üç blok tasarlanmıştır. Bunlardan ilki MLP tipi YSA Bloğu, ikincisi RS232 Seri Haberleşme Bloğu ve üçüncüsü LCD' e Karakter Yazma Bloğu'dur. Proje kapsamında asıl tasarlanmak istenen blok YSA bloğudur.

YSA, insan beyninin özelliklerinden olan öğrenme yolu ile ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirebilen bilgisayar sistemleridir. Yapay sinir ağları, olayların örneklerine bakmakta, onlardan ilgili olay hakkında genellemeler yapmakta, bilgiler toplamaktadır. Bu aşama YSA eğitimi olarak adlandırılır. Bir sonraki aşama test aşamasıdır. Burada YSA'ya daha önce görmediği örnekler uygulanır ondan eğitim aşamasında öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilmesi istenir. Proje kapsamında tasarlanan

YSA eğitimi bilgisayar ortamında gerçeklenmekte ve elde edilen değerler doğrultusunda donanımsal tasarımına geçilmektedir.

Bir veri kümesinin sınıflandırılması için tasarlanan ağ yapısının çalışma performansı değerlendirilmek için uygulanacak test girişleri, hazırlanan ara yüz programı ile PC üzerinden FPGA üzerinde tasarlanmış olan RS232 alıcı bloğuna seri kanal yardımıyla aktarılacaktır. Girişler YSA üzerinde değerlendirilip sınıflandırıldıktan sonra girilen verinin hangi gruba ait olduğu kit üzerinde bulunan karakter LCD üzerinde gözlemlenecektir.

6.2 Tasarım Alt Blokları

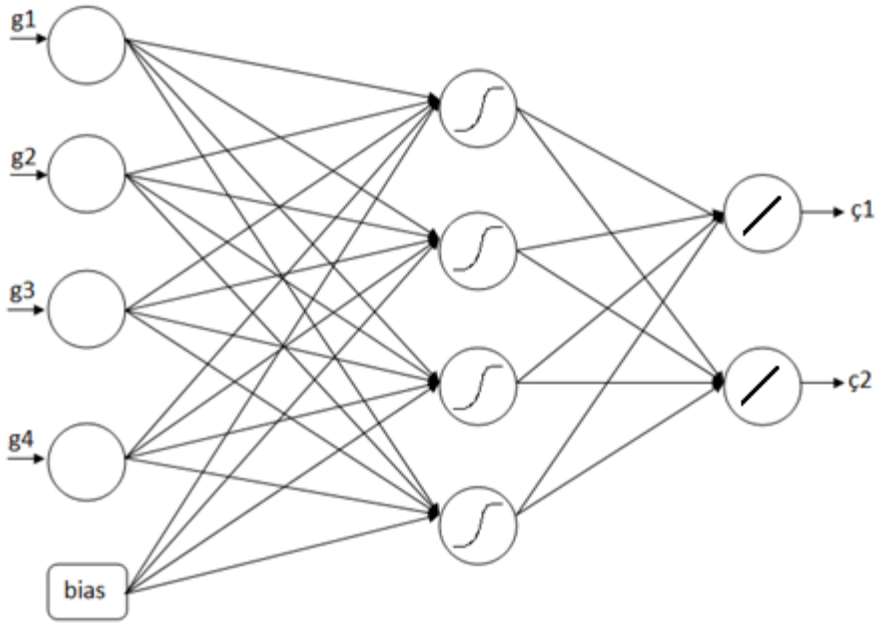
Bu bölümde sistemin genel yapısında bulunan bloklara yer verilmiş ve tasarım süreçleri ayrıntılı bir şekilde anlatılmaya çalışılmıştır.

6.2.1 FPGA ile MLP Tipi YSA Algoritmasının Tasarımı

Bu bölümde çalışmada kullanılan MLP tipi YSA yapısına ve bu yapı altında bulunan matematiksel alt blokların donanımsal tasarımına yer verilmiştir.

6.2.1.1 Tasarlanan Ağ Yapısı

Çalışmamızda kullanılan Ağ(4-4-2), Şekil 6.3 de görüldüğü gibi girdi katmanı bir gizli katman ve çıkış katmanından oluşur. Dört girişe iki çıkışa sahiptir. Aktivasyon fonksiyonu olarak sigmoid fonksiyonu seçilmiştir. YSA eğitimi ve test işleminde kullanılan ağırlık değerleri tez danışmanı Yrd. Doç. Dr. Burcu Erkmen tarafından elde edilmiştir. Bu ağırlık değerleri test işleminde kullanılmak üzere LUT'a aktarılmıştır. Elde edilen ağırlık değerleri Tablo 6.1 ve Tablo 6.2 de verilmiştir.



Şekil 6.3 Üç katmanlı MLP tipi yapay sinir ağı yapısı

w_{11}^{1}	w_{11}^{2}	w_{11}^{3}	w_{11}^{4}	w_{11}^{5}
-0.3202	0.1459	0.2716	-0.1058	0.7233
w_{12}^{1}	w_{12}^{2}	w_{12}^{3}	w_{12}^{4}	w_{12}^{5}
-0.3066	0.4239	-0.3336	-0.6798	-0.0645
w_{13}^{1}	w_{13}^{2}	w_{13}^{3}	w_{13}^{4}	w_{13}^{5}
-0.2718	0.3020	-1.4308	-1.1455	1.0106
w_{14}^{1}	w_{14}^{2}	w_{14}^{3}	w_{14}^{4}	w_{14}^{5}
0.3433	0.1186	0.2969	0.2843	-0.4491

Tablo 6.1 Tasarımda kullanılan girdi katmanına bağlı ağırlıklar

w_{21}^{1}	w_{21}^{2}	w_{21}^{3}	w_{21}^{4}
0.6552	0.1178	-1.0076	-0.7843
w_{22}^{1}	w_{22}^{2}	w_{22}^{3}	w_{22}^{4}
1.0031	-0.9509	0.5138	0.3994

Tablo 6.2 Tasarımda kullanılan çıkış katmanına bağlı ağırlıklar

Not: Ağırlıkların genel ifadesi w_{kj}^{i} şeklindedir. Burada "i" hangi girişi çarpan olarak aldığını, "k" bulunduğu katman sayısını, "j" ise hangi nörona ait olduğunu göstermektedir.

Örnek olarak;

w_{12}^{3} : Birinci katmanının ikinci girişi ile çarpılıp üçüncü sigmoid için giriş oluşturur.

Uygulamada kullanılan "bias" değeri 0.9 alınmıştır.

Ağ yapısında çıkışlar bir bitlik verilerdir. Matematiksel işlemler sonucunda nöron çıkışında elde edilen f_j değerinin mutlak büyüklüğü 0,4'den küçük ise çıkış "0", 0,6'dan büyük ise çıkış "1" değerini alır.

6.2.1.2 32 Bit Kayan Noktalı Sayıların İfade Edilmesi

Kayan noktalı sayılarının gösterimi için IEEE 754 standardı kullanılmıştır. IEEE 754 standardına göre 32 bit kayan noktalı sayıların bit düzeyindeki veri yapısında, bit dizisindeki bitlerin bir kısmı üs, bir kısmı çarpan olarak kullanılırken bir tanede işaret bit kullanılmıştır. Sekil-6.4'de gösterilen 32 bit kayan noktalı sayı haritasına göre en anlamlı bit işaret biti olarak kullanılırken, 8 biti üs ve 23 biti de çarpan olarak ayrılmıştır. Sayı genel olarak su şekilde gösterilir.(Ç: çarpan, T: taban, Ü: üs)

$\pm Ç * T^{\pm Ü}$

T, taban bilgisi olup tüm sayılar için aynı değer alacağından dolayı bellekte saklanmasına gerek yoktur. Bu nedenle Ç,Ü ve işaret bitinin saklanması yeterlidir. T, Ç ve Ü'nün gösterimleri şöyledir.

Taban, T=2

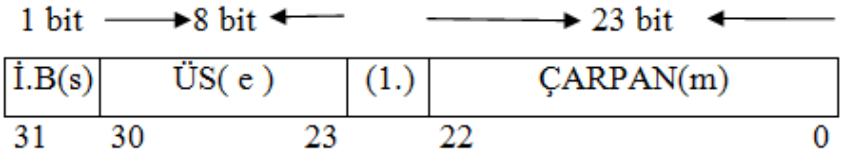
Üs, Ü → 2'ye tümleyen seklinde tutulur.

Çarpan, Ç → Doğal ikili kodda tutulur.

Bu kabuller değiştiğinde kayan noktalı sayının yorumlanması da değişir. Ayrıca 32 bit IEEE 754 standardına göre bazı Ü ve Ç değerlerine Tablo-6.3'de görüldüğü gibi özel anlam yüklenmiştir.

Parametre	32 Bitlik Kayan Noktalı Sayı
İşaret Biti	0, artı ; 1,eksi
Çarpan Uzunluğu	23
Üs Uzunluğu	8
Üssün Tabanı	2
Üssün en büyük değeri	127
Üssün en küçük değeri	-126
10 Tabanına Göre En Küçük Değer	10^{-38}
10 Tabanına Göre En Büyük Değer	10^{38}
Kesir Sayısı/ Genişliği	2^{23}= 8 Mega
Sıfır Sayısı İçin	Ü=0, Ç=0
Sonsuz Sayısı İçin	Ü=225, Ç=0

Tablo 6.3 32 Bitlik IEEE 754 Kayan Noktalı Sayı Özellikleri



Şekil 6.4 32 bit kayan noktalı sayının bit haritası(IEEE 754)

İki tabanı için genel gösterimi aşağıdaki gibidir.

Sayı=$(-1)^{s}$*(1+f) * $2^{\text{üs-bias}}$

i: İşaret bitini temsil etmektedir. İşaret biti sıfırsa sayının pozitif, bire eşitse negatif olduğunu belirtir.

bias: IEEE 754 standartlarında verilen ve üs sayısından çıkarılan değerdir. 32 bit için bu değer 127 olarak belirtilmiştir.
f: Burada çarpan olarak verilmektedir. Sayının kesir kısmıdır. Bu sayı daima sıfır ile bir arasında olmalıdır.
üs-bias: Bu ifade verilen sayının 2'nin *üs-bias* kuvvetine eşit veya büyük, *üs-bias+1* kuvvetinden küçük olduğu değeri gösterir.

Daha kolay bir ifade ile kayan noktalı sayı dönüşümünü ifade edecek olursak;
Ondalık sayı ikili koda çevrilir. Daha sonra virgülden önce 1 kalacak şekilde duruma göre nokta sağa ve ya sola kaydırılır. Bununla beraber üs elde edilmiş olur. Elde edilen üsse 127 ile eklenerek ÜS(e) elde edilir. Ve bu sayı 8 bitle ifade edilir. Elde edilen virgüllü sayıdan virgülden sonraki kısım(kesirli kısım) ÇARPAN(f) 'ı ifade eder. Burada da sayının sonuna "0" eklenerek 23 bitlik çarpan kısmının tamamı doldurulur. İşaret biti ise sayı negatif bir sayı ise "1" sayı pozitif bir sayı ise "0" dır.
Örnek olarak 45.781 sayısını ele alalım.

$45.781_{10}=101101.11001_2$ sayısı normalize kayan nokta gösterimi;

Önce sayı virgülden önce 1 kalacak şekilde kesirli hale getirilir. (En büyük ağırlıklı biti dışında kesirli sayı haline getirilir.)

$101101.11001=1.0110111001*2^5$

İşaret biti = 0(pozitif)

Üst(e)=5 + 127 =$132_{10}=10000100_2$

Kesir (f)=0110111001…00(MSB =1 gösterilmez, "hidden bit")

0	10000100	(1.)	01101110010000000000000

Bunun sonucunda IEEE normalize floating point sayı;

$01000010001101110010000000000000_2$ elde edilir.

6.2.1.3 32 Bit Kayan Noktalı Sayıların Çarpımı

Çarpma işlemini ondalık bir örnekle açıklayacak olursak

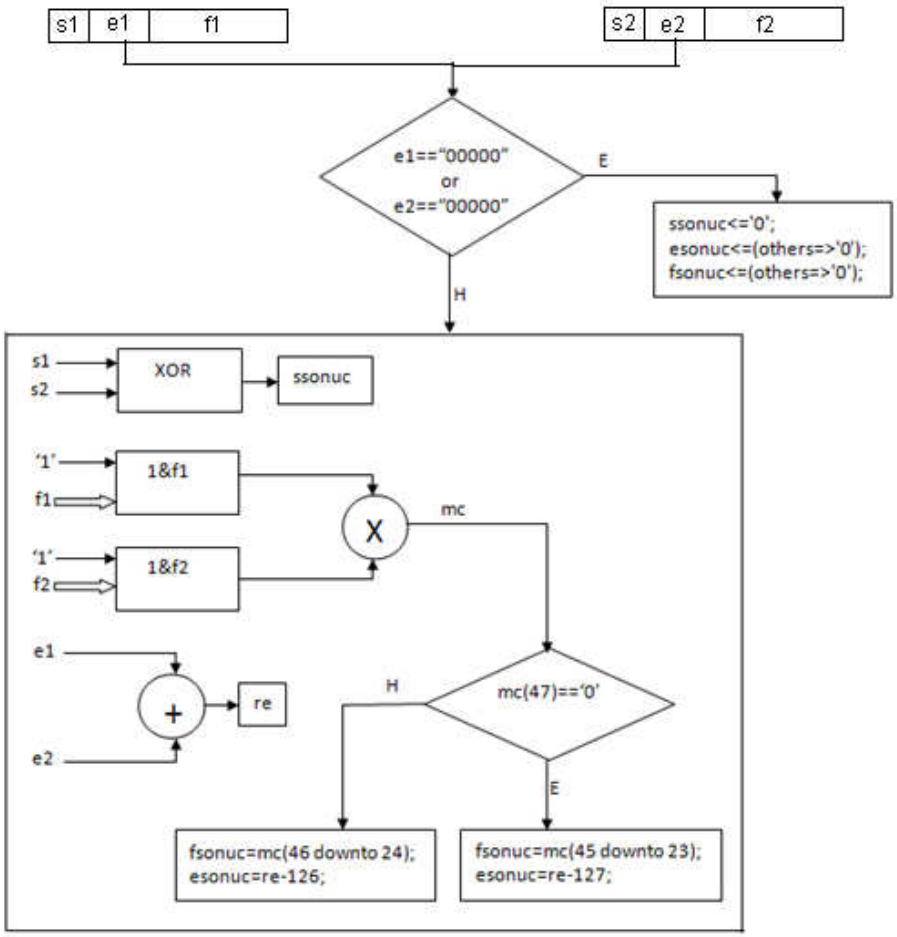
$$3.25 \times 10^{3}$$
$$\underline{x \quad -2.63 \times 10^{-1}}$$

İlk adım: Üsler toplanır.

İkinci adım: Katsayılar çarpılır.

Üçüncü adım: Sayıların işaretleri birbirine eşitse sonuç değerin işareti pozitif değilse negatiftir.

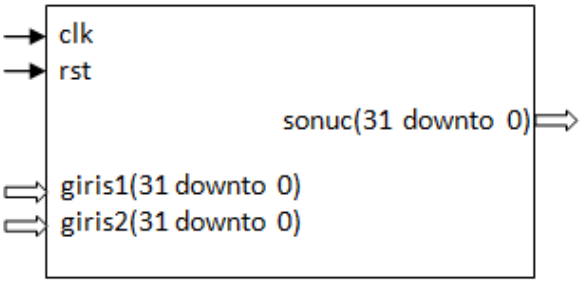
Dördüncü adım: Sonucu normalize etmek.



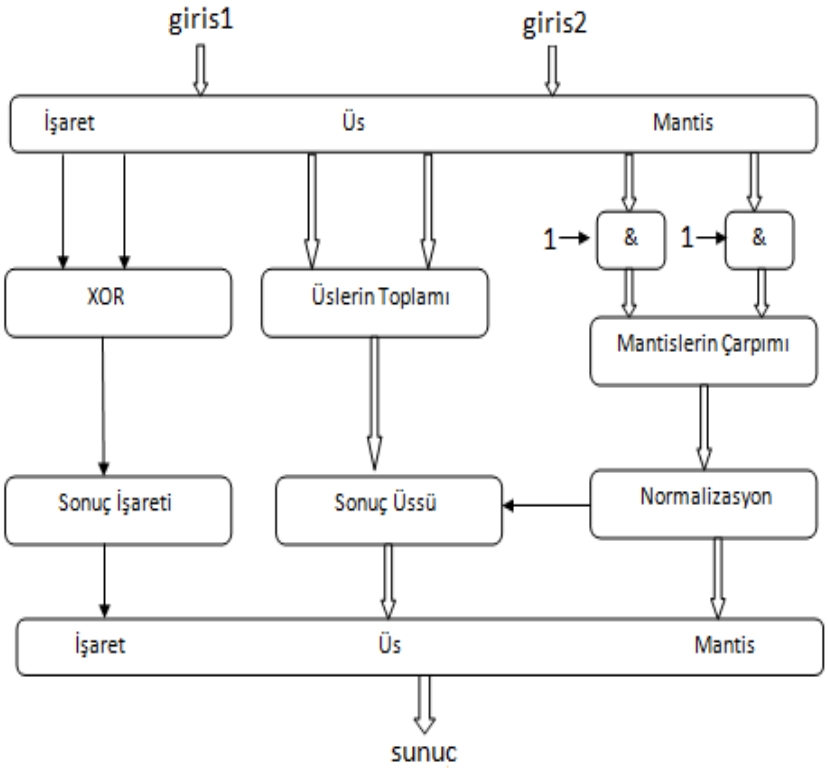
Şekil 6.5 Kayan noktalı sayılarda çarpma işlemi algoritması.

Bu algoritmada 8 bitlik iki sayıda lojik toplama işlemi gerçekleştirilirken eğer taşma meydana geliyorsa en anlamlı bit atılır. Bunun anlamı 8 bitlik iki sayının toplamı 8 bit olmalıdır. Çıkarma işleminde ise 126'nın 2'ye tümleyeni 130 olduğu için toplam ifade 131 olarak bulunur.

Şekil 6.6'da yer alan algoritmayı kullanarak ağ yapısında kullanılacak olan Çarpma bloğunun genel yapısı şekil 6.7'de, blok diyagramı ise şekil 6.8'de yer almaktadır.



Şekil 6.6 Çarpma devresi genel yapısı



Şekil 6.7 Kayan noktalı sayılarda çarpma işlemi blok diyagramı

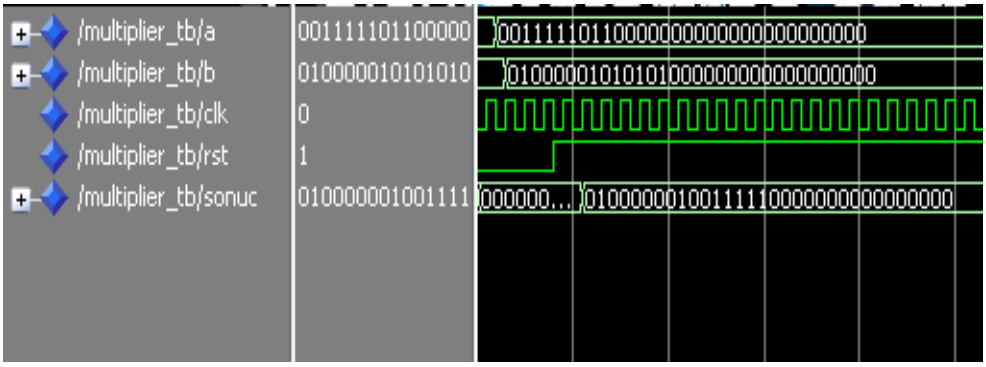
Tasarlanan blokta “clk”: 50 Mhz saat işareti girişi, “rst”: Çarpma işlemi sonucunu istenildiği zaman sıfırlayan girişi, “a”: çarpanlardan ilkinin değerini, “b”: çarpanlardan ikincisinin değerini, “sonuc” ise çarpma işlemi sonucu elde edilen değeri ifade eder.

Lojik Birimler	**Kullanılan Sayı**	**Mevcut Sayı**	**Yüzde**
Dilim Sayısı	84	4656	%1
Flip-Flop Dilim Sayısı	68	9312	%0
Dört Girişli LUT Sayısı	121	9312	%1
IOB Sayısı	99	232	%42
MULT18X18 Sayısı	4	20	%20
GCLK Sayısı	1	24	%4

Tablo 6.4 Çarpma bloğu FPGA kullanım oranları

Çarpma işlemi bloğu simülasyon sonuçları:

Test işleminde fonksiyonuna giris1 ile 0.375(00111110110000000000000000000000), giris2 ile 13.25(01000001010101000000000000000000) değerleri girilmiştir.
sonuc = 4.96875(01000000100111110000000000000000)olarak elde edilmiştir.
Doğru sonuç = 4.96875(01000000100111110000000000000000)’dir.
Hata= ((Doğru sonuç-Sonuç)/Sonuç)*100
Hata = %0 olarak bulunur.



Şekil 6.8 Tasarlanan çarpma bloğunun simülasyon çıktıları

6.2.1.4 32 Bit Kayan Noktalı Sayıların Toplamı

Toplama işlemini ondalık örnekle açıklayacak olursak

$$\begin{array}{r} 3.25 \times 10^{3} \\ +\ 2.63 \times 10^{-1} \\ \hline \end{array}$$

İlk adım: toplama işlemli yapabilmek için üsler eşitlenir. Bu işlemde genelde yüksek üsse eşitleme olur.

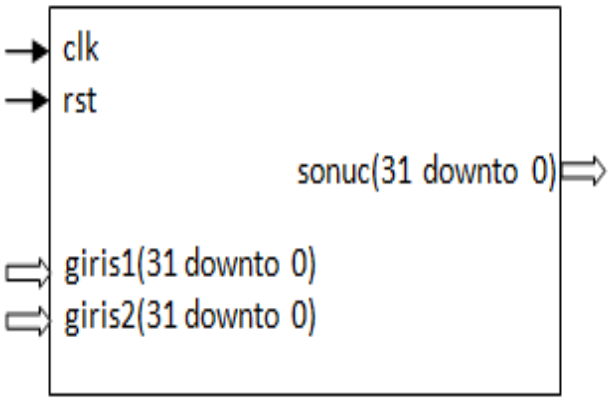
$$\begin{array}{r} 3.250000 \times 10^{3} \\ +\ 0.000263 \times 10^{3} \\ \hline \end{array}$$

İkinci adım: toplama.

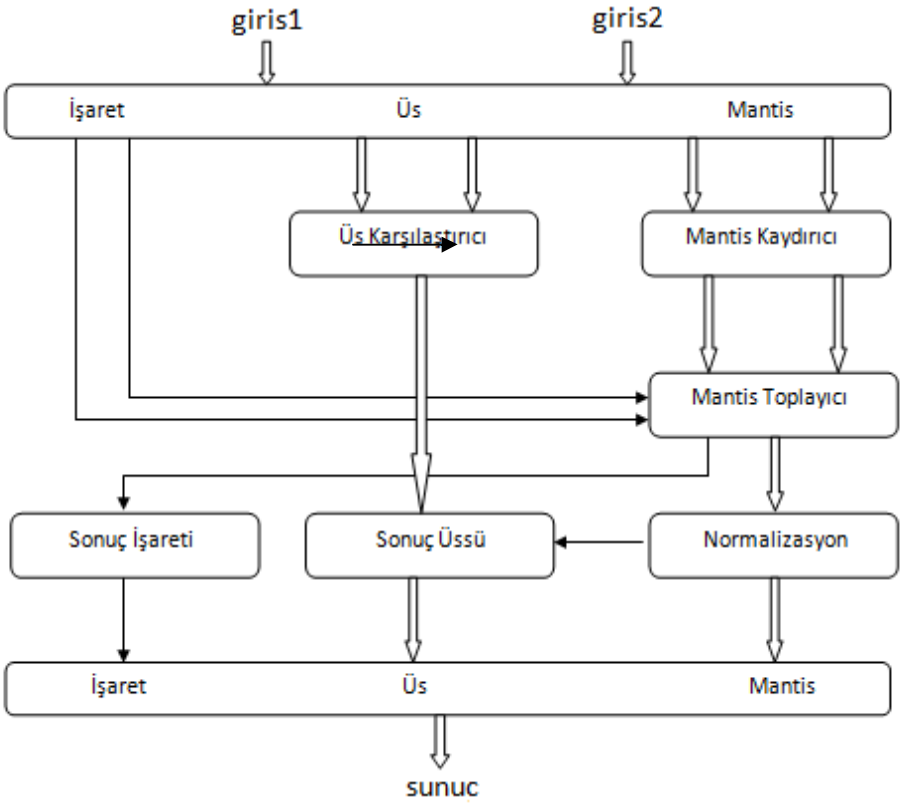
$$\begin{array}{r} 3.250000 \times 10^{3} \\ +\ 0.000263 \times 10^{3} \\ \hline 3.250263 \times 10^{3} \end{array}$$

Üçüncü adım: Sonucu normalize etmek.

Şekil 6.10'da Toplama bloğunun genel yapısı, Şekil 6.11'de tasarlanan devrenin blok diyagramı yer almaktadır.



Şekil 6.9 Toplama devresi genel yapısı



Şekil 6.10 Kayan noktalı sayılarda toplamı işlemi blok diyagramı

Tasarlanan blokta “clk”: 50 Mhz saat işareti girişi, “rst”: Toplama işlemi sonucunu istenildiği zaman sıfırlayan giriş, “a”: toplama işlemi için ilk değeri, “b”: toplama işlemi için ikinci değeri, “sonuc” ise toplama işlemi sonucu elde edilen değeri ifade eder.

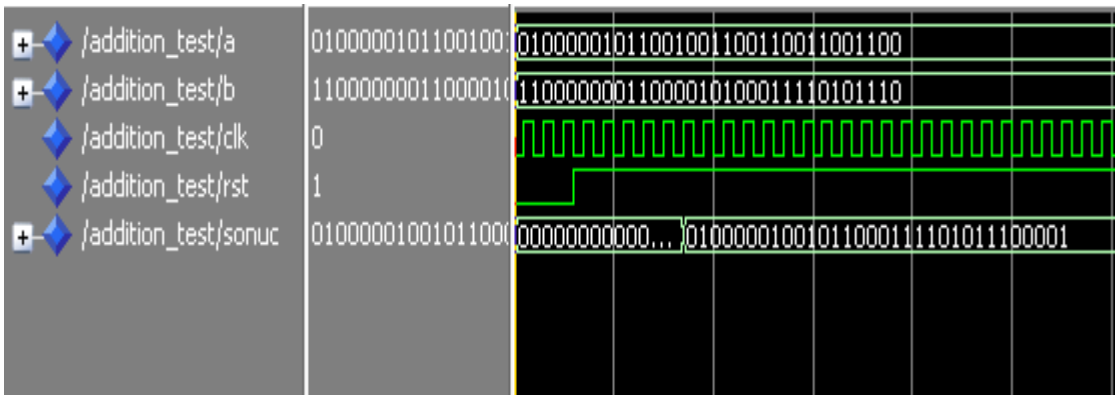
Lojik Birimler	**Kullanılan Sayı**	**Mevcut Sayı**	**Yüzde**
Dilim Sayısı	338	4656	%7
Flip-Flop Dilim Sayısı	243	9312	%2
Dört Girişli LUT Sayısı	645	9312	%6
IOB Sayısı	99	232	%42
MULT18X18 Sayısı	0	20	%0
GCLK Sayısı	1	24	%4

Tablo 6.5 Toplama bloğu FPGA kullanım oranları

Toplama işlemi bloğu simülasyon sonuçları:

Test işleminde fonksiyonuna giris1 ile 14.3(01000001011001001100110011001100), giris2 ile -3.52(11000000011000010100011110101110) değerleri girilmiştir.

sonuc = 10.75(01000000100100101100011110101110000 1)olarak elde edilmiştir.
Doğru sonuç = 10.75(01000000100101100011110101110000 1)’dir.
Hata= ((Doğru sonuç-Sonuç)/Sonuç)*100
Hata = %0 olarak bulunur.



Şekil 6.11 Toplama bloğunun simülasyon çıktıları

6.2.1.5 Kayan Noktalı Sayılarda Sigmoid Fonksiyonu Hesabı

Sigmoid aktivasyon fonksiyonu(1) giriş ve çıkış arasında yumuşak bir transfer işlemi sağlar, böylece keskin aktivasyon fonksiyonlarına nazaran (örneğin; doyumlu doğrusal aktivasyon fonksiyonu) nöron cevapları geliştirilebilir.

$$f = \frac{2}{1 + e^{-2u}} - 1$$

(1)

Sigmoid fonksiyonu bloğunu tasarlarken çeşitli yöntemler düşünülmüş ve denenmiştir. Bunlardan ilki Taylor serisi açılımı(2) ile exponansiyel işleminin ifade edilmesidir.

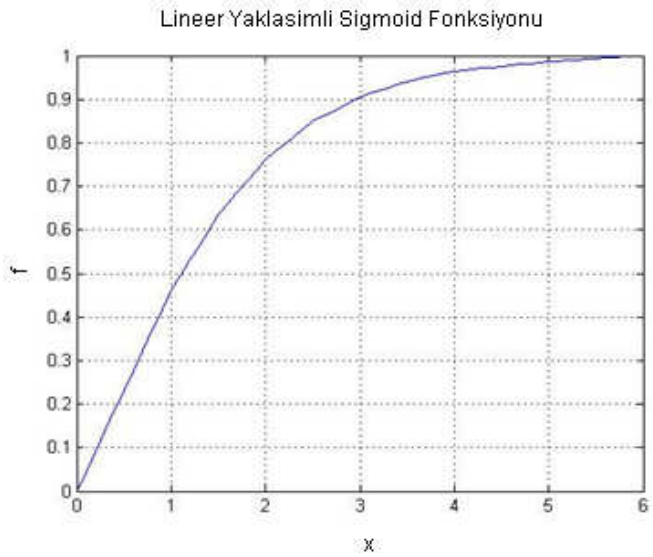
$$e^x = 1 + \frac{1}{1!}x + \frac{1}{2!}x^2 + \frac{1}{3!}x + \frac{1}{4!}x^4 \ldots$$

(2)

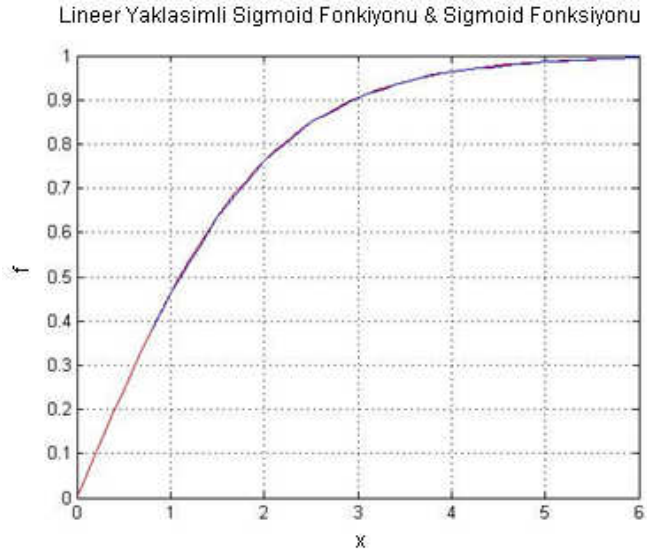
Taylor serisi açılımıyla elde edilen sigmoid fonksiyonu çok fazla çarpma ve toplama işlemi gerektirmekte ve FPGA üzerinde büyük bir alan kaplamaktadır. Ayrıca exponansiyel işlemi bloğu tasarımı dışında 2 tane toplama ve bir de bölme işlemi gerektirmektedir. Bu sebepten kullanımından vazgeçilmiştir.

6.2.1.5.1 Parçalı Lineer Yaklaşımı ile Sigmod Fonksiyonu Hesabı

Tasarımda düşünülen ikinci yöntem parçalı lineer yaklaşımı ile fonksiyonun gerçeklenmesi olmuştur. Bu yapıda sigmoid fonksiyonun pozitif kısmının 14 parçaya bölünerek her parçanın başlangıç değerlerine bağlı olarak lineer denklemler bulunmuş ve bulunan denklemlerin eğimleri başlangıç noktaları ve başlangıç noktalarının değerleri bakma tablosuna kaydedilmiş ve işlemler sırasında kullanılmıştır. Yapının genel tanımını açıklayacak olursak; Girilen değerin 14 parçaya bölünmüş yapıda hangi parçaya ait olunduğunun bulunur. Girilen değer o parçanın başlangıç noktasının değerinden çıkartılır ve parçanın eğimi ile çarpılır. Daha sonra çarpım sonucu elde edilen değer tekrar parçanın başlangıç noktasının değeri ile toplanır ve sonuç elde edilir. En son olarak elde edilen giriş değerinin işareti bulunan sonuç değerinin işaretine atanır.



Şekil 6.12 Parçalı lineer yaklaşıklıkla elde edilen fonksiyon grafiği



Şekil 6.13 Sigmoid fonksiyonu ile lineer yaklaşıklıkla elde edilen fonksiyon

Lojik Birimler	Kullanılan Sayı	Mevcut Sayı	Yüzde
Dilim Sayısı	866	4656	%18
Flip-Flop Dilim Sayısı	643	9312	%7
Dört Girişli LUT Sayısı	1597	9312	%17
IOB Sayısı	66	232	%28
MULT18X18 Sayısı	4	20	%20
GCLK Sayısı	1	24	%4

Tablo 6.6 Parçalı lineer yaklaşımlı sigmoid fonksiyonu bloğu FPGA kullanım oranları

Sigmoid Fonksiyonunun Test Sonuçları:

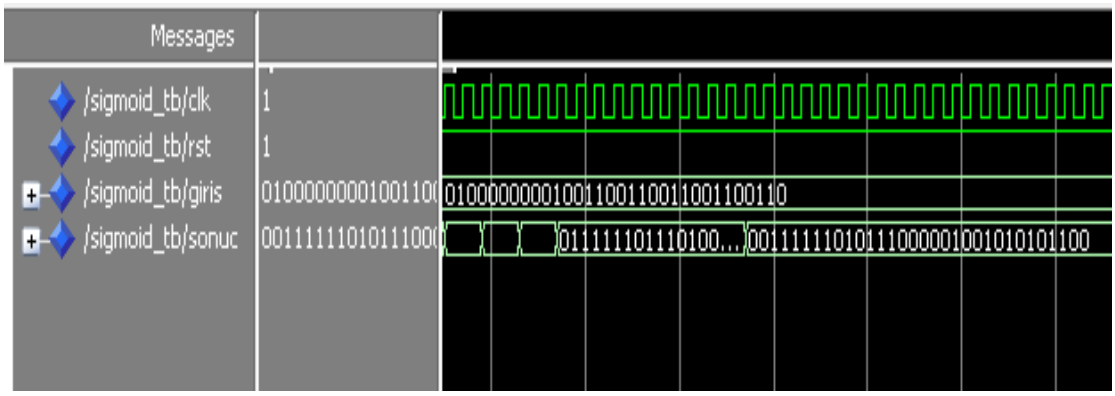
Test işleminde fonksiyona giriş olarak 2.7 değeri girilmiştir.

Sonuç=0.859655022 olarak elde edilmiştir.

Doğru sonuç= 0.8741 dir.

Hata= ((Doğru sonuç-Sonuç)/Sonuç)*100

Hata ~= %1.7 olarak bulunur.

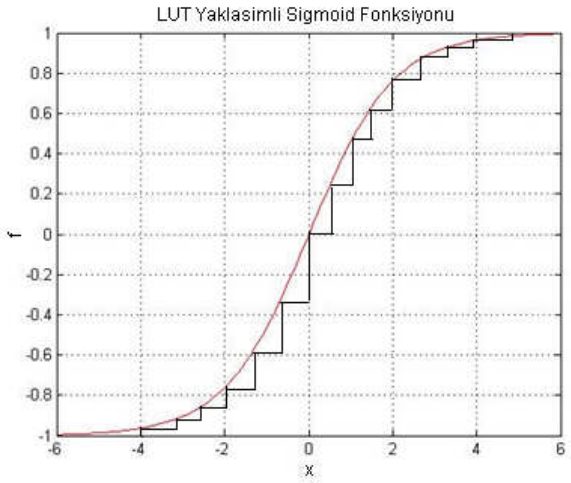


Şekil 6.14 Parçalı lineer yaklaşımlı sigmoid fonksiyonu simülasyon çıktıları

Tablo 6.6 da görüldüğü gibi tasarlanan parçalı lineer sigmoid fonksiyonun kapladığı alan istenilen seviyenin çok üstündedir ve YSA tasarımında FPGA alanının kapasitesini aşmakla sonuçlanmaktadır. Bu sebepten sigmoid fonksiyonu hesaplamada başka bir yöntem arayışına girilmiş ve Bakma Tablosu(LUT-Look Up Table) kullanılmasına karar verilmiştir.

6.2.1.5.2 LUT Yaklaşımı ile Sigmoid Fonksiyonu Hesabı

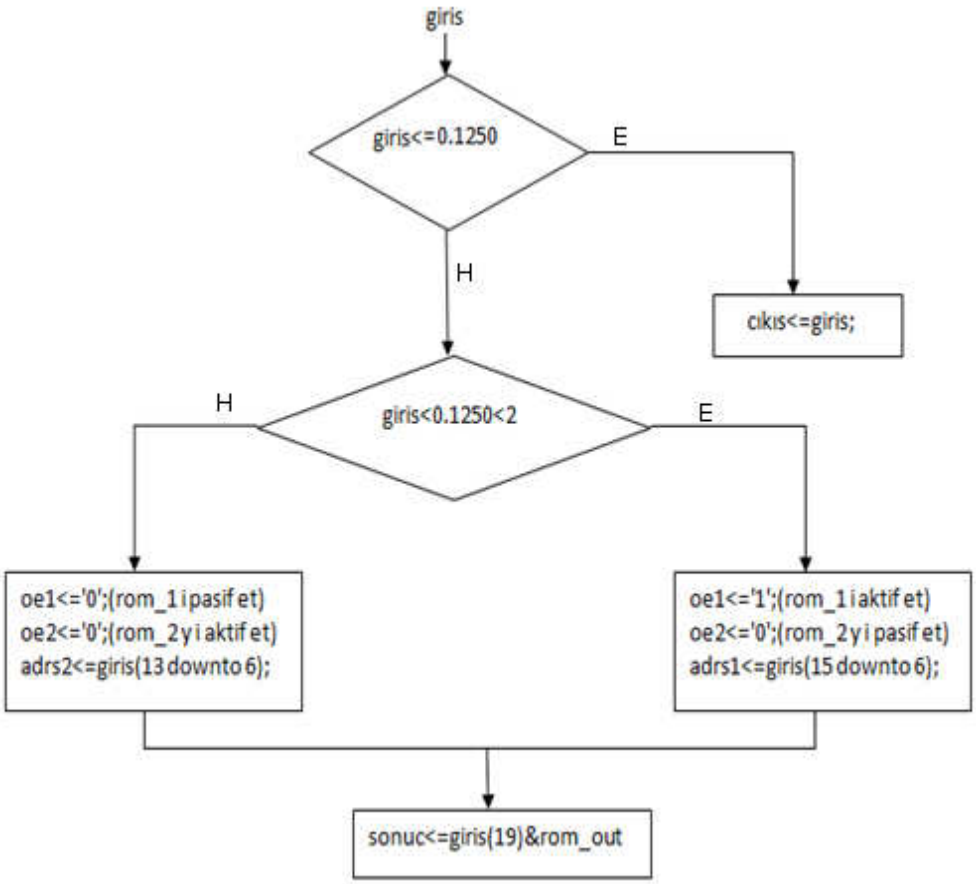
Bakma Tablosu metodunda sigmoid fonksiyonunu belli aralıklarla örneklenir ve örnekleme değerleri hazırlanan bir ROM a kaydedilir. sigmoid fonksiyonun girişi LUT için ROM'dan okunacak verinin adresi olarak kullanılır. İki örnekleme noktasındaki girişler için çıkışa aynı değer atanır.



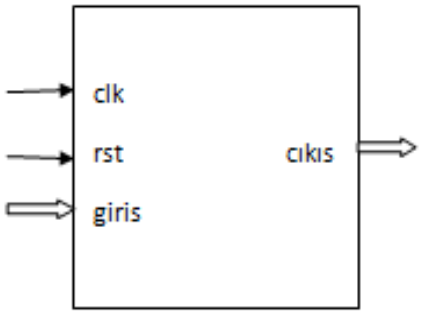
Şekil 6.15 LUT Yaklaşımlı sigmoid fonksiyonu

LUT oluşturulmadan önce 32 bit kayan noktalı sayılarla oluşturulan toplama ve çarpma bloklarında da optimizasyona gidilip bu bloklar; bir bitlik işaret, 5 bitlik üs ve 14 bitlik mantis ten oluşan 20 bitlik kayan noktalı sayı formatında tekrar tasarlanmıştır. Buna bağlı olarak sigmoid fonksiyonu için oluşturulan LUT da bu format kullanılmıştır.

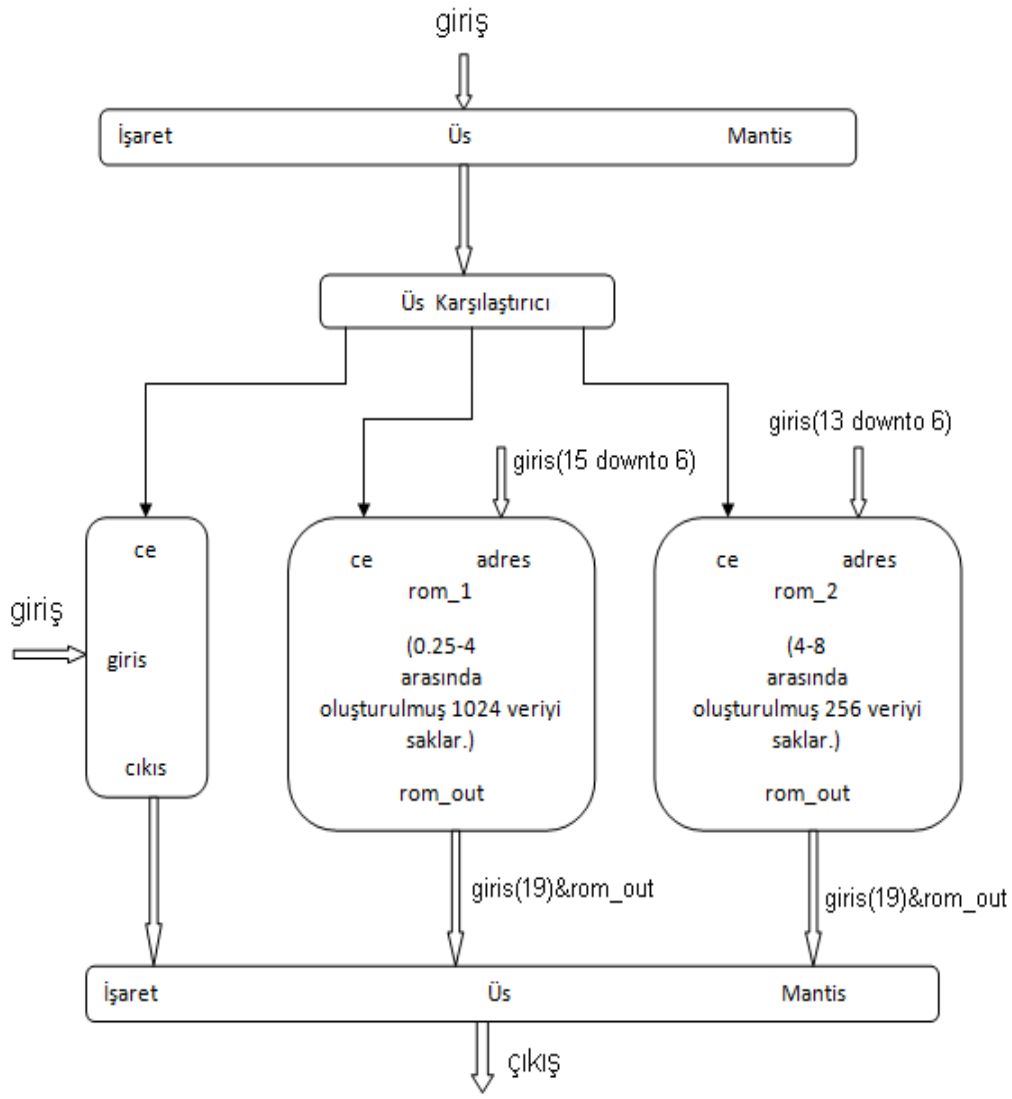
Sigmoid fonksiyonu orijine göre simetrik bir fonksiyondur. Bu sebepten LUT, giriş işaretine bakılmadan oluşturulmuştur. Yani veriler işaret biti hariç 19 bitlik verilerdir. Buna göre 0,250-8 arasında 1280 tane veri hazırlanıp sigmoid bloğunun alt blokları olan rom_1 ve rom_2 blok ROM ları içine gömülmüştür. Hazırlanan bakma tabloları ek olarak verilen CD içerisinde yer almaktadır. Tasarlanan sigmoid fonksiyonun blok diyagramı Şekil 6.19 de verilmiştir.



Şekil 6.16 LUT Yaklaşımlı Sigmoid Fonksiyonu akış şeması



Şekil 6.17 LUT Yaklaşımlı Sigmoid bloğu genel yapısı



Şekil 6.18 LUT Yaklaşımlı Sigmoid Fonksiyonu blok diyagramı

Tasarlanan blokta “clk”: 50 Mhz saat işareti girişi, “rst”: Sigmoid işlemi sonucunu istenildiği zaman sıfırlayan giriş, “giris”: sigmoid fonksiyonu için 20 bitlik giriş değerini, “sonuc” ise sigmoid fonksiyonu sonucu elde edilen 20 bitlik değeri ifade eder.

Tablo 6.7’de bakma tablosu yaklaşımlı sigmoid fonksiyonu ile elde edilen YSA’nın FPGA yararlanma oranları verilmiştir.

Lojik Birimler	**Kullanılan Sayı**	**Mevcut Sayı**	**Yüzde**
Dilim Sayısı	36	4656	%0
Flip-Flop Dilim Sayısı	41	9312	%0
Dört Girişli LUT Sayısı	55	9312	%0
IOB Sayısı	42	232	%18
BRAM Sayısı	3	20	%15
GCLK Sayısı	1	24	%4

Tablo 6.7 LUT Yaklaşımlı Sigmoid Fonksiyonu FPGA kullanım oranları

LUT Yaklaşımlı Sigmoid Fonksiyonunun Test Sonuçları:

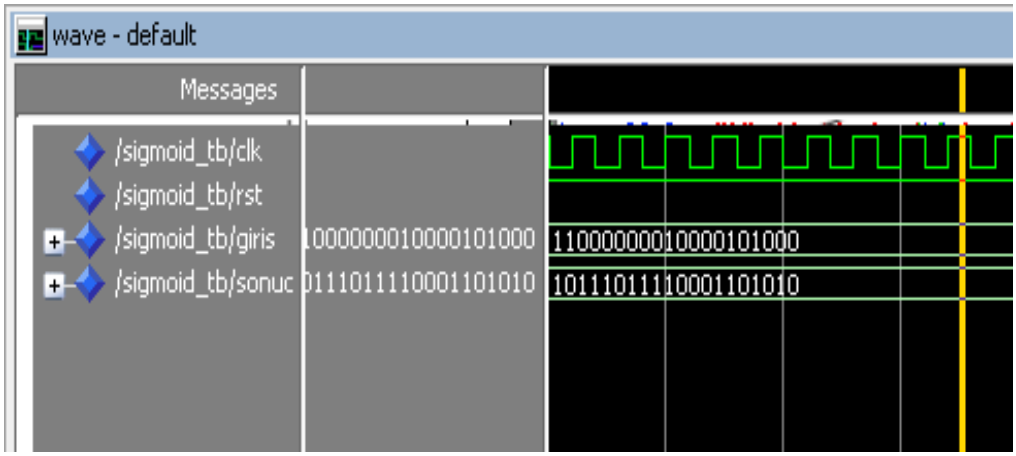
Test işleminde fonksiyona giriş olarak -2.13 değeri girilmiştir.

Sonuç=-0.9720 olarak elde edilmiştir.

Doğru sonuç= -0.9721’dir.

Hata= ((Doğru sonuç-Sonuç)/Sonuç)*100

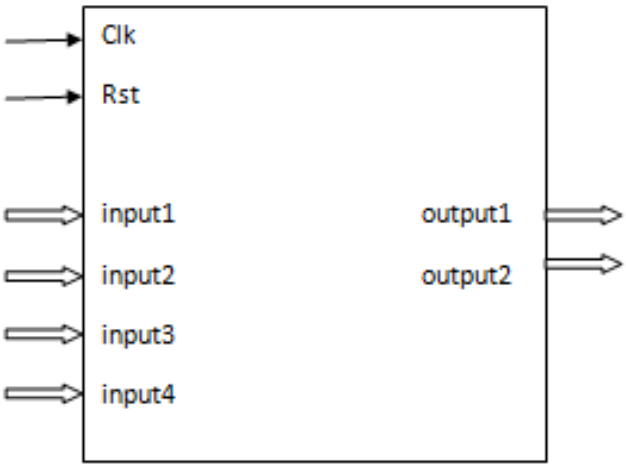
Hata ~= %0 olarak bulunur.



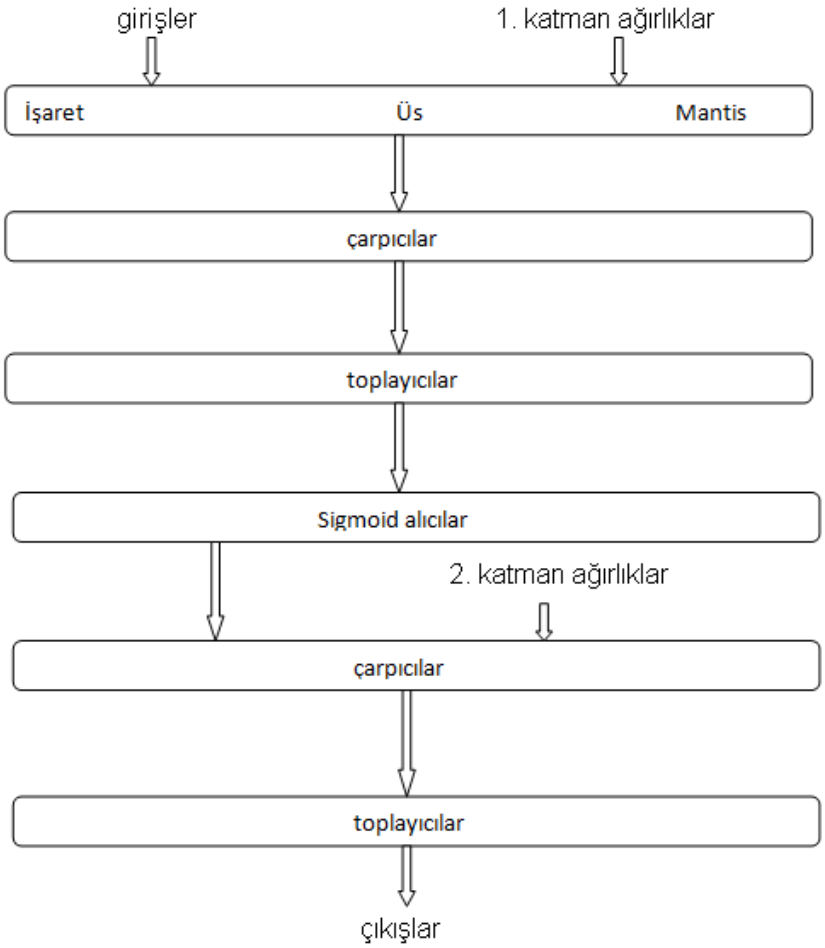
Şekil 6.19 LUT Yaklaşımlı Sigmoid Fonksiyonu Bloğu Simülasyon Çıktıları

6.2.1.6 Yapay Sinir Ağı Donanımının Tasarımı

20 bit kayan noktalı sayı formatında tasarlanan sigmoid, toplama ve çarpma blokları kullanılarak Şekil 6.3’deki ağ yapısına, Tablo 6.1 ve Tablo 6.2’deki ağırlıklara bağlı kalarak 3 katmanlı 4 giriş iki çıkışlı olarak elde edilen YSA’nın blok diyagramı Şekil 6.22’deki gibidir.

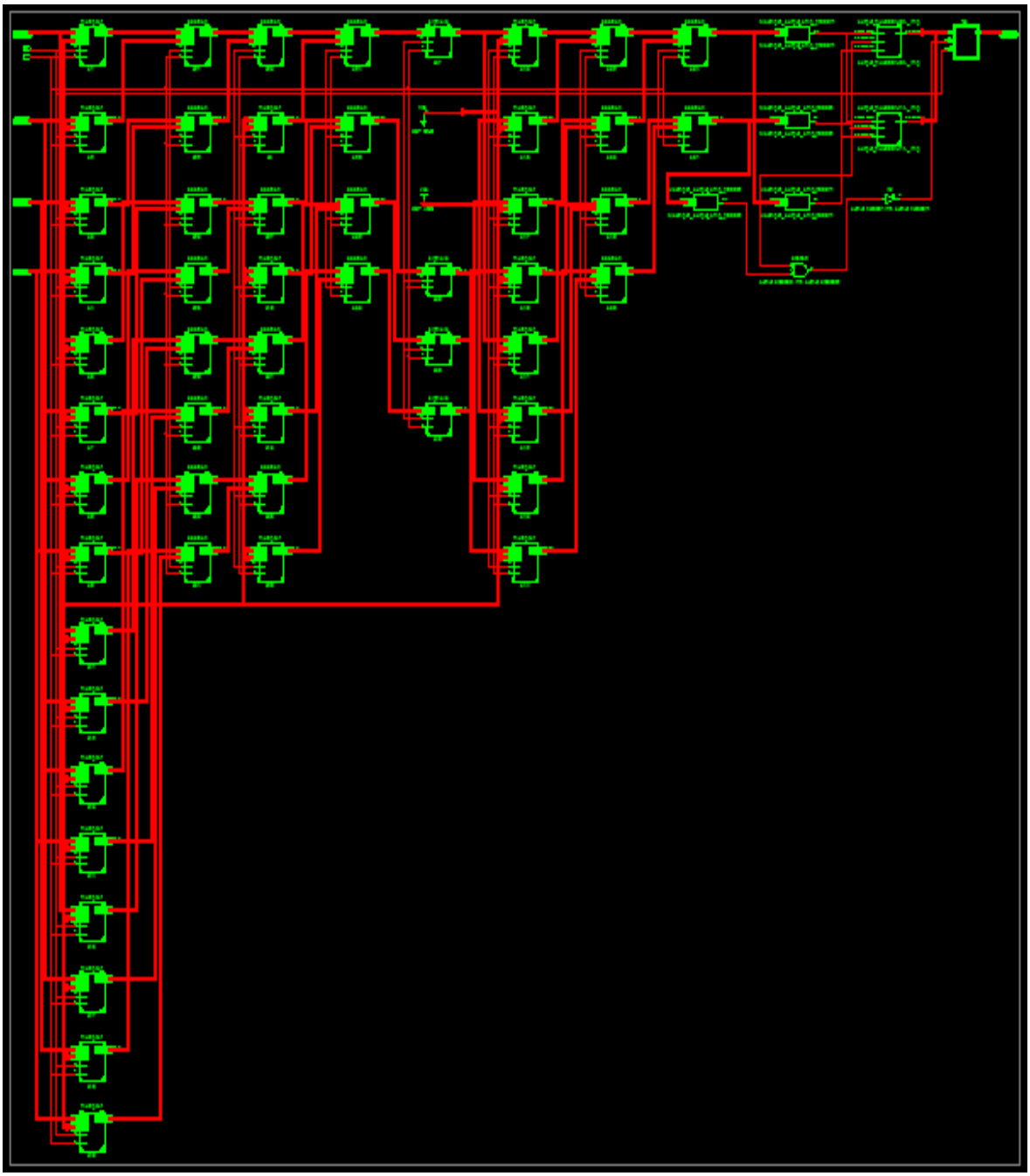


Şekil 6.20 Yapay Sinir Ağı genel yapısı



Şekil 6.21 Yapay sinir ağı blok diyagramı

Tasarlanan blokta “clk”: 50 Mhz saat işareti girişi, “rst”: Ağın çıkışını istenildiği zaman sıfırlayan giriş, “input1”, “input2”, “input3”, “input4”: sınıflandırılacak veri kümesinin tasarlanan YSA donanımında sınıflandırma performansının değerlendirilmesi için test girişlerini oluşturmaktadır. Girişler 20 bitlik verilerdir. Output1 ve output2 sınıflandırma sonucu elde edilen 1 bitlik çıkışlardır. İkinci katmanın son toplayıcı çıkışında elde edilen değerin mutlak değeri 0.4 den küçükse çıkışa 0 değeri, 0.6 dan büyükse çıkışa 1 değeri atanacaktır. 0.4-0.6 değerleri arası bölge tanımsızdır(High Z atanır).



Şekil 6.22 Yapay Sinir Ağı RTL Şeması

Sınıflandırılan veri kümesinde 3 grup yer almaktadır. Bu üç sınıftan herhangi birisine ait olan bir örneğin test girişlerinin ağ girişlerine uygulanmasıyla ağ çıkışında elde edilen değer “00” ise girilen örnek verilerin 1. Gruba ait olduğunu, “01” ise 2. Gruba ait olduğunu ve “11” ise 3. Gruba ait olduğunu gösterir. Bunun dışında oluşan durumlar için sınıflandırma tanımsızdır.

Lojik Birimler	**Kullanılan Sayı**	**Mevcut Sayı**	**Yüzde**
Dilim Sayısı	4216	4656	%70
Flip-Flop Dilim Sayısı	3295	9312	%26
Dört Girişli LUT Sayısı	7849	9312	%66
IOB Sayısı	84	232	%36
BUFGMUX Sayısı	1	24	%4
GCLK Sayısı	1	24	%4
BRAM Sayısı	12	20	%60
MULT18X18 Sayısı	20	20	%100

Tablo 6.8 YSA bloğu FPGA kullanım oranları

YSA Sınıflandırma Test Sonuçları:

Test işleminde ağa giriş olarak veri kümesinde yer alan birinci girişe ait bir örneğin verileri girilmiştir. Performans test işlemi veri kümesi ekte yer almaktadır.
input1= -0.4765
input2 = 0.2250
input3 = -0.7780
input4 = -0.8250
Sonuç = “00” olarak elde edilmiştir. Yani veri 1. Gruba aittir.

Şekil 6.23 YSA Sınıflandırma simülasyon sonuçları

Sınıflandırma işlemi doğru olarak yapılmıştır.

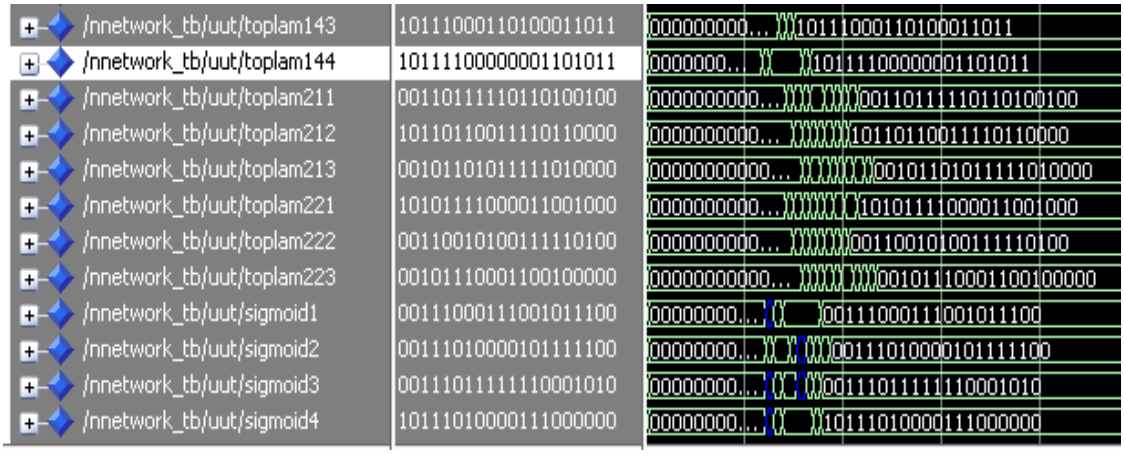
İç sinyaller ayrıntılı şekilde incelendiğinde,

Toplam213, birinci nöron için, toplam223, ikinci nöron için çıkış değeri oluşturmaktadır. Değerlerin mutlak büyüklüğü 0.4’den küçükse çıkışa ‘0’, 0.6’dan büyükse ‘1’ değerini almaktadır. Çıkışların doğru değerleri;

toplam213= 0.0866 ,
toplam223= 0.0950’dir.

Simülasyon sonuçlarına göre,

toplam213=~0.08575,
toplam223=~0.0968’dir
Hata= ((Doğru sonuç-Sonuç)/Sonuç)*100
Hata ~= %1.8 olarak bulunur.



Şekil 6.24 YSA Sınıflandırma ayrıntılı simülasyon sonuçları

Tasarlanan ağın zaman zamanlama özeti:

Timing Summary:

Hız Derecesi: -5

Minimum periyod: 7.675ns (Maximum Frequency: 130.293MHz)

Clok'tan önce girişin minimum varış süresi: 13.890ns

Clock'tan sonra çıkış için gereken maksimum zaman: 4.040ns

6.2.2 FPGA Tabanlı RS232 Arayüzü Tasarımı

Bu bölümde YSA girişine test verilerinin uygulanmasında kullanılacak olan Spartan-3E Starter kiti üzerinde yer alan RS232 seri kanal arabirimine ve bilgisayarla FPGA'in haberleştirilmesine yer verilmiştir.

6.2.2.1 UART Seri Haberleşme Protokolü

"Universal Asynchronous Receiver/Transmitter" UART olarak kısaltılmaktadır. Asenkron alıcı/verici yapısıdır. UART genelde entegre devrelerin bir parçası olarak bilgisayar ya da ara yüz aygıtın seri potu üzerinden seri haberleşme için kullanılır.

UART kontrolörü bilgisayar alt sistemlerinde seri haberleşmede anahtar bileşendir. UART veriyi byte olarak alır ve ardışıl olarak her bir biti gönderir. Hedef noktada, bu bitler byte olarak toplanır. Her bir UART kaydırma yazmacı içerir. Bu kaydırma yazmacı seri ve paralel yapıyı dönüştürmede önemli bir metoddur.

Haberleşme "full dublex" (sinyali gönderme ve alma işleminin aynı anda olması) ya da "half dublex" (aynı anda sadece gönderme veya alma işleminin olması) olabilir.

Asenkron iletimde, "teletype" tarzı UART'lar başlangıç biti,5-8 veri biti (LSB 1. Bittir), opsiyonel "parity bit"i ve dur biti gönderir. Başlangıç biti veri hattının boş olmama durumudur. Dur biti veri hattının boş olma durumudur ve bu bir sonraki karakterin başlayabilmesi için bir gecikme sağlar. Buna asenkron başla-dur iletimi denir.

Senkron iletimde saat verisi akışından ayrı olarak tutulur ve başla-dur bitleri kullanılmaz. Bu daha fazla biti kullanılabilir veri olarak gönderdiği için uygun kanallarda haberleşmenin daha etkili olmasını sağlar.

Asenkron iletimde UART'a bir kelime verildiğinde, başlangıç biti denilen bit iletilecek her bir kelimeye eklenir. Başlangıç biti alıcı kısma veri gönderileceği uyarısını yapmaktadır ve alıcıda alıcı ile verici arasındaki senkronizasyonu saat darbeleri ile sağlamaktadır.

Başlangıç bitinden sonra ilk gönderilen en az ağırlıklı bit olmak üzere kelimenin her bir biti gönderilir. Her bir bit alıcı kısma aynı zaman miktarında iletilir ve alıcı periyodun yarısı sürede hattı bit 1 ya da 0 mı diye kontrol eder. Gönderici kısmı alıcının bitin değerini kontrol ettiğini bilmez. Gönderici sadece saat darbesi geldiği zaman kelimenin bir sonraki bitini göndermeye başlayacağını bilir.

Gönderilmesi için veri girildiği zaman, verici kendisinin oluşturduğu bir "parity bit"i ekler. Bu "parity bit" alıcı tarafından basit hata kontrolü için kullanılır. Son olarak vericiden dur biti gönderilir.

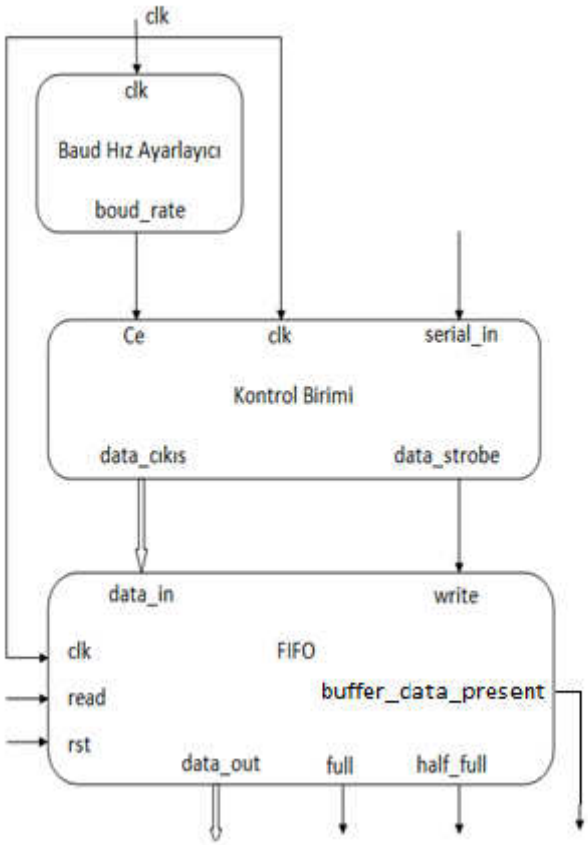
Alıcı kelimenin tüm bitlerini aldığı zaman, "parity bit"i kontrol edilecektir ve dur bitine bakacaktır. UART okuma yapıyorken dur bitini beklediği zaman alamazsa UART girilen kelimenin yanlış olduğunu düşünecek ve merkezi işlem gören kısma hata verecektir. Buna çerçeve hatası denir ve alıcı ve vericinin saat hızlarının aynı olmadığı ya da iletimin yarım kaldığı durumlardır.

Şekil 6.25 UART standardı

Başlangıç biti her zaman lojik 0'dır ve buna space (boşluk) denir. Sonraki gelen 5-8 bit kullanıcı ya da üreticinin ayarladığı karakterdir. ASCII kod sekizinci biti "parity bit" olarak ayarlanabilir. Sonraki bir ya da iki bit her zaman lojik 1 dir ve dur biti olarak çağrılır.

6.2.2.2 RS232 Alıcı Bloğu Tasarımı

Proje kapsamında FPGA ile seri kanaldan gelen verileri alabilmek için Uart alıcı bloğu tasarlanmıştır. Bu blok temek olarak üç yapıdan oluşmaktadır. RS232 ile FPGA in senkron çalışabilmesi için boud hız ayarlayıcı, seri girişten gelen verileri, oluşturulan baud hızıyla alarak buffera yazma işlemi kontrol birimi ve FIFO tipi bufferdan oluşmaktadır.



Şekil 6.26 RS232 Seri alıcı ara yüzü blok diyagramı

Boud hız ayarlayıcı: Seri kanal baud hız ayarını yapmak için kullanılmaktadır. Oluşturulan sistemde 38400 Hz Baud rate kullanılmaktadır. 50 Mhz lik saat işaretini 38400 Hz lik saat işaretine dönüştürme işlemi yapar.

Kontrol Birimi: RS232 seri girişten gelen verileri, oluşturulan baud hızıyla alarak buffera vermekte ve buffera verilerin yazılmasını sağlayan kontrol işaretini üretmektedir.

FIFO: Gelen seri verileri kaydederek 8 bitlik bir paket halinde kullanılmak üzere dışarıya vermektedir.

Tasarlanan blokta "Serial_in": Standart 8 bitlik giriş verisi olarak kullanılmaktadır. Buffer dolu değilse otomatik olarak veriler buffera yazılır. "Data_out": gelen 8 bit verinin paralel çıkışıdır. "read": buffer ile verilen 8 bitlik paralel verinin okunduğunu bildirir. "rst":16 bytelık bufferın resetlenmesini sağlayan giriştir. "ce": Belirlenen Baud hızını bildiren giriştir. "Buffer_full": Bufferın dolduğu ve yeni veri alacak yer bulunmadığını belirten kontrol

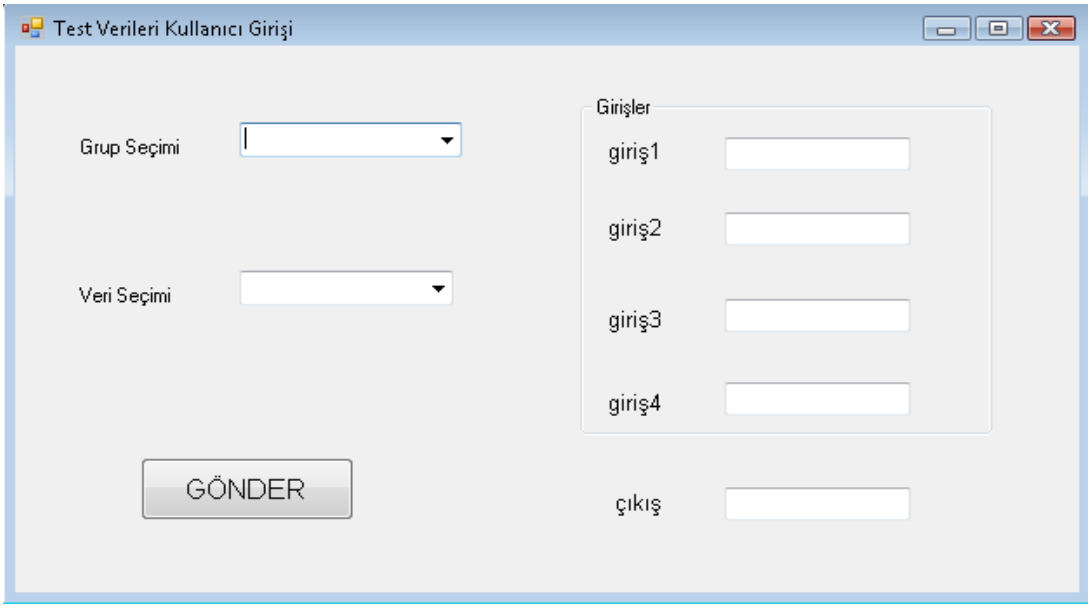
işaretidir.“Half_full”:16 byte bufferda 8 bytelık ya da daha fazla verinin henüz okunmadığını belirten kontrol işaretidir. “clk”: 50 Mhz lik saat işaretidir.

Lojik Birimler	Kullanılan Sayı	Mevcut Sayı	Yüzde
Dilim Sayısı	83	4656	%1
Flip-Flop Dilim Sayısı	92	9312	%1
Dört Girişli LUT Sayısı	98	9312	%1
IOB Sayısı	5	232	%2
BUFGMUX Sayısı	1	24	%4
GCLK Sayısı	1	24	%4

Tablo 6.9 RS232 Seri Ara yüzü FPGA kullanım Oranları

6.2.2.3 YSA Test Girişleri Kullanıcı Arayüzü Tasarımı

Bu tasarımda, C# programı ile bir kullanıcı arayüzü gerçekleştirilmiştir. Bu arayüz ile kullanıcı test etmek istediği veri değerlerini FPGA üzerinde tasarlanmış olan alıcıya seri kanal yardımıyla göndermektedir. Girişler alıcı blok aracılığıyla YSA girişlerine uygulanmaktadır.



Şekil 6.27 Kullanıcı arayüzü

Kullanıcı arayüzünde, “Grup Seçimi” ComboBox’ı, sınıflandırma yapılacak test girişlerine ait grup isimlerini bulundurmaktadır. Buradan seçilen gruba göre gruba ait örnekler “Veri Seçimi” ComboBox’ına yüklenir. “Veri Seçimi” ComboBox’ı ile test edilecek girişler seçildikten sonra “GÖNDER” Buton’u ile seri arabirim üzerinden FPGA’e aktarılır. Ayrıca gönderilen girişler ve sınıflandırma sonucu beklenen değer “çıkış” TextBox’ında gözlemlenmektedir. Test verileri Ek 2 de yer almaktadır.

6.2.3 Spartan-3E LCD Modülü Kullanımı

Bu bölümde Spartan-3E Starter kiti üzerinde yer alan LCD modül kullanımından bahsedilmiştir.

6.2.3.1 Karakter LCD Modül

Kit üzerinde bulunan LCD modül 2 sıra 16 karakter (toplam 32 karakter) gösterebilen bir yapıdadır. LCD modül 8 bit veri yolunu desteklese de kit üzerindeki FPGA pinlerinin daha efektif kullanılması bakımından 4 bitlik bir veri yolu ile sürülmektedir. [7]

LCD modül birçok uygulamada ASCII karakterleri ve bizim üretmiş olduğumuz karakterleri göstermek için oldukça uygundur. Buna rağmen LCD display modül kit üzerindeki 50 Mhz lik saat sinyali ile karşılaştırıldığında oldukça yavaş kalmaktadır. Bu yüzden modül belirli zaman gecikmeleri ile sürülmelidir. Bu zaman gecikmeleri PicoBlaze işlemcisi kolayca yapılabilmektedir. [7] Fakat tasarlanan bu blokta gecikmeler kod ile ayarlanmıştır.

LCD modül üzerinde bulunan StrataFlash hafıza LCD ye yazdırmak istenilen verileri depolamaktadır. StrataFlash ilgili sinyalleri, LCD modülün pinleri ve bu pinlerin nasıl kullanılacağını Tablo 6-10'da gösterilmektedir.

Sinyal İsmi	FPGA Pinleri	Fonksiyon	
SF_D<11>	M15	Data bit DB7	StrataFlash Pinleri ile paylaşımlıdır. SF_D<11:8>
SF_D<10>	P17	Data bit DB6	
SF_D<9>	R16	Data bit DB5	
SF_D<8>	R15	Data bit DB4	
LCD_E	M18	Read/Write Enable darbesi 0: Disable 1:Read/Write Enable	
LCD_RS	L18	Register seçim 0:Yazma operasyonları esnasında komut registerı. 1:Read/Write operasyonları için veri	
LCD_RW	L17	Read/Write Kontrol 0: Yazma. LCD verileri alır. 1: Okuma. LCD verileri okunur.	

Tablo 6.10 Fonksiyon sinyalleri[7]

6.2.3.1.1 LCD Modül Hafıza Alanları

A. DDRAM

Ekranda gösterilecek karakterlerin tutulduğu alandır. DDRAM de tutulan değer CG RAM ve CG ROM a referans gösterir.[7] Şekil 6.28'de DDRAM alanındaki adreslerin ekranda nereye karşılık geldiğini yer almaktadır.

B. CG ROM

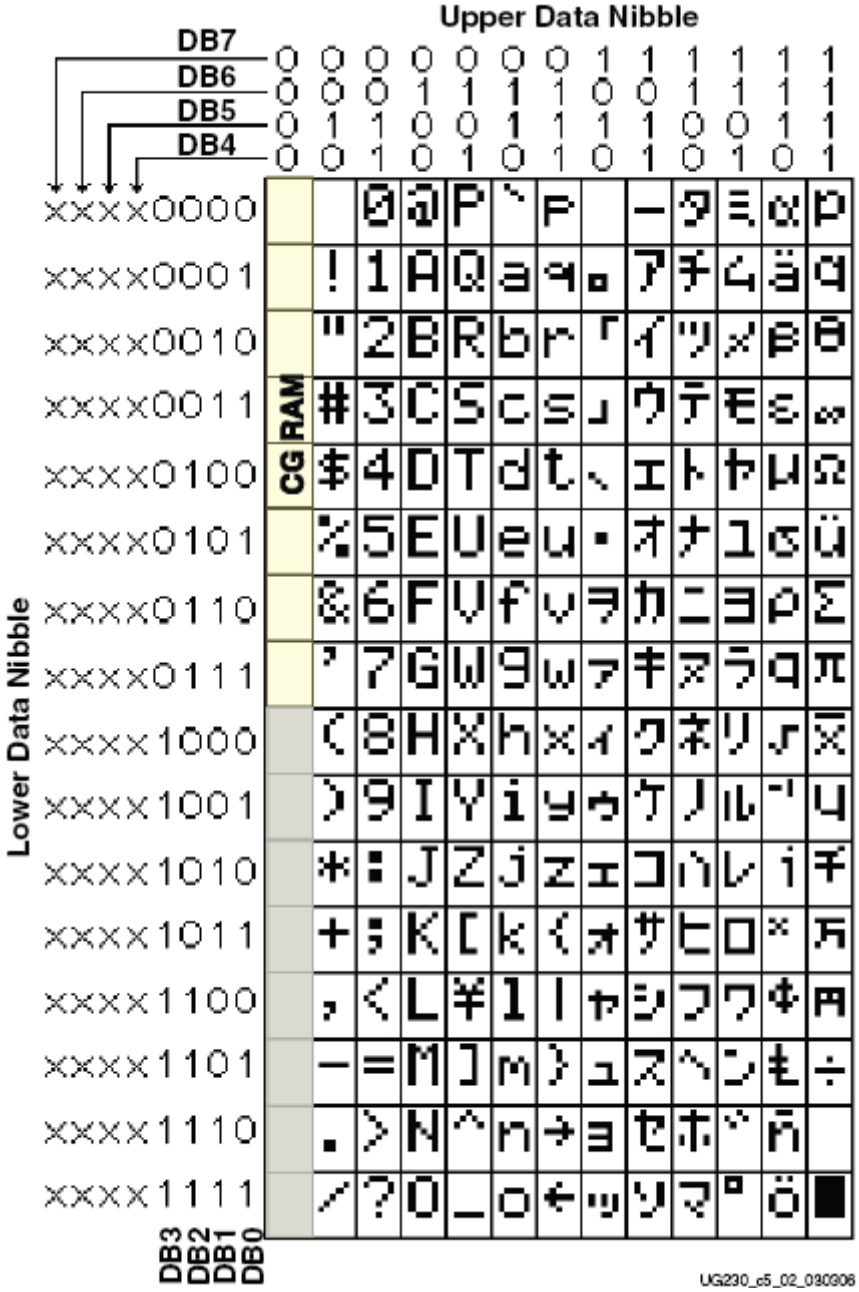
CG ROM (Character Generation ROM) LCD modülün gösterebileceği karakterlerin saklandığı okunabilir bir hafıza bloğudur. LCD modülde her karakter 5*8 lik bir matris ile ifade edilmektedir. CG ROM alanındaki önceden tanımlı karakterlerin gösterimi için bu alana referans verilmesi gerekmektedir.[7]

C. CG RAM

CG RAM alanı DDRAM ile 0x00 ile 0x07 arasında adresleme yapıldığında kullanılabilir durumdadır. Bu alan kullanılmadan önce kullanıcı tarafından tanımlanan karakterler bu alana doğru bir şekilde girilmelidir.

CG RAM alanına bir karakter girmek için öncelikle CG RAM adres seçme komutu ve bunun ardından da yazma komutunu çalıştırılmalıdır. Adres girme komutu çağırıldıktan sonra girilen ilk 3 bit sıra sayısına sonraki 3 bit de

CGRAM adresine karşılık gelmektedir. Daha sonra CG RAM yazma komutu ile data bitlerinden alt 5 bit data olarak ele alınır. Data alanındaki 1'ler o alanın ışık vereceğini 0'lar ise sönük kalacağını göstermektedir. Kullanılmayan 3 bit ise diğer komutlara uygunluk açısından "0" seçilebilir.[7]



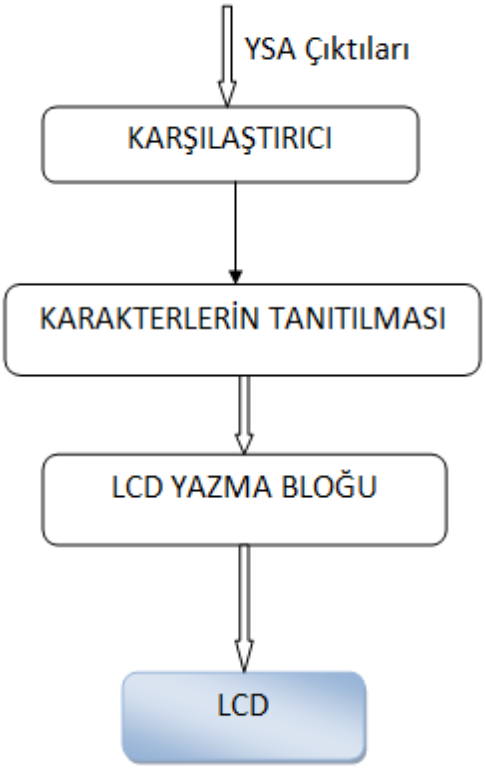
Şekil 6.28 Karakter tanımları[7]

6.2.3.2 LCD Ekranına Yazı Basmak

LCD ekranına yazı yazmak için öncelikle ilgili DDRAM alanına veri girilmelidir. Bunun için öncelikler "Set DDRAM address" komutu ile adres belirtildikten sonra CG RAM ve ROM içerisindeki karakterlere göre veri girilmelidir. Adres otomatik arttırma ya da azaltma modunda ise yeniden adres girmeden veri girilebilir. Veri girişi "Write data to CG RAM or DDRAM" komutu ile gerçekleştirilir.

6.2.3.3 LCD Modül Tasarımı

Tasarımda YSA çıktılarına bağlı olarak veri sınıflarının karakter LCD üzerinde yazması tasarlanmıştır. Program YSA çıktılarının "00" olması durumunda LCD'e "1. GRUP ", "01" olması durumunda "2. GRUP ", "11" olması durumunda 3. GRUP ve bunların dışındaki durumlarda "TANIMSIZ" yazacak şekilde tasarlanmıştır.

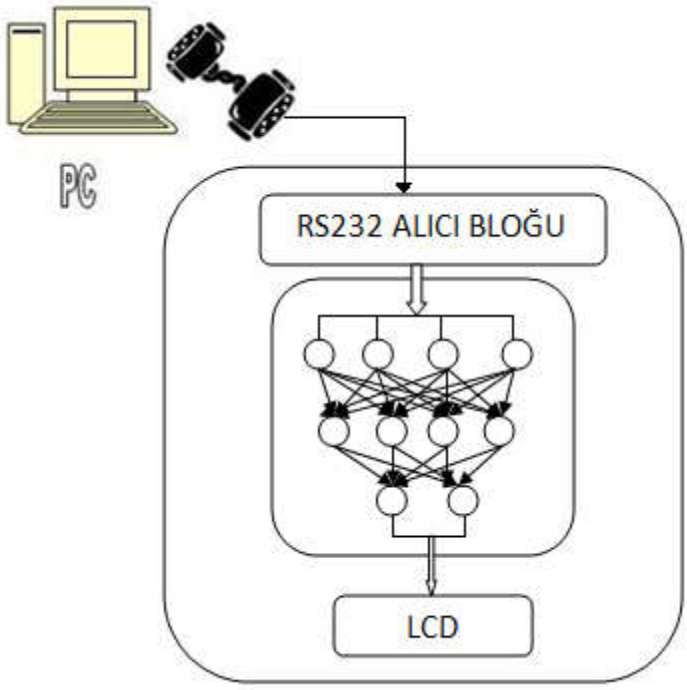


Şekil 6.29 LCD modül blok diyagramı

Lojik Birimler	Kullanılan Sayı	Mevcut Sayı	Yüzde
Dilim Sayısı	95	4656	%1
Flip-Flop Dilim Sayısı	81	9312	%1
Dört Girişli LUT Sayısı	129	9312	%2
IOB Sayısı	20	232	%8
BUFGMUX Sayısı	1	24	%4
GCLK Sayısı	1	24	%4

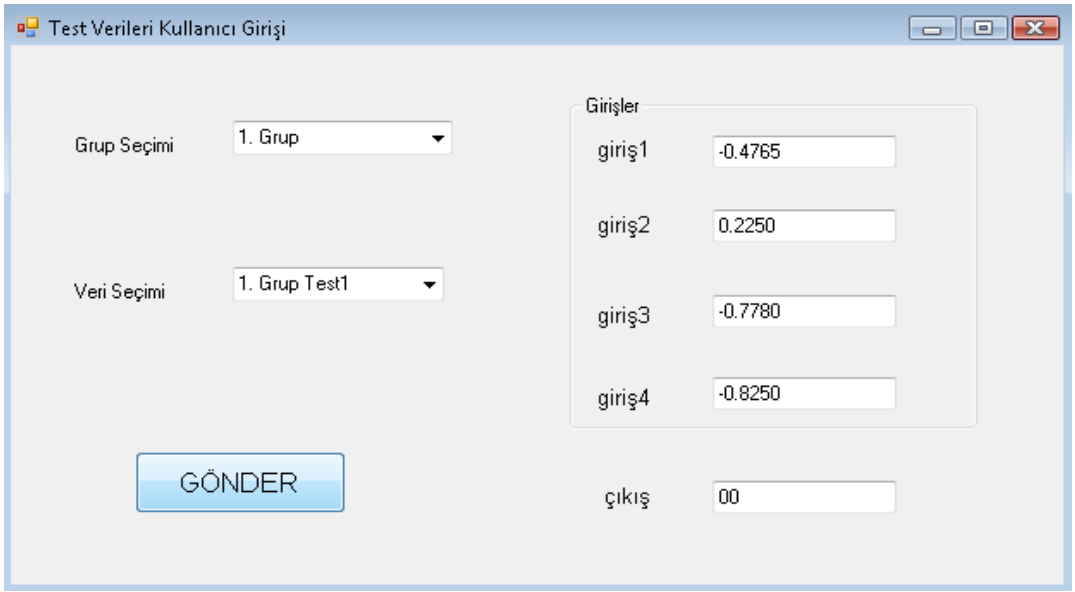
Tablo 6.11 LCD Modül FPGA kullanım oranları

6.3 Tasarlanan Sistem ve Çalışmasının Gözlemlenmesi



Şekil 7.1 Sistemin Blok diyagramı

Sınıflandırma işleminin test edilmesi;



Şekil 7.2 Birinci gruba ait test giriş değerleri gönderimi kullanıcı ara yüzü

Test için gönderilen birinci gruba ait verilerin FPGA'e gömülü YSA üzerinde sınıflandırma sonucu Şekil 7.3 de gösterilmektedir.

Şekil 7.3 Sınıflandırma sonucunun LCD üzerinde gözlemlenmesi

7. SONUÇLAR

Bu projede FPGA üzerinde VHDL donanım tanımlama dili kullanılarak birçok katmanlı yapay sinir ağı tasarlanmıştır. Bilgisayar ortamında eğitilerek elde edilen ağırlık değerleri ağa uygulanmış ve bir veri kümesinin sınıflandırma performansı ağın daha önce görmediği test verileriyle gözlemlenmiştir.

Ağın tasarlanması aşamasında, ağın matematiksel alt yapıları blok blok tasarlanıp test edildikten sonra ağ paralel yapısı oluşturulmuştur. Bu bloklar toplama, çarpma ve aktivasyon fonksiyonu olarak alınan sigmoid fonksiyonu işlemlerini gerçekleyen yapılardır. Matematiksel işlemlerinin gerçeklenmesinde sayı gösterimindeki dinamiklik ve hassas işlem yapma kapasitesi sebebiyle kayan noktalı sayı(floating point number) sistemi kullanılması tercih edilmiştir.

Sigmoid aktivasyon fonksiyonu tasarlanırken bazı zorluklarla karşılaşılmıştır. Doğrusal olmayan bu fonksiyonun matematiksel tanımı ile tasarımı birçok toplama ve çarpma devreleri gerektirdiği için slikon alanında çok fazla yer kaplamaktadır. Bu sebepten sigmoid fonksiyonunu gerçeklerken seri açılımları kullanmak yerine LUT(Look Up Table) kullanılarak tasarlanmasına karar verilmiş ve 1280 değerli bir LUT oluşturulmuştur.

Ağa test girişleri RS232 seri kanal kullanılarak PC üzerinde hazırlanan bir arayüz programı ile aktarılmış, ağın sınıflandırma sonuçları LCD üzerine bastırılmıştır.

Kitin kapasitesinin yetersiz olması sebebiyle IEEE 754 kayan nokta standardı(32 bit) yerine 20 bitlik kayan noktalı sayı sistemi kullanılmıştır. Bu optimizasyon hassasiyetin azalması ile sonuçlanmıştır. Buda matematiksel bloklarda hata oluşmasına dolayısıyla ağın tamamında hataya sebep olmuştur.

32 bit kayan nokta formatı kullanılarak sistemdeki hatalar giderilip hassasiyet artırılabilir. Diğer bir yöntem olarak fix point kullanılarak işlemler, daha az sayıda bit ile ve daha kolay bir şekilde gerçekleştirilebilir. Ayrıca sistem veri kümesinden test işlemi yapılası yerine CMOS görüntü sensörü kullanılarak çok daha kullanışlı bir hale getirilebilir.

8.KAYNAKLAR

[1] http://www.yapay-zeka.org
[2] Çavuşlu, Ali,(2006), "FPGA ile yapay sinir ağı eğitiminin donanımsal gerçeklenmesi", Kocaeli.
[3]http://members.comu.edu.tr/boraugurlu/courses/bm434/week4/hafta4.pdf
[4] BELEN, Selçuk,(2008), "FPGA Teknolojisi Kullanarak Sayısal Haberleşme Sinyallerinin Süzülmesi", Dokuz Eylül Üniversitesi, Elektrik Elektronik Mühendisliği, İzmir.
[5] Hilal GÜNEREN,(2009), Aselsan Staj Dökümanları
[6] Çavuşlu, A, "Kayan Noktalı Sayılarla Çarpma İşlemi"
[7] Spartan-3E Starter Kit User Guide
[8] http://pages.cs.wisc.edu/~smoler/x86text/lect.notes/arith.flpt.html
[9]http://www3.itu.edu.tr/~orencik/BilgMimYenYakl2007/Mehmet_Aktas/FPGA_Mimarisi_Rapor.pdf
[10] http://tr.wikipedia.org/wiki/Yapay_sinir_a%C4%9Flar%C4%B1

EKLER

Ek 1 Spartan-3E Starter Kit katalog bilgisi

Spartan-3E FPGA Starter Kit Board User Guide

UG230 (v1.1) June 20, 2008

XILINX®

Ek 2 Test veri kümesi

TEST_Giriş

-0.4765	0.2250	-0.7780	-0.8250
-0.5824	-0.1500	-0.7780	-0.8250
-0.6882	0	-0.8085	-0.8250
-0.7412	-0.0750	-0.7475	-0.8250
-0.5294	0.3000	-0.7780	-0.8250
-0.3176	0.5250	-0.6864	-0.6750
-0.7412	0.1500	-0.7780	-0.7500
-0.5294	0.1500	-0.7475	-0.8250
-0.8471	-0.2250	-0.7780	-0.8250
-0.5824	-0.0750	-0.7475	-0.9000
-0.3176	0.3750	-0.7475	-0.8250
-0.6353	0.1500	-0.7169	-0.8250
-0.6353	-0.1500	-0.7780	-0.9000
-0.9000	-0.1500	-0.8695	-0.9000
-0.1059	0.6000	-0.8390	-0.8250
-0.1588	0.9000	-0.7475	-0.6750
-0.3176	0.5250	-0.8085	-0.6750
-0.4765	0.2250	-0.7780	-0.7500
-0.1588	0.4500	-0.6864	-0.7500
-0.4765	0.4500	-0.7475	-0.7500
-0.3176	0.1500	-0.6864	-0.8250
-0.4765	0.3750	-0.7475	-0.6750
-0.7412	0.3000	-0.9000	-0.8250
-0.4765	0.0750	-0.6864	-0.6000
-0.6353	0.1500	-0.6254	-0.8250
0.5294	0	0.2288	0.0750
0.2118	0	0.1678	0.1500
0.4765	-0.0750	0.2898	0.1500
-0.2647	-0.6750	0.0153	0
0.2647	-0.3000	0.1983	0.1500

-0.1588	-0.3000	0.1678	0
0.1588	0.0750	0.2288	0.2250
-0.5824	-0.6000	-0.1983	-0.2250
0.3176	-0.2250	0.1983	0
-0.4235	-0.3750	-0.0153	0.0750
-0.5294	-0.9000	-0.1373	-0.2250
-0.0529	-0.1500	0.0763	0.1500
0	-0.7500	0.0153	-0.2250
0.0529	-0.2250	0.2288	0.0750
-0.2118	-0.2250	-0.1068	0
0.3706	-0.0750	0.1373	0.0750
-0.2118	-0.1500	0.1678	0.1500
-0.1059	-0.3750	0.0458	-0.2250
0.1059	-0.7500	0.1678	0.1500
-0.2118	-0.5250	-0.0153	-0.1500
-0.0529	0	0.2593	0.3750
0.0529	-0.3000	0.0153	0
0.1588	-0.5250	0.2898	0.1500
0.0529	-0.3000	0.2288	-0.0750
0.2118	-0.2250	0.1068	0
0.1588	0.0750	0.6254	0.9000
-0.1059	-0.3750	0.3508	0.4500
0.5824	-0.1500	0.5949	0.6000
0.1588	-0.2250	0.5034	0.3750
0.2647	-0.1500	0.5644	0.6750
0.8471	-0.1500	0.8085	0.6000
-0.5824	-0.5250	0.1678	0.3000
0.6882	-0.2250	0.7169	0.3750
0.3706	-0.5250	0.5644	0.3750
0.6353	0.3000	0.6559	0.9000
0.2647	0	0.3508	0.5250
0.2118	-0.3750	0.4119	0.4500
0.4235	-0.1500	0.4729	0.6000

-0.1588	-0.5250	0.3203	0.5250
-0.1059	-0.3000	0.3508	0.8250
0.2118	0	0.4119	0.7500
0.2647	-0.1500	0.4729	0.3750
0.9000	0.4500	0.8390	0.6750
0.9000	-0.4500	0.9000	0.7500
0	-0.7500	0.3203	0.1500
0.4765	0	0.5339	0.7500
-0.2118	-0.3000	0.2898	0.5250
0.9000	-0.3000	0.8390	0.5250
0.1588	-0.3750	0.2898	0.3750
0.3706	0.0750	0.5339	0.6000

İÇİNDEKİLER **Sayfa**

SİMGE LİSTESİ

Bit	İkili düzendeki her bir rakam (0 ve 1)
Bayt	8 bitten oluşan ikili sayı kümesi
Hz	Frekans birimi

KISALTMA LİSTESİ

ASCII	American Standard Code for Information Interchange
CLB	Configurable Logic Block
FPGA	Field Programmable Gate Array
FPL	Field Programmable Logic
FPN	Floating Point Number
IEEE	Institute of Electrical and Electronics Engineers
LCD	Liquid Crystal Display
LSB	Least Significant Bit
LUT	Look Up Table
MLP	Multi Layer Perceptron
MSB	Most Significant Bit
PC	Personel Computer
RAM	Random Access Memory
ROM	Read Only Memory
UART	Universal Asynchronous Receiver/Transmitter
UCF	User Constraints File
XOR	Exclusive Or
VHDL	*VHSIC* Hardware Descpription Language
VLSI	Very Large Scale Integration
YSA	Yapay Sinir Ağı

ŞEKİL LİSTESİ **Sayfa**

ÇİZELGE LİSTESİ **Sayfa**

ÖNSÖZ

Yapay sinir ağları, olayların örneklerine bakıp, bu örneklerin dağılımlarını öğrenip hakkında genellemeler yapan, bilgiler toplayan ve daha sonra hiç görmediği örnekler ile karışılışınca öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilen yapılardır.

Bu projede çok katmanlı algılayıcı tipi yapay sinir ağları incelenmiş ve FPGA üzerinde VHDL programlama dili kullanılarak tasarımı gerçekleştirilmiştir. Ağın sınıflandırma performansının donanım üzerinde gözlemlenebilmesi için proje kapsamında kullanılan Spartan-3E Starter kiti üzerinde yer alan seri kanal ve LCD modüller kullanılmıştır. Seri kanal ağa uygulanacak test girişlerini aktarmakta LCD ise ağın sınıflandırma performansının gözlemlenebilmesini sağlamaktadır.

Tez çalışmalarım sırasındaki özverili yardımlarından dolayı danışman hocam Yrd. Doç. Dr. Burcu ERKMEN'e, her türlü yardım ve fikirlerini esirgemeyen hocalarım Araş. Gör. Evren CESUR, Araş. Gör. Nerhun YILDIZ ve arkadaşım Melike ATAY'a teşekkürlerimi sunarım. Ayrıca bu yoğun dönemimde manevi desteklerini esirgemeyen ailem ve ev arkadaşlarıma şükran borçluyum.

Projeme verdikleri maddi desteklerinden dolayı Elektrik Mühendisleri Odası İstanbul Şubesine teşekkür ederim.

Hilal GÜNEREN

Mayıs 2010

ÖZET

Yapay Sinir Ağları(YSA), insan beyninin özelliklerinden olan öğrenme yolu ile yeni bilgiler türetebilme, yeni bilgiler oluşturabilme ve keşfedebilme gibi yetenekleri, herhangi bir yardım almadan otomatik olarak gerçekleştirebilmek amacı ile geliştirilen bilgisayar sistemleridir. Yapay sinir ağları, olayların örneklerine bakmakta, onlardan ilgili olay hakkında genellemeler yapmakta, bilgiler toplamakta ve daha sonra hiç görmediği örnekler ile karışılışınca öğrendiği bilgileri kullanarak o örnekler hakkında karar verebilmektedirler. YSA doğasında paralel bilgi akışına sahiptir. Bu sebeple gerçek başarımlarını ancak paralel çalışan mimariler üzerinde gösterebilirler.

Bitirme tezi kapsamında yapay sinir ağı algoritmalarından biri olan Çok Katmanlı Algılayıcılar(MLP) hakkında bilgi edinilmiş ve paralel matematiksel yapısı donanımsal olarak FPGA üzerinde VHDL programlama dili kullanılarak gerçekleştirilmiş, ağın sınıflandırma performansı gözlenmiştir.

Tasarım aşamasında Xilinx firmasının Spartan-3E adlı kiti ve üzerinde bulunan xc3s500e kodlu FPGA kullanılmıştır. Program Xilinx firmasının ISE 11.1 Project Navigator kullanılarak derlenmiştir. Gerekli simülasyonlar ModelSIM kullanılarak gerçekleştirilmiştir.

Ağ tasarımında aktivasyon fonksiyonu olarak sigmoid fonsiyonu kullanılmıştır. Ağ algoritmasında yer alan matematiksel blokların tasarımında sayı gösterimindeki dinamiklik ve hassas işlem yapma kapasitesi sebebiyle kayan noktalı sayı(floating point number) sistemi tercih edilmiştir. Öncelikle kayan nokta sayı sistemi formatında toplama çarpma ve sigmoid fonksiyonu blokları tasarlanmış ve daha sonra bu bloklardan genel ağ yapısı oluşturulmuştur. Tasarlanan ağ yapısının sınıflandırma performansı simülasyonlarla test edildikten sonra kart test aşamasına geçilmiştir. Performansın kit üzerinde gözlemlenebilmesi için kit üzerinde yer alan seri kanal ve LCD kullanılmıştır. Sınıflandırma test verileri, C# ile hazırlanan ara yüz programı yardımıyla PC üzerinden seri kanal ile ağ girişine uygulanmıştır. Ağın sınıflandırma performansı kit üzerinde yer alan LCD' de gözlemlenmiştir.

Anahtar Kelimeler: Çok katmanlı yapay sinir ağı, kayan noktalı sayılar, toplama, çarpma, sigmoid fonksiyonu, seri kanal, LCD, FPGA, VHDL,

ABSTRACT

Artificial neural network is a designed computer system which aims to realize some abilities such as creating and discovering new ideas without any help through the way of learning which is one of the most important abilities of the brain. Artificial neural network looks at the samples of the events, makes some generalizations about the event which is related to them and then whenever it encounters with new samples, it can give decisions about them by using learned information. Artificial neural network has parallel information flow in its nature. For this reason, it can show its real ability only on the architectures which are working parallelly.

As a thesis, some information are gotten about Multi Layer Perceptron(MLP) which is one of the artificial neural network algorithms, parallel mathematical structure is made on FPGA as hardware by using VHDL as a programming language, the classification performances of the network is observed.

At the stage of design, the kit of Xilinx firm which is named Spartan-3E and a FPGA with code xc3s500e.The program is organized by using ISE 11.1 Project Navigator of the firm Xilinx. The necessary simulations are provided by using ModelSIM.

At the stage of neural design, sigmoid function is used as activation function. At the stage of mathematical blocks in network algorithm, floating point system is preferred because it has dynamic number system and the capacity to make accurate calculations. Firstly the blocks of addition, multiplication, sigmoid function are designed and then general network structure is made by these block. After the classification performance of designed network structure is tested by simulations, the stage of card test is started. In order to be observed the performance on the kit, serial channel on the kit is used. The data of classification test is implemented to the input of network through PC and serial channel with the help of interface program. The classification performance of network is observed on the LCD which takes part on the kit.

Key Words: Multi Layer Artificial Neural Network , floating point numbers, addition, multiplication, sigmoid function, serial channel, LCD, FPGA, VHDL.

YILDIZ TEKNİK ÜNİVERSİTESİ

ELEKTRİK – ELEKTRONİK FAKÜLTESİ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ BÖLÜMÜ

Bitirme Projesi

FPGA ÜZERİNDE KAYAN NOKTA SAYI FORMATI KULLANILARAK YAPAY SİNİR AĞI TABANLI SINIFLANDIRMA İŞLEMİ

PROJE SAHİBİ

HİLAL GÜNEREN

PROJE DANIŞMANI

YRD. DOÇ. DR. BURCU ERKMEN

İstanbul,2010