T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ALANDA PROGRAMLANABİLİR NÖRON DİZİSİ TASARIMI

Osman YÜKSEL

YÜKSEK LİSANS TEZİ

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Programı

Danışman

Prof. Dr. Burcu ERKMEN

Ocak, 2024

T.C.

YILDIZ TEKNİK ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ALANDA PROGRAMLANABİLİR NÖRON DİZİSİ TASARIMI

Osman YÜKSEL tarafından hazırlanan tez çalışması 22.01.2024 tarihinde aşağıdaki jüri tarafından Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Haberleşme Mühendisliği Anabilim Dalı, Elektronik Programı **YÜKSEK LİSANS TEZİ** olarak kabul edilmiştir.

Prof. Dr. Burcu ERKMEN
Yıldız Teknik Üniversitesi
Danışman

Jüri Üyeleri

Prof. Dr. Burcu ERKMEN, Danışman
Yıldız Teknik Üniversitesi _____

Doç. Dr. Sadiye Nergis TURAL POLAT, Üye
Yıldız Teknik Üniversitesi _____

Prof. Dr. Neslihan ŞENGÖR, Üye
İstanbul Teknik Üniversitesi _____

Danışmanım Prof. Dr. Burcu ERKMEN sorumluluğunda tarafımca hazırlanan "Alanda Programlanabilir Nöron Dizisi Tasarımı" başlıklı çalışmada veri toplama ve veri kullanımında gerekli yasal izinleri aldığımı, diğer kaynaklardan aldığım bilgileri ana metin ve referanslarda eksiksiz gösterdiğimi, araştırma verilerine ve sonuçlarına ilişkin çarpıtma ve/veya sahtecilik yapmadığımı, çalışmam süresince bilimsel araştırma ve etik ilkelerine uygun davrandığımı beyan ederim. Beyanımın aksinin ispatı halinde her türlü yasal sonucu kabul ederim.

Osman YÜKSEL

İmza

Aileme

TEŞEKKÜR

Çalışmam boyunca benden desteğini esirgemeyen aileme, arkadaşlarıma ve danışman hocama teşekkürü bir borç bilirim.

Osman YÜKSEL

İÇİNDEKİLER

SİMGE LİSTESİ

X	Hexadecimal (Onaltılık) Sayı

KISALTMA LİSTESİ

AXIS	AXI Stream BUS
AXI4Lite	AXI-4 Lite BUS
AXI HP	AXI High-Performance Slave Port
CPI	Cycles Per Instruction
FIFO	First in First Out
SoC	System-on-a-Chip
FPGA	Field Programmable Gate Array
HDL	Hardware Description Language
VHDL	Very High-Speed Integrated Circuit Hardware Description Language
DSP	Digital Signal Processor
SIMD	Single Instruction Multiple Data
SNN	Spiking Neural Network
IF	Integrate-and-Fire Neuron Model
LiF	Leaky-integrate-and-Fire Neuron Model
IZH	Izhikevich Neuron Model
STDP	Spike-timing Dependent Plasticity
MUX	Multiplexer
CPU	Central Processing Unit
DDR	Double Data Rate
RAM	Random Access Memory
PE	Processing Element
FP16	Half-Precision Floating Point
INT8	8-Bit Signed Integer / Byte
FP32	Single Precision Floating Point
UINT16	Unsigned Short
SRM	Spike Response Model
RC	Rate Coding
TTFS	Time-to-first-Spike Coding
RTL	Register Transfer Level
NMC	Neuron Model Calculator
MAC	Multiply and Accumulate

ŞEKİL LİSTESİ

TABLO LİSTESİ

ÖZET

Alanda Programlanabilir Nöron Dizisi Tasarımı

Osman YÜKSEL

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Programı

Yüksek Lisans Tezi

Danışman: Prof. Dr. Burcu ERKMEN

Bu çalışmada programlanabilir nöron modeline ve programlanabilir sinaptik plastisiteye sahip FPGA tabanlı bir nöromorfik donanım tasarımı ele alınmıştır. Literatürde halihazırda var olan programlanabilir lojik tabanlı nöromorfik sistemlerin değişmeyen nöron modelleri içermesi ve sınırlı sinaptik plastisiteye sahip olması nedeniyle bu çalışmanın ana hedefi tümüyle programlanabilir ve öğrenme işlemlerini FPGA üzerinde gerçekleştirebilen bir nöromorfik donanım tasarlamaktır. Çalışmada hedef platform olarak AMD Zynq-7000 SoC platformu kullanılmış olup, nöromorfik donanımın bütün alt modülleri VHDL donanım tanımlama dili ile yazılmıştır. Sistemin testleri PYNQ işletim sistemi ile yine aynı hedef platform üzerinde gerçekleştirilmiştir. Çalışmada FPGA tabanlı bir nöromorfik sistemin tümüyle programlanabilir ve esnek bir yapıda tasarlanabileceği, SoC platformunun da kullanılmasıyla beraber literatürde var olan programlanabilir lojik tabanlı sistemlere kıyasla daha esnek, farklı problemlerin çözümünde kullanılmaya uygun, farklı ağ topolojilerine, nöron modellerine ve öğrenme kurallarına uyum sağlayabilecek bir donanım tasarlanabileceği ortaya konulmuştur.

Anahtar Kelimeler: Atışlı sinir ağları, nöromorfik donanım, FPGA, atımlı nöron modelleri, VHDL.

ABSTRACT

Field Programmable Neural Array Design

Osman YÜKSEL

Department of Electronics and Communications Engineering

Master of Science Thesis

Supervisor: Prof. Dr. Burcu ERKMEN

This study focuses on the design of a neuromorphic hardware with a programmable neuron model and programmable synaptic plasticity based on FPGA. As the existing programmable logic-based neuromorphic systems in the literature contain fixed neuron models and have limited synaptic plasticity, the main aim of this study is to design a completely programmable neuromorphic hardware capable of performing learning processes entirely on FPGA. The AMD Zynq-7000 SoC platform is used as the target platform in this study, and all sub-modules of the neuromorphic hardware are written in VHDL hardware description language. The system tests were conducted on the same target platform using the PYNQ operating system. The study demonstrates that an FPGA-based neuromorphic system can be designed to be entirely programmable and flexible. With the use of the SoC platform, it can be more flexible compared to existing programmable logic-based systems in the literature, suitable for solving different problems, adaptable to different network topologies, neuron models, and learning rules.

Keywords: Spiking neural networks, neuromorphic hardware, FPGA, spiking neuron models, VHDL.

1
GİRİŞ

Günümüzde, bilgisayar teknolojileri insan hayatının her alanında derin etkiler yaratmış durumdadır. Ancak, geleneksel bilgisayar mimarileri, karmaşık problemleri çözmekte sınırlı performans göstermektedir. Bu noktada, insan beyninin işleyişinden ilham alınarak geliştirilen nöromorfik donanımlar ön plana çıkmaktadır.

Nöromorfik donanımlar, yapay zeka, robotik ve benzeri alanlarda bir devrim vaat eden, beynin sinir ağı yapılarına dayanan bir bilgisayar donanımıdır. Nöromorfik donanımlar biyolojik nöronların davranışlarını taklit ederek paralel çalışma yeteneği sunar ve enerji verimliliği açısından önemli avantaj sağlar. Özellikle, büyük veri setlerinin işlenmesi, karmaşık desenlerin tanınması ve öğrenme süreçlerinde potansiyelini ortaya koymaktadır.

Nöromorfik donanımlar, geleneksel yapay sinir ağları yerine atışlı sinir ağları (Spiking Neural Networks) kullanarak hesaplamaları gerçekleştirmektedir. Atışlı sinir ağları 3. nesil yapay sinir ağları olarak adlandırılmakta olup, nöronlar arası haberleşmenin geleneksel yapay sinir ağlarındaki gibi sürekli sayı değerleri yerine ikilik tabanda 1-bit ile ifade edilebilen atımlarla gerçekleştirmesinden ötürü CPU ve GPU gibi genel amaçlı donanımlarda çalışan yazılım sistemleri yerine uygulamaya özel donanım olarak tasarlanmasını daha uygun kılmaktadır. Bu durum da daha hızlı ve esnek sistemlerin yapay zeka uygulamalarında kullanılmasının önünü açmaktadır.

Halihazırda nöromorfik donanım teknolojisinin emekleme aşamasında olması nedeniyle prototip aşamasından çıkmış ve ticari olarak elde edilebilen bir nöromorfik donanımdan bahsedebilmek mümkün değildir. Teknolojinin henüz yeni yeni gelişmeye başlaması nedeniyle nöromorfik donanım geliştirme süreçleri, böyle bir donanımın hızlı bir şekilde tasarlanması ve prototiplenmesi için en uygun platformlar olan programlanabilir lojik kullanılarak yapılmaktadır.

Literatürdeki [1], [2], [3], [4], [5], [6] programlanabilir lojik tabanlı nöromorfik sistemler sistemler çoğunlukla yüksek hızda çıkarım yapan sabit nöron modelleri ile çalışan sistemlerden oluşmaktadır. Önerilen sistemlerden çok az bir kısmı tam anlamıyla programlanabilir ve esnek bir donanım mimarisi sunmaktadır.

Bu tez çalışmasında programlanabilir lojik temelli, programlanabilir nöronlara ve programlanabilir sinaptik plastisiteye sahip bir nöromorfik donanım, en küçük işlem biriminden başlanarak sistem seviyesine gelene kadar tüm alt modülleri, modüllerin işlevleri ve modüllerin tasarım yaklaşımları detaylı bir biçimde ele alınarak adım adım tasarlanmıştır.

Bölüm 2'de nöromorfik hesaplama konsepti, atışlı sinir ağları, nöron modelleri ve öğrenme kuralları kısaca tanıtılmıştır. Bölüm 3'de nöromorfik donanımların temel gereksinimleri, nöromorfik donanım tasarımında en çok kullanılan donanım mimarileri tanıtılmıştır. Bölüm 4'te teze konu olan donanımın tasarlandığı hedef platform, donanım gereksinimleri ve gereksinimleri karşılayacak tasarım yaklaşımları ortaya konulmuştur. Bölüm 5'te önceki bölümlerde ortaya konulan gereksinimleri karşılayacak alt modüller, giriş çıkışları, çalışma prensipleri ve sistemin işlemesinde üstlendiği görevler detaylandırılarak anlatılmıştır. Bölüm 6'da Bölüm 5'te en küçük parçasından başlayarak ortaya çıkarılan sistemin dış dünya ile haberleşmesinde kullanılan donanım arayüzleri tanıtılırken, Bölüm 7 donanımın bu arayüzler yardımıyla programlanmasını ve çalıştırılmasını kapsamaktadır. Bölüm 8'de örnek bir atışlı sinir ağının donanımda çalıştırılmasına ilişkin deneysel sonuçlar verilmiştir.

2
NÖROMORFİK HESAPLAMA

2.1 Nöromorfik Hesaplama Konsepti

Nöromorfik hesaplama konsepti, biyolojik sinir sistemlerinden ilham alarak tasarlanmış yapay sinir ağları ve hesaplama sistemlerini kapsamaktadır. Nöromorfik hesaplama, biyolojik sinir sistemlerinin işleyişinin taklit eden yapay sinir ağları ve yapay nöron modelleriyle karmaşık bilgi işleme görevini gerçekleştirmeyi hedeflemektedir.

Nöromorfik sistemler, geleneksel yapay sinir ağlarından farklı olarak atımlı nöron modelleri ile çalışmaktadır. Atımlı nöron modelleri biyolojik nöronların davranışlarını, diferansiyel denklemler içeren matematiksel modellerle taklit etmektedir.

Nöromorfik hesaplamada nöronlar arası bilgi alış verişi atımlarla (spike) gerçekleştirilmektedir. Atımların doğası gereği basit bilgi yapıları olması nedeniyle nöromorfik sistemler hem enerji verimliliği, hem de bilgi işleme hızı açısından geleneksel yapay sinir ağlarına üstünlük sağlamaktadır.

2.2 Atışlı Sinir Ağları

Atışlı sinir ağları, biyolojik sinir sistemlerinin davranışlarını taklit eden yapay sinir ağlarının bir türüdür. Atışlı sinir ağları geleneksel yapay sinir ağlarından farklı olarak nöronlar arasında bilgi transferinde sürekli değerler yerine, atım adı verilen kısa süreli palslar kullanmaktadır. Atışlı sinir ağları tezin geri kalanında SNN ağları olarak anılacaktır.

SNN ağlarını geleneksel yapay sinir ağlarından ayıran en önemli özellik, SNN ağlarında eğitim sürecinde atımların zamanlamalarının dikkate alınması, nöron modellerinin biyolojik nöronların zamana bağlı diferansiyel içeren matematiksel modellerini kullanması nedeniyle hesaplamalara zaman bilgisinin de dahil edilmesidir.

SNN ağlarında nöronlar, girişlerindeki sinapslara gelen atımların nöronu uyarması ve belirli bir eşiğe ulaştıktan sonra nöronun ateşlenip bağlı olduğu diğer nöronları uyarmasıyla bilginin aktarımını ve işlenmesini gerçekleştirir.

SNN ağlarında eğitim işlemi, sinapsların değerlerinin nöronun ateşlenme zamanına göre ayarlanması ile gerçekleştirilmektedir. Girişlere gelen atımların türevlenebilir olmaması nedeniyle SNN ağlarında gradyan temelli eğitim işlemleri doğrudan uygulanamamaktadır. Gradyan tabanlı öğrenme kuralları bazı matematiksel manipülasyonlar ve ANN-SNN dönüşümleri ile kullanılabilmektedir.

2.3 Sık Kullanılan Nöron Modelleri

2.3.1 Integrate-and-Fire Nöron Modeli

Integrate-and-Fire nöron modeli, SNN ağlarında kullanılan en basit nöron modelidir. 1907 yılında Louis Lapicque [7] tarafından kurbağalar üzerinde yaptığı çalışmalarının sonucunda ortaya konulmuştur. IF nöron modelinin matematiksel ifadesi (2.1) ve reset koşulu (2.2)'de verilmiştir.

$$\frac{\mathrm{dV(t)}}{\mathrm{dt}} = \frac{1}{C} I(t) \tag{2.1}$$

$$\mathrm{V(t)} \geq V_{th} \ , \ V \leftarrow V_R \tag{2.2}$$

(2.1) ifadesinde V(t), nöronun membran gerilimini, C nöronun membran kapasitansını, I(t) nöronun sinapslarına ulaşan atımların yarattığı nöron giriş akımını temsil etmektedir. IF nöron modelinde membran gerilimi belirli bir eşik seviyesine ulaşıncaya kadar sinaptik akım terimi I(t)'nin integrasyonu ile ifade edilmektedir. (2.2) ifadesinde V_{th} nöronun membran geriliminin eşik değerini belirtirken, V_R nöronun membran geriliminin ateşlemeden sonra aldığı değeri göstermektedir. IF nöron modeli basit yapısı nedeniyle genellikle ANN ağlarının SNN ağlarına dönüştürülmesinde kullanılmaktadır.

2.3.2 Leaky-Integrate-and-Fire Nöron Modeli

Leaky-Integrate-and-Fire modeli basitçe IF nöron modelinin biyolojik sinir ağlarının davranışına daha uygun hale getirilmesi için membran voltajının zamanla değişimini de içeren halidir. Matematiksel ifadesi (2.3) ve reset koşulu (2.4) ifadelerinde görülebilir.

$$\tau_m \frac{dV(t)}{dt} = -V(t) + RI(t) \tag{2.3}$$

$$V(t) \geq V_{th} \ , \ V \leftarrow V_R \tag{2.4}$$

(2.3) ifadesinde τ_m nöronun membran zaman sabitini, V(t) nöronun t anındaki membran gerilimini, R nöronun membran direncini, I(t) ise sinapslardan gelen giriş akımını belirtmektedir. (2.4) ifadesinde V_{th} nöronun membran geriliminin eşik değerini belirtirken, V_R nöronun membran geriliminin ateşlemeden sonra aldığı değeri göstermektedir. LiF nöron modeli, membran voltajının zamana bağlı olarak da değişmesi nedeniyle biyolojik nöronların davranışlarını IF nöronuna kıyasla daha iyi taklit etmektedir.

2.3.3 Hodgkin-Huxley Nöron Modeli

Hodgkin-Huxley [8] nöron modeli, biyolojik nöronlardakli iyonik akımlar ve membran gerilimi arasındaki ilişkiyi temel alan bir nöron modelidir. Hodgkin ve Huxley, denklem setini özellikle dev kalamar sinir lifleri üzerinde yaptıkları deneysel çalışmalar temelinde geliştirmişlerdir. Hodgkin-Huxley modeli (2.5), (2.6), (2.7) ve (2.8) ile verilen dört adet diferansiyel denklemle ifade edilmektedir.

$$I_C = C_m \frac{dV_m}{dt} + \bar{g}_K n^4 (V_m - V_K) + \bar{g}_{Na} m^3 h (V_m - V_{Na}) + \bar{g}_l (V_m - V_l) \tag{2.5}$$

$$\frac{dn}{dt} = \alpha_n(V_m)(1-n) - \beta_n(V_m)n \tag{2.6}$$

$$\frac{dm}{dt} = \alpha_m(V_m)(1-m) - \beta_m(V_m)m \tag{2.7}$$

$$\frac{dh}{dt} = \alpha_h(V_m)(1-h) - \beta_h(V_m)h \tag{2.8}$$

(2.5) ifadesi membran gerilimi V_m ile nöronun iyonik kanallarının giriş akımları arasındaki ilişkidir. (2.6) ifadesi potasyum kanalının açılma olasılığının denklemidir. (2.7) ve (2.8) ifadeleri ise sırasıyla sodyum kanalının açılma ve kapanma olasılığını vermektedir.

2.3.4 Izhikevich Nöron Modeli

Izhikevich nöron modeli [9] karmaşık nöronal aktiviteyi matematiksel olarak tanımlamak için geliştirilmiş bir nöron modelidir. Bu model 2003 yılında Eugene M. Izhikevich tarafından ortaya atılmıştır. Izhikevich nöron modeli Hodgkin-Huxley nöron modelinin karmaşıklığına göre daha basit bir matematiksel ifade sunması bakımından biyolojik nöronların modellenmesinde tercih edilmektedir. Izhikevich nöron modelinin matematiksel ifadesi (2.9) ve (2.10)'de, ateşleme koşulu (2.11)'de verilmiştir.

$$\frac{\mathrm{dV(t)}}{\mathrm{dt}} = 0.04V^2(\mathrm{t}) + 5\mathrm{V(t)} + 140 - \mathrm{U(t)} + \mathrm{I(t)} \tag{2.9}$$

$$\frac{\mathrm{dU(t)}}{\mathrm{dt}} = \mathrm{a(bV(t) - U(t))} \tag{2.10}$$

$$\mathrm{V(t)} \geq V_{th}, \quad V \leftarrow V_R, \qquad u \leftarrow u + d \tag{2.11}$$

(2.9) ve (2.10) ifadelerinde V(t) nöronun membran gerilimini, U(t) nöronun membran geriliminin toparlanma oranını, I(t) giriş akımını temsil etmektedir. a ve b boyutsuz sabitlerdir. a ve b sabitlerinin değerine göre Izhikevich nöronunun karakteristiği değişmektedir. (2.11) ifadesinde V_{th} nöronun membran geriliminin eşik değerini belirtirken, V_R nöronun membran geriliminin ateşlemeden sonra aldığı değeri göstermektedir. (2.11)'deki d sabiti ateşleme sonrası membran gerilimini düzenleyen sabit katsayıdır.

2.3.5 SRM Nöron Modeli

SRM nöron modeli [10] nöronun sinapslarına ulaşan atımlara verdiği yanıtları matematiksel olarak modellemeyi amaçlamaktadır. Bu model nöronun nasıl ateşlendiğini ve farklı giriş örüntülerine karşı nasıl cevaplar ortaya çıkardığını açıklamakta kullanılmaktadır. SRM nöronunun matematiksel modeli (2.12)'de, ateşleme koşulu (2.13) ve (2.14)'te, ateşleme koşulundaki değişim (2.15)'te incelenmektedir.

$$V(t) = \sum_{\mathrm{f}} \eta(\mathrm{t} - \mathrm{t^f}) + \int_0^{\infty} \kappa(\mathrm{s})\mathrm{I(t - s)ds} + \mathrm{V_{rest}} \tag{2.12}$$

$$V_{th}, \rightarrow V_{th}(t) \tag{2.13}$$

$$\mathrm{t} = \mathrm{t}^{\mathrm{f}}\ ,\ V(t)\ =\ V_{th}(t) \tag{2.14}$$

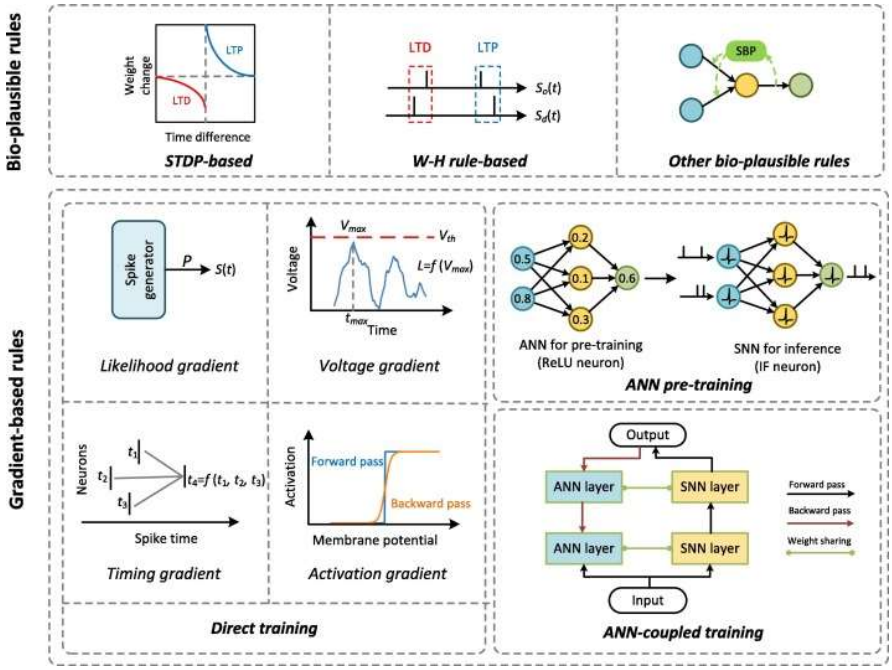
$$\frac{d[V(t) - V_{th}(t)]}{dt} > 0 \tag{2.15}$$

(2.12) ifadesinde t^{f} f. atımın zamanı, $\mathrm{V}_{\mathrm{rest}}$ nöronun girişinin sıfır olduğu durumdaki membran gerilimini, I(t-s) t-s anındaki nöron giriş akımını, κ(s) nöron giriş akımının membran gerilimine katkısını kontrol eden bir filtreyi temsil etmektedir. (2.13) ve (2.14) ifadelerindeki V_{th} eşik gerilimini temsil etmektedir.

SRM modelini diğer nöron modellerinden ayıran en önemli özelliği, eşik geriliminin zamana bağlı bir diferansiyel denklemle değişmesidir. SRM modeli, nöronların sinirsel aktivitesini daha ayrıntılı ve zamanla değişen girdilere duyarlı bir şekilde simüle etmek için oldukça kullanışlı bir nöron modelidir.

2.4 SNN Öğrenme Kuralları

SNN ağlarında öğrenme kuralları Biyolojik Öğrenme Kuralları ve Gradyan Tabanlı Öğrenme kuralları olarak iki sınıfta incelenmektedir. Şekil 2.1'de SNN öğrenme kurallarının sınıfları görsel olarak verilmiştir.



Şekil 2.1 Snn öğrenme kurallarının sınıfları [11]

2.4.1 Biyolojik Öğrenme Kuralları

Biyolojik öğrenme kuralları, biyolojik nöronların sinapslarının zamanla ve gelen uyartılarla değişimlerinin modellenerek SNN ağlarına uygulanması ile gerçekleştirilir. Biyolojik öğrenme kurallarında nöronların ağırlıkları nöronların ateşlenme zamanlamalarına göre güncellenmektedir. SNN ağlarında biyolojik öğrenme kuralları STDP tabanlı, Widrow-Hoff, ve SBP olarak üç alt sınıfta incelenir.

2.4.2 Gradyan Tabanlı Öğrenme Kuralları

Gradyan tabanlı öğrenme kuralları geleneksel yapay sinir ağlarında kullanılan gradyan tabanlı optimizasyon yöntemlerinin belirli sınırlar dahilinde SNN ağlarına uygulanmasıyla gerçekleştirilmektedir. Gradyan tabanlı SNN öğrenme kurallarında biyolojik öğrenme kurallarına ek olarak nöronun ateşlenme eşiği, varsa zaman sabiti, inert periyot süresi veya ateşlenme sonrası voltaj değeri gibi parametreler de eğitime tabi tutulabilir.

3
NÖROMORFİK DONANIMLAR

3.1 Nöromorfik Donanımlar

Nöromorfik donanımlar, insan beyninin işleyişinden ilham alarak tasarlanan, karmaşık hesaplama görevlerini gerçekleştirmeyi amaçlayan donanımlardır. Nöromorfik donanımlar geleneksel bilgisayar mimarisinden farklı olarak sinir ağlarına benzeyen yapılarıyla paralel işlem kabiliyeti sağlamaktadır.

Nöromorfik donanımlar çoğunlukla SNN odaklı tasarlanmaktadır. SNN odaklı çalışma büyük miktarda veriyi paralel işleme kabiliyetinin yanı sıra, SNN ağlarının atımlar yardımıyla haberleşmesinden ötürü enerji verimliliği açısından potansiyel olarak avantaj sağlamaktadır.

Nöromorfik donanımların geleneksel bilgisayar mimarilerine karşı paralel işlem kabiliyeti ve enerji verimliliği açısından avantajı bulunmasına rağmen bir takım dezavantajları da bulunmaktadır. Söz konusu sistemlerin tasarımı ve optimizasyonlarının zorluğu, yapay zeka problemlerinin çözümünde halihazırda kullanılan yazılım de donanım altyapılarına entegrasyonu çözülmesi gereken problemlerdir.

Nöromorfik donanımlar hali hazırda ticari olarak elde edilebilir değildir. Intel Loihi [12] ve Human Brain Project kapsamında geliştirilen SpiNNaker [13] prototipleri şu anda kullanılan en gelişmiş donanım mimarileri olmasına karşın, henüz nöromorfik donanımların ticari kullanımından ve genel bir mimariden bahsetmek mümkün değildir. Bu konuda çalışmalar halen devam etmektedir.

3.2 Nöromorfik Donanımların Temel Gereksinimleri

Nöromorfik hesaplamanın en önemli amacı nöronları gerçekleştiren sistemlerin ve alt bileşenlerinin, ağ mimarilerinin istenen hesaplamaları nasıl yarattığını, bilginin nasıl temsil edildiğini, hataya nasıl dayanıklı olduğunu, öğrenmesini, değişimlere nasıl uyum sağladığını (plasticity) ve evrimsel değişimleri incelemektir. Bu amaç ışığında herhangi bir nöromorfik sistemin gerçekleştirmesi gereken 5 temel gereksinimden söz edebilmek mümkündür.

Gereksinim 1: SNN ağları biyolojik nöronların matematiksel modelleri ile çalışmaktadır. Söz konusu matematiksel modeller zamana bağlı diferansiyel denklemlerle ifade edildiği için SNN ağlarını çalıştıracak bir dijital nöromorfik donanımın diferansiyel denklemleri çözebiliyor olması gerektiği gibi zamanın da takibini yapabiliyor olması gerekmektedir.

Gereksinim 2: Biyolojik nöronlar genellikle rastgele bağlantılar yaptığı için SNN ağlarını çalıştıracak bir nöromorfik donanımın sağlayabileceği sinaps sayısı herhangi bir nöron için herhangi bir bağlantı topolojisini destekleyecek sayıda olmalıdır.

Gereksinim 3: Biyolojik sinir ağlarında bilgiler sinapslar arasında impulslar yardımıyla nöronlar arasında transfer edilmektedir. Dolayısıyla bir nöromorfik donanımda da nöronlar arası bilgi transferi yine impulslar ile gerçekleştirilmelidir.

Gereksinim 4: Biyolojik sinir ağlarında öğrenme işlemi sürekli ve çalışma esnasında gerçekleştirilmektedir. Dolayısıyla bir nöromorfik donanım içerdiği her nöron için çalışma zamanında istenen öğrenme fonksiyonlarını gerçekleştirebilmelidir.

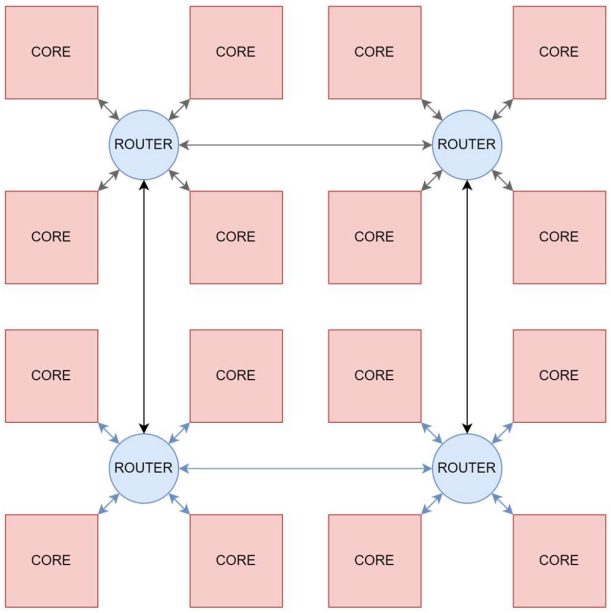
Gereksinim 5: Biyolojik nöronlar doğaları itibariyle olaya dayalı (event driven) çalışma prensibine sahiptir. Bir nöromorfik donanımın da bu özelliği takip ederek olay odaklı çalışması gerekmektedir.

3.3 Nöromorfik Donanım Mimarileri

Tez konusu olan donanım mimarisi VHDL donanım tanımlama dili ile ZYNQ-7000 SoC hedef platformu üzerinde tasarlanmıştır. ZYNQ-7000 mimarisinin blok diyagramı Şekil 4.1'de verilmiştir.

3.3.1 Network-on-a-Chip Mimarisi

NoC mimarisi paralel işlem kabiliyeti hedeflenen uygulamalarda sıklıkla kullanılan bir donanım tasarım yaklaşımıdır. Bu nedenle nöromorfik donanım tasarımlarında kullanılmaktadır. NoC mimarisinin genel blok diyagramı Şekil 3.1'de gösterilmektedir.



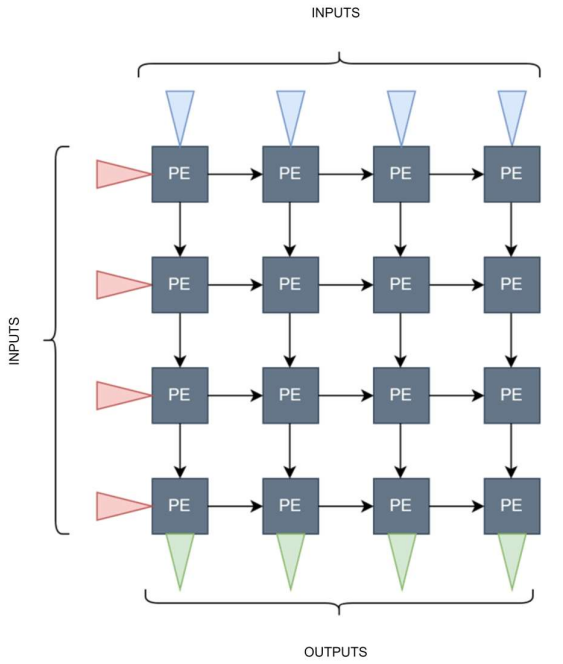
Şekil 3.1 Genel network-on-a-chip mimarisi

NoC mimarisinde gerçekleştirilecek işlemler her bir işlem ünitesi arasında paylaştırılır ve aradaki yönlendirici bağlantılar yardımıyla işlem birimleri hem birbirleriyle, hem de dış dünya ile haberleşmektedir.

NoC mimarisinin en büyük avantajı bir kere programlandıktan sonra bir daha programlamaya gerek duyulmadan sistemin çalışmasına olanak tanımasıdır. Bu nedenle olay tabanlı çalışmaya oldukça uygun bir yapıdadır. Bütün bu avantajlarına rağmen sinir ağının NoC mimarisine haritalanmasının zor olması, sistemin hafızasının çip üzerinde olması zorunluluğu nedeniyle alan gereksinimin artması ve mesaj tabanlı haberleşme nedeniyle donanımın bütün ağ için verilen bir t zaman aralığında bütün hesaplamaları bitirip bitiremeyeceğinin kesin olarak bilinmemesi dezavantajlarını da beraberinde getirmektedir.

3.3.2 Sistolik Diziler

Paralel işlem kabiliyetinin gerektiği uygulamalarda kullanılan bir diğer yaklaşım ise sistolik dizi mimarisidir. Sistolik diziler geleneksel yapay zeka uygulamalarında ve modern GPU ünitelerinde sıkça kullanılmaktadır. Sistolik dizilerin genel mimari blok diyagramı Şekil 3.2'de verilmiştir.



Şekil 3.2 Genel sistolik dizi mimarisi

Sistolik dizilerde her bir işlem elemanı girişlerinden aldığı verileri alıp, işleyerek işlem sonucunu komşu işlem birimlerine göndermek suretiyle hesaplama işlemlerini paralel hale getirmektedir. Her bir işlem birimi yapılacak hesaplamanın küçük bir parçasını gerçekleştirmekle sorumludur.

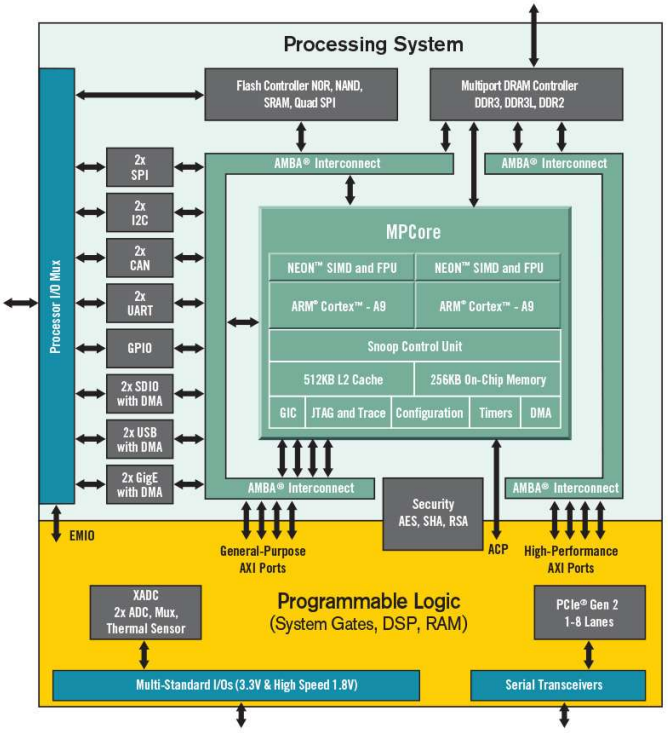
Sistolik dizinin avantajları, çok yüksek hızlara çıkabilmesi ve çalışma zamanının kesin olarak bilinmesi ve sinir ağını donanım üzerine haritalamanın oldukça kolay olmasıdır. Sistolik dizinin en büyük dezavantajı ise işlenecek sinir ağının sistolik dizinin kapasitesinden büyük olması durumunda ağın alt parçalara ayrılıp sisteme verilmesi ve her bir parça için sistolik dizinin tekrar programlanması gereksinimidir.

Tezin konusu olan Alanda Programlanabilir Nöron Dizisi Tasarımı'nda önerilen donanım mimarisi FPGA üzerinde daha kolay gerçeklenebilmesi ve tahmin edilebilir olması nedeniyle sistolik dizi olarak belirlenmiştir.

4
ÖNERİLEN DONANIM MİMARİSİ

4.1 ZYNQ-7000 SoC Hedef Platformu

Tez konusu olan donanım mimarisi VHDL donanım tanımlama dili ile ZYNQ-7000 SoC hedef platformu üzerinde tasarlanmıştır. ZYNQ-7000 mimarisinin blok diyagramı Şekil 4.1'de verilmiştir.



Şekil 4.1 Zynq-7000 mimarisi [14]

ZYNQ-7000 platformu, çift çekirdekli ARM Cortex-A9 işlemcisi ve Artix-7 FPGA donanımlarını aynı çip içinde bulundurmaktadır. Tezin konusu olan Alanda Programlanabilir Nöron Dizisi donanımı ve bütün alt sistemleri Artix-7 kaynakları kullanılarak tasarlanmıştır.

4.2 Alanda Programlanabilir Nöron Dizisi Donanım Gereksinimleri

SNN ağları, nöronları arasında rastgele ve seyrek bağlantılar bulundurabilirler. Buna ek olarak literatürde var olan nöron modellerinin ve öğrenme kurallarının zamana bağımlı olması nedeniyle SNN ağlarını çalıştırması hedeflenen bir nöromorfik donanımın hem rastgele bağlantılara izin vermesi, hem zamanın takibini yapabilmesi, hem de farklı nöron modellerini destekleyebilmesi beklenir.

Tez konusu olan donanımın tasarım öncesinde belirlenen temel gereksinimleri şöyledir.

1. Donanım farklı nöron tiplerini çalıştırabilmelidir.
2. Sinaptik plastisitenin sağlanabilmesi için her nöronun rastgele bağlantılar yapmasına imkan vermelidir.
3. Ağ davranışını gözlemlemek ve öğrenme kurallarını uygulayabilmek için zamanın takibini yapabilmelidir.
4. Her nöron için farklı öğrenme kuralı uygulamaya imkan tanımalıdır.

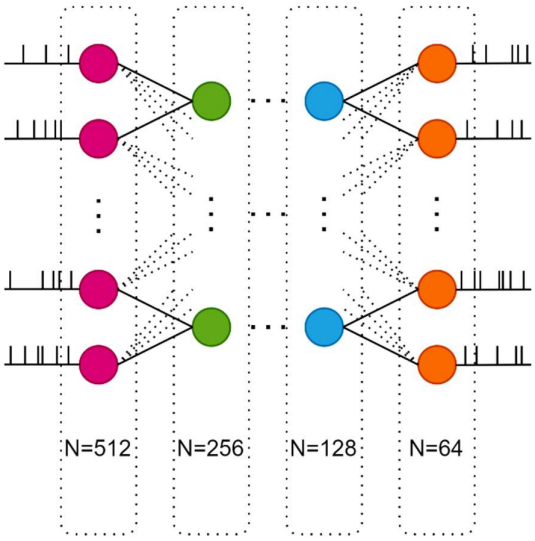
Donanımın alt modüllerinin tasarımında belirlenen 4 temel gereksinimin karşılanması hedeflenmiştir.

4.3 Tasarım Tercihleri ve Tasarım Parametreleri

Donanım tasarımında mimari olarak sistolik dizinin kullanılması sistolik diziyi programlamak ve sistolik dizinin bulundurduğu nöron sayısından daha fazla sayıda nöron içeren ağları küçük ağ parçalarına bölüp çalıştırabilmek için CPU ve DDR RAM kullanılmasını zorunlu kılmaktadır. Bu durum da nöron ağırlıklarının, nöron parametrelerinin ve öğrenme kurallarının veri tiplerini ve hafıza ihtiyaçlarını belirlemeyi gerektirmektedir.

4.3.1 Nöron Ağırlıklarının Temsili: INT8 Kuantalama Metodu

Yapay sinir ağlarında toplam parametre sayısı, ağın içerdiği nöron sayısı, ağın kaç katmandan oluştuğu ve katmanlar arasındaki bağlantı topoloji türüne bağlı olarak artış gösterme eğilimindedir. Katmanları arasında tam bağlantı bulunan 512 nöronluk bir giriş katmanına, 256 ve 128 nörondan oluşan iki adet gizli katmana ve 64 nörondan oluşan bir çıkış katmanına sahip atımlı sinir ağının temsili, toplam nöron sayısı ve ağırlıklarının sayısı Şekil 4.2'de gösterilmiştir.



Şekil 4.2 960 nörondan oluşan örnek atımlı sinir ağı

Her bağlantı topolojisine uyum sağlayabilmek için her zaman en kötü olasılık dikkate alınmalıdır. Görece küçük bir ağda bile sadece bağlantı tipinden ötürü ağırlıkların sayısının nöron sayısının 179.2 katına ulaşabilmesi, ağırlıkların hem DDR RAM'de hem de donanımın içinde fazla yer kaplamadan saklanabilmesi için ağırlıkların veri tipinin mümkün olan en ekonomik veri tipi olarak seçilmesini zorunlu kılmaktadır.

INT8 kuantalama yapay sinir ağları uygulamalarında hafıza kullanımını düşürmek için en sık kullanılan yöntemlerden biridir. INT8 kuantalama yönteminde 32-Bit kayan nokta aritmetiği formatı ile temsil edilen nöron ağırlıkları, ağırlıkların alabilecekleri minimum ve maksimum değerler üst sınır olmak şartıyla 8-Bit ile kuantalanarak INT8 formatına dönüştürülür.

INT8 kuantalamanın matematiksel temelleri sırasıyla (4.1), (4.2) ve (4.3) eşitlikleri ile ifade edilir.

$$R = \frac{2^8}{w_{max} - w_{min}} \tag{4.1}$$

$$Z = -\text{round}(R \times w_{min}) \tag{4.2}$$

$$W_{quant} = \text{round}(R \times W_{norm} + Z) \tag{4.3}$$

$$W_{norm} = \frac{W_{quant} - Z}{R} \tag{4.4}$$

(4.1)'deki R ifadesi kuantalama faktörünü belirtmektedir. (4.2)'deki Z ifadesi kuantalamanın sıfır noktasını belirlemektedir. (4.3)'deki W_{quant} ifadesi W_{norm} reel sayısının kuantalanmış halini ifade etmektedir. (4.4) ifadesinde W_{norm} sayısı kuantalanmış versiyonundan geri elde edilmektedir. Donanımda çalıştırılan tüm sinir ağlarında ağırlıklar reel sayı olarak [-1,1] aralığında değer almaktadır. INT8 kuantalama ile bu sayı aralığı [-128,127] değer aralığına haritalanmaktadır. Ağırlık çözünürlüğünün reel sayı cinsinden değeri (4.5) yardımıyla hesaplanabilir.

$$\text{Ağırlık Çözünürlüğü} = \frac{1 - (-1)}{255} = 0.00784313725 \tag{4.5}$$

Ağırlık çözünürlüğü reel sayı cinsinden donanımda temsil edilebilecek en düşük ağırlık değerine göstermektedir.

INT8 kuantalama metodu, 8-bit tamsayının 32-bitlik kayan nokta aritmetiği formatının ¼'ü kadar hafıza gerektirmesi ve 8-bitlik tamsayının donanım içinde çok basit lojik devrelerle işlenebilmesi sayesinde yüksek girişim hızı sağlamaktadır.

Ayrıca INT8 formatı birçok yüksek seviyeli programlama dili tarafından doğrudan desteklenmektedir. Sağladığı avantajlar nedeniyle INT8 veri tipi bu çalışmada ağırlıkları temsil etmek için seçilmiştir. Ağırlık vektörü tezin geri kalanında da SYNAPSE olarak adlandırılacaktır.

4.3.2 Zamanın Takibi

Zamanın takibi, sinapsların eğitimi işlemlerinde büyük önem arz etmenin yanı sıra istenildiği takdirde ağın davranışları ve zamana göre değişimleri gözlemleyebilmek açısından da önem teşkil etmektedir. Bu nedenlerden ötürü zamanın takibinin donanım tarafından gerçekleştirilmesi gereklidir.

Zaman takibi 8-bit uzunluğunda INT8 formatında TIMESTAMP adı verilen bir vektörle yapılmaktadır. Zaman takibinde TIMESTAMP vektörünün alabileceği

değer aralığı [0,127] kapalı aralığı olarak belirlenmiştir. Zaman vektörü ağırlık vektöründen farklı olarak asla negatif değer almamaktadır.

Zaman takibi TIMESTAMP vektörünün girişim işlemi sırasında güncellenmesiyle yapılmaktadır. Her ağırlık değeri, o ağırlıkta gerçekleşen en son atımın üstünden ne kadar zaman geçtiğinin bilgisini saklayan bir TIMESTAMP vektörü ile eşleştirilmiştir. TIMESTAMP vektörünün atım değerine göre güncellenmesi Tablo 4.1'de gösterilmiştir.

Tablo 4.1 Timestamp vektörünün atıma göre değişimi

Atım	$TIMESTAMP_{N+1}$
0	$TIMESTAMP_N + 1$
1	0

Tablo 4.1'de $TIMESTAMP_N$ ifadesi TIMESTAMP vektörünün o andaki değerini temsil ederken, $TIMESTAMP_{N+1}$ ifadesi TIMESTAMP vektörünün atım gerçekleşip gerçekleşmemesine göre sonraki zaman diliminde alacağı değeri göstermektedir. Atım gerçekleştiğinde TIMESTAMP vektörü 0'a eşitlenerek o zaman diliminde bir atım gerçekleştiğini belirtirken, atım gerçekleşmemesi durumunda TIMESTAMP vektörü 1 artırılarak o zaman diliminde o ağırlıkta herhangi bir atımın gerçekleşmediğini bildirir.

4.3.3 Birleşik Sinaptik Veri

Ağırlığı temsil eden SYNAPSE vektörü ve o ağırlıkta en son gerçekleşen atımın zaman bilgisinin tutulduğu TIMESTAMP vektörü 16-bit uzunluğunda tek bir vektör olarak donanımda işlenmektedir. Bu vektör donanımda NMODEL_SYNAPTIC_DATA olarak adlandırılmaktadır. 16 bitlik birleşik sinaptik verinin bit düzeni Tablo 4.2'de gösterilmiştir.

Tablo 4.2 Synaptic_data vektörünün bit düzeni

NMODEL_SYNAPTIC_DATA [15:0]	
SYNAPSE [15:8]	TIMESTAMP [7:0]

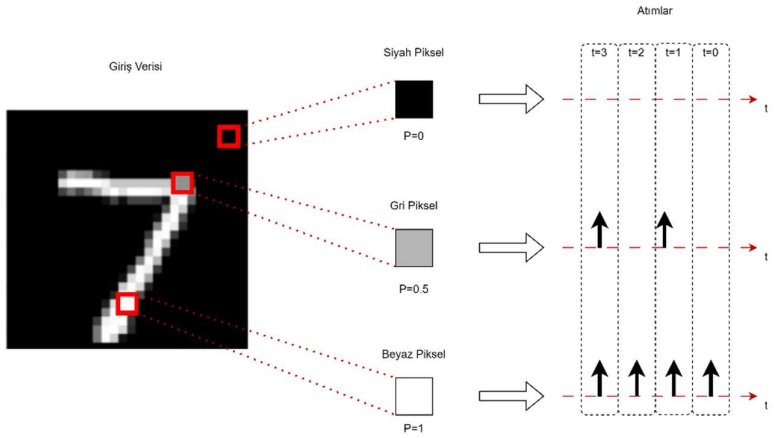
NMODEL_SYNAPTIC_DATA vektörünün ilk 8 biti ağırlık değeri olan SYNAPSE vektörü iken, ikinci 8 bitlik kısmı TIMESTAMP vektörüdür. Donanıma işlenmek üzere verilen birleşik sinaptik veriler işlendikten sonra donanımdan yine aynı formda geri elde edilirler.

4.3.4 Atım Kodlama Teknikleri

Resim piksellerinden gerçek zamanlı sinyallere kadar atımlar ile kodlanabilen her veri SNN ağları tarafından işlenebilmektedir.

SNN uygulamalarında ağın işleyeceği verilerin atımlara dönüştürülmesi için değişik kodlama teknikleri kullanılmaktadır. En çok kullanılan iki kodlama tekniği Oransal Kodlama (Rate Coding) ve Atım Gecikmesi Kodlama (Time-to-first-Spike Coding) tekniklerdir [15].

Oransal Kodlama tekniğinde veri, olasılıksal işlemler kullanılarak bir atım sekansına dönüştürülmektedir. Atım sekansının yoğunluğu verinin sayısal değeri ile değişim göstermektedir. Oransal Kodlamanın bir siyah beyaz resim ile gerçekleştirilen bir örneği Şekil 4.3'te gösterilmiştir.

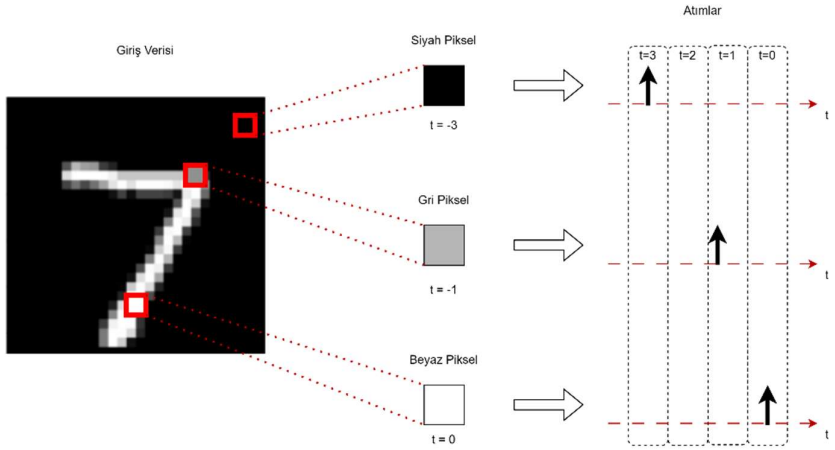


Şekil 4.3 Oransal kodlama örneği

Oransal Kodlama tekniğinde giriş verisi değerine göre atım gerçekleşme olasılığı belirlemektedir. Şekil 4.3'teki örnekte beyaz pikselin atım olasılığı %100, gri pikselin %50 ve siyah pikselin atım gerçekleştirme olasılığı 0 olarak kodlanmıştır.

Atım Gecikmesi Kodlama metodunda ise her bir veri sadece bir adet atımla kodlanmaktadır. Verinin değerine göre atımın zaman çerçevesinde bulunduğu yer değişmekte olup verilerin atım karşılıklarının farkı atımların nörona ilk ulaştığı

zamanın farkı ile belirtilmektedir. Şekil 4.3'teki resmin atım gecikmesi kodlama metodu ile atımlara dönüştürüldüğü örnek Şekil 4.4'te gösterilmiştir.



Şekil 4.4 Atım gecikmesi kodlama örneği

Şekil 4.4'te verilen atım gecikmesi kodlamasında ise her bir giriş verisi bir zaman referansından ne kadar sonra atım gönderileceği bilgisi ile kodlanmıştır. Örnekte beyaz piksel, aynı zamanda referans olan, t=0 anında gönderilen atımla kodlanırken, gri piksel t=-1, siyah piksel t=-3 anında atım gönderilmesi ile kodlanmıştır.

Atımların hangi teknikle kodlanacağı uygulamadan uygulamaya değişiklik göstermektedir. Oransal Kodlama atım gecikmesi kodlama yöntemine kıyasla daha yüksek başarım sağlamaktadır ancak daha fazla atım gönderilmesi nedeniyle girişim sayısını artırmasından ötürü daha çok güç tüketimine yol açmaktadır. Atım Gecikmesi Kodlama metodu girişim sayısının az olması nedeniyle daha az güç tüketmektedir. Ancak ağın istenen başarım ölçütlerine yakınsaması için ağ topolojisinde, nöron modellerinde ve öğrenme kurallarında daha hassas ayarlamalar yapılmasını gerektirmektedir.

Donanım için atımların nasıl kodlandığından ziyade atımların varlığı ya da yokluğu anlam ifade etmektedir. Dolayısıyla tasarlanan donanım, atımların ikilik sistemde 1-bit ile ifade edilmesi şartıyla bütün atım kodlama tekniklerine karşı tamamen agnostiktir.

4.3.5 Nöron Modellerinin Temsili

SNN ağlarında kullanılan nöron modelleri, biyoloji temelli olmalarından ötürü ANN ağlarında kullanılan aktivasyon fonksiyonlarına kıyasla daha karmaşık hesaplamalar gerektirmektedir.

Zaman terimini içeren basit diferansiyel denklemlerin varlığı nöron modellerinin hesaplanmasında numerik metodların kullanılmasını zorunlu kılmaktadır. Her girişim işleminden sonra numerik metodlarla belirlenen zaman aralığı için nöron durumları iteratif olarak hesaplanmaktadır.

Tez konusu olan donanımda sabit bir nöron modeli yerine, her nöron modelini çalıştırabilecek yeterlikle olan, kendi komut setine sahip ve mikro kod ile programlanabilen bir nöron emülatörü kullanılmaktadır.

Nöron modeli çalıştırılırken nörona ait üç farklı parametre cinsi bütün nöronlar için her zaman adımında hesaplanmaktadır. Hesaplanan parametreler aşağıda verilmiştir.

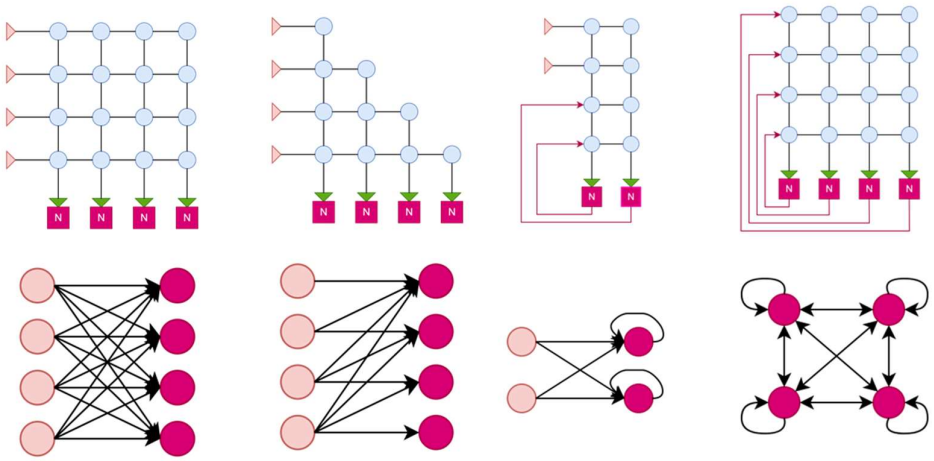
1. **Nöron Durumları:** Nöron durumları genellikle bir nöronun membran potansiyeli, eşik potansiyeli, membran gerilimi sönümlenme hızı gibi nöron modeline özgü, nöronun davranışlarını belirleyen parametrelerdir. Her nöron için her zaman adımında nöron durumları hesaplanmaktadır. Nöron durumları donanımda 16-bit yarı-hassas kayan nokta aritmetiği sayı formatı ile temsil edilmektedir. Yarı hassas kayan nokta aritmetiği formatı tezin geri kalanında FP16 kısaltması kullanılarak ifade edilecektir.
2. **Nöron Son Ateşleme Zamanı:** Nöron son ateşleme zamanı her nöron için her zaman adımında hesaplanmaktadır. Sinapsların ateşlenme zamanı ile tam uyum sağlaması için nöron son ateşleme zamanı 8-bit ile ifade edilir ve [0,127] kapalı aralığında değer alır. Nöron ateşlendiği zaman son ateşleme zamanı bilgisi sıfırlanır, nöron ateşlenmezse son ateşleme bilgisi 1 artırılır. Son ateşleme zamanı hesaplanması zorunlu bir parametredir. Donanımda NMODEL_SPIKE_TIME adıyla ifade edilmekte olup tezin geri kalanında da bu şekilde ifade edilecektir.
3. **İnert Periyot:** İnert Periyot İngilizce “Refractory Period” terimine karşılık gelmektedir. İnert periyot nöronun ateşlendikten sonra belirli bir süre boyunca uyarıya karşı duyarsız olduğu zaman dilimidir. İnert periyot her

nöron için 8-bit ile ifade edilir. Nöron ateşlendikten sonra önceden belirlenen inert periyot değeri ilgili birime iletilir ve inert periyot değeri her zaman adımında 0 değerine ulaşıncaya kadar 1 azaltılır. İnert periyot aktif olduğu sürece nöron girişim sonucunu dikkate almadan durum parametrelerini hesaplamaktadır. Nöron modelinin inert periyot bulundurması tamamen opsiyoneldir.

4.3.6 Sinaptik Plastisite

Farklı bağlantı türlerine uyum sağlama yeteneği donanımın değişik problemlerin çözümünde kullanılmasına imkan vermektedir. Programlanabilir nöron modeli ve nöronlar arasında farklı türlerde bağlantı yapabilme yeteneği tasarlanan donanımın yeniden kullanılabilirliğini önemli ölçüde artırmaktadır.

Sistolik diziler yapıları nedeniyle yüksek paralel işlem kabiliyeti sunmasının yanında, her bir işlem elemanının ayrı ayrı devreden çıkarılabilir yapıda tasarlanması halinde farklı bağlantı türlerine kolayca uyum sağlayabilmektedir. 4x4 boyutunda bir sistolik dizi yapısının farklı bağlantı türlerine nasıl uyum sağladığı Şekil 4.5'te gösterilmiştir.



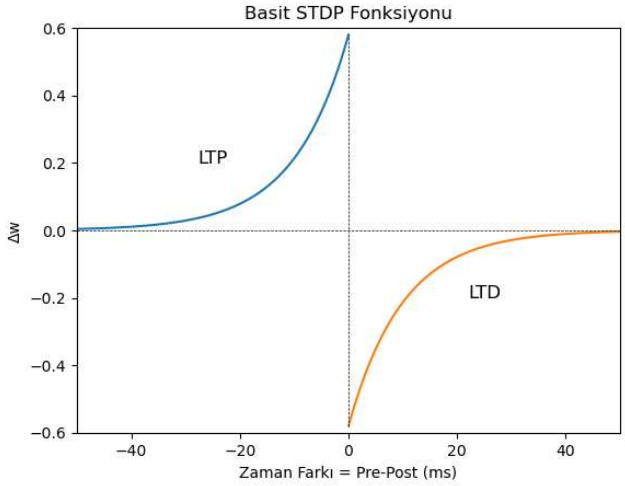
Şekil 4.5 4x4'lük bir sistolik dizi üstünde farklı sinir ağı bağlantı şemaları

Sistolik dizi mimarisi versatil yapısı ve paralel işlem kabiliyeti nedeniyle tezin konusu olan donanımın ana iskeleti olarak seçilmiştir. Tasarlanan ve 5. Bölüm'de tanıtılacak olan tüm alt modüller bir araya gelerek bir sistolik dizi mimarisini oluşturmaktadır.

4.3.7 Öğrenme Kuralları ve Parametreleri

Öğrenme işlemi, girişimden sonra en fazla hesaplama gücünün gerektiği işlemdir. Bölüm 4.3.1'de gösterildiği üzere görece küçük bir ağın bağlantı tipine göre 100.000'den fazla ağırlık içermesi ağırlıkların öğrenme işleminde paralel biçimde işlenmesini çalışma zamanının kısaltılması için gerekli kılmaktadır.

Donanım sadece STDP [16] temelli öğrenme kurallarını desteklemektedir. STDP (Spike-Timing Dependent Plasticity) metodu nöron ağırlıklarını pre-sinaptik nöronlardan post-sinaptik nöronlara ulaşan atımların zaman farkını dikkate alarak nöronlar arası bağlantıları güçlendirmek veya zayıflatmak suretiyle uyarlayan denetimsiz bir öğrenme biçimidir. Basit bir STDP metoduna ait görsel Şekil 4.6'da gösterilmiştir.



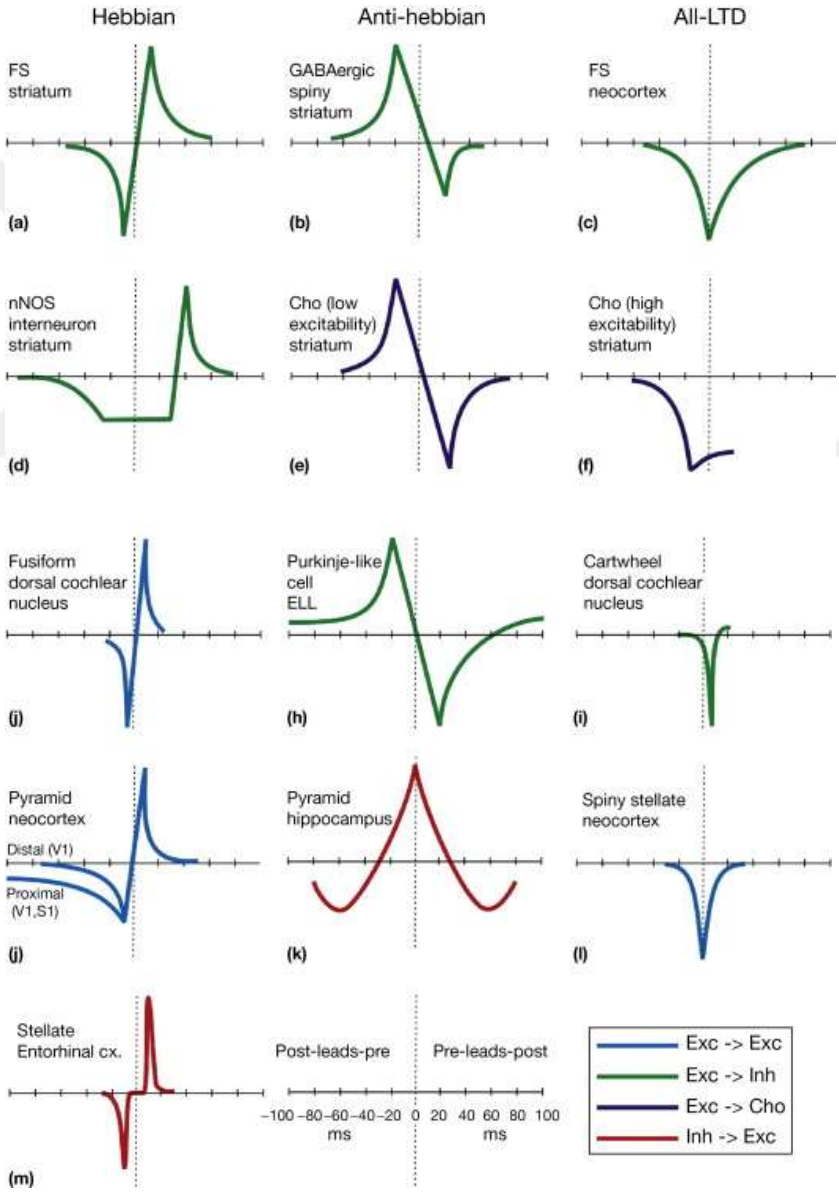
Şekil 4.6 Basit stdp fonksiyonu

Şekil 4.6'da görülen basit STDP fonksiyonunun zaman penceresi [-50, +50] ms aralığı olarak seçilmiştir. Fonksiyonun LTP ile isimlendirilen eğrisi pre-sinaptik nöronun post-sinaptik nörondan daha önce ateşlenmesi durumunda, LTD ile isimlendirilen eğrisi ise post-sinaptik nöronun pre-sinaptik nörondan daha önce ateşlenmesi durumunda kullanılmaktadır. LTP eğrisi iki nöron arasındaki ağırlığı artırma eğiliminde iken, LTD eğrisi iki nöron arasındaki ağırlığı düşürme eğilimindedir. Şekildeki grafiğin genel matematiksel ifadesi (4.6) ile hesaplanır.

$$\Delta w = \begin{cases} Ae^{t/\tau} & , \quad t < 0 \\ -Ae^{-t/\tau} & , \quad t > 0 \end{cases} \tag{4.6}$$

(4.6)'daki Δw ağırlıktaki değişimi göstermektedir, t pre-sinaptik ve post-sinaptik nöronların ateşlenmesi arasındaki zaman farkını temsil ederken τ öğrenme fonksiyonunun zaman sabitini göstermektedir. A ise öğrenme fonksiyonunun ölçeklenmesi için kullanılan pozitif sabit bir katsayıdır.

Bütün STDP fonksiyonları simetrik olmak zorunda değildir. STDP fonksiyonları temel olarak üç sınıfa ayrılmaktadırlar. STDP metodunun üç ayrı sınıfı Şekil 4.7'de gösterilmiştir.



Şekil 4.7 Stdp öğrenme metodunun üç farklı sınıfı [16]

Tasarlanan donanım STDP temelli öğrenme kurallarını uygulayan ve sistolik dizinin işlem elemanlarına entegre edilmiş paralel çalışan öğrenme motorları bulundurmaktadır. Öğrenme motorları STDP fonksiyonlarının 256 noktada örneklenmiş hallerini INT8 formatında bir tablo halinde bulundurup, 4.3.3'de Birleşik Sinaptik Veri bölümünde gösterilen TIMESTAMP vektörü ve 4.3.5. Nöron Durumlarının Temsili bölümünde söz edilen nöronun son ateşlenme zamanını sırasıyla pre-sinaptik ve post-sinaptik nöron ateşlenme zamanları olarak alıp birbirinden çıkararak STDP tablosunda ilgili Δw değerinin adresini elde etmektedir. Daha sonra Δw değeri ile (4.7) ve (4.8)'deki eşitliklere göre ağırlık değerini modifiye etmektedir.

$$w_{n+1} = w_n - N\Delta w(w_n - w_{max}) \quad , \quad \Delta w > 0 \tag{4.7}$$

$$w_{n+1} = w_n + N\Delta w(w_n - w_{min}) \quad , \quad \Delta w < 0 \tag{4.8}$$

(4.7) ve (4.8) eşitliklerindeki w_n ifadesi sinaptik ağırlığın önceki değerini belirtmektedir. w_{n+1} ifadesi ağırlığın öğrenme işlemi sonrasında aldığı değeri işaret etmektedir. N değeri öğrenme oranını temsil ederken, w_{min}ve w_{max} sinaptik ağırlığın alt ve üst sınırlarını temsil etmektedir.

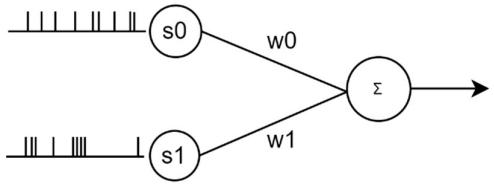
(4.7) ve (4.8) eşitlikleri donanım içerisinde yine INT8 kuantalama formatında sonuç verecek şekilde yapılmaktadır. Dolayısıyla donanımda kuantama farkındalığı ile eğitim işlemi uygulanmaktadır.

5
DONANIMIN TEMEL YAPI TAŞLARI ve YETENEKLERİ

5.1 Atım-Ağırlık Girişimi ve Zaman Takibi

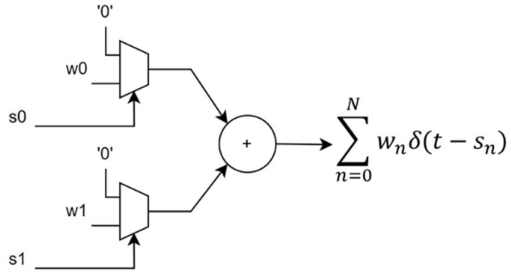
5.1.1 Basit Girişim Devresi

Atımlar ve sinaptik ağırlıkların girişimi herhangi bir SNN ağının çalıştırılmasının ilk adımıdır. Bu adımda ikilik tabanda sisteme verilen atımların 1 veya 0 olması durumunda atımların gerçekleştiği sinapsların ağırlık değerlerinin çıkıştaki toplamda yarattığı değişim hesaplanmaktadır. İki adet sinapsa sahip olan bir nöronun girişim modelinin temsili Şekil 5.1'de gösterilmiştir.



Şekil 5.1 İki sinapsa sahip bir nöronun girişim modeli

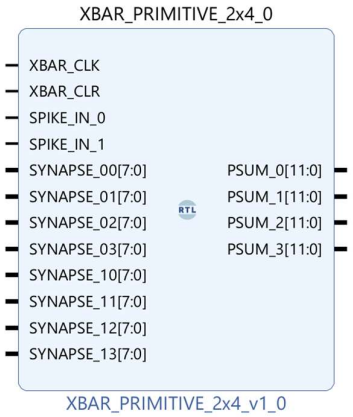
Donanımda atımlar ikilik sistemde 1-bit ile temsil edildikleri için ağırlıkların atımın değerine çıkıştaki toplama yansıtılması seçici ve toplayıcı devreler kullanarak kaynak kullanımı açısından ekonomik biçimde gerçekleştirilmektedir. Söz konusu basit girişim devresi Şekil 5.2'de gösterilmiştir.



Şekil 5.2 İki sinapsa sahip nöronun basit girişim devresi

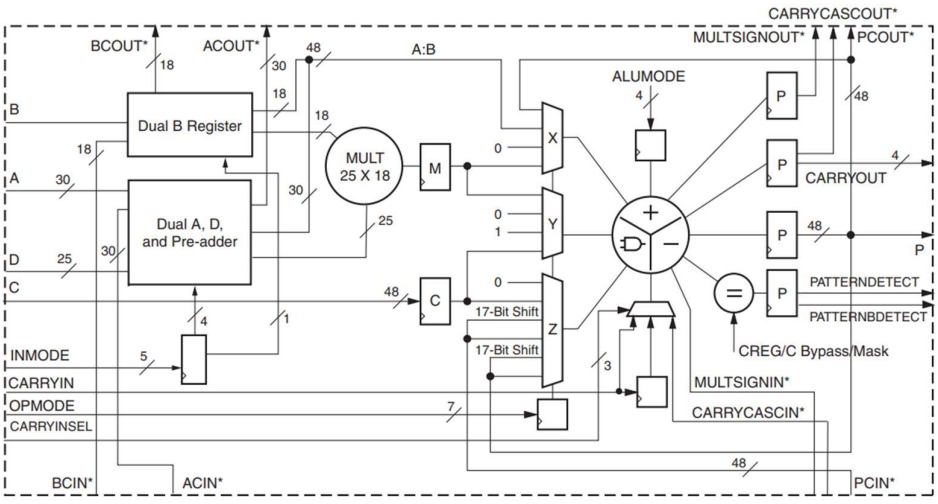
5.1.2 XBAR Ünitesi ve Çalışma Prensibi

XBAR ünitesi atım-ağırlık girişimini gerçekleştiren ünitedir. XBAR ünitesi, her biri 2 adet sinapsa sahip 4 nöronun atım-ağırlık girişimi işlemini gerçekleştirmektedir. XBAR ünitesinin giriş-çıkışlarını gösteren RTL modül hali Şekil 5.3'te verilmiştir.



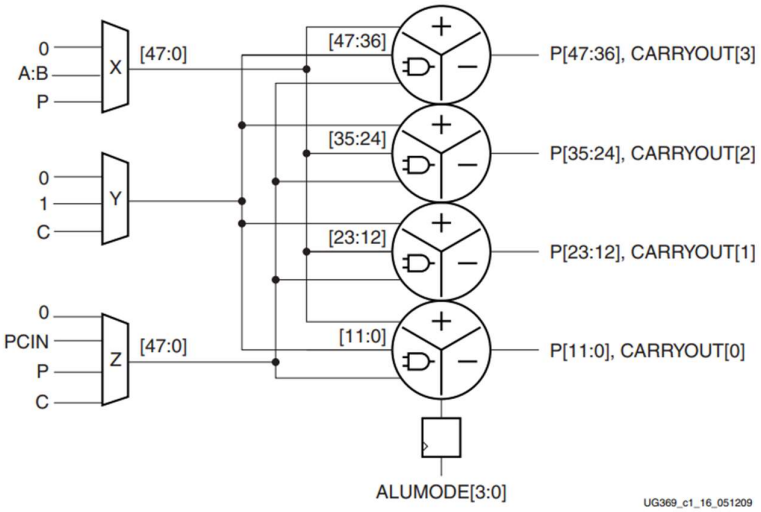
Şekil 5.3 Xbar ünitesi rtl modül görüntüsü

FPGA donanımları matematiksel işlemleri hızlandırmak ve bu işlemlerin kaynak tüketimlerini minimum seviyede tutabilmek için DSP üniteleri barındırmaktadır. Zynq-7000 SoC içindeki DSP ünitesinin blok diyagramı Şekil 5.4'te verilmiştir.



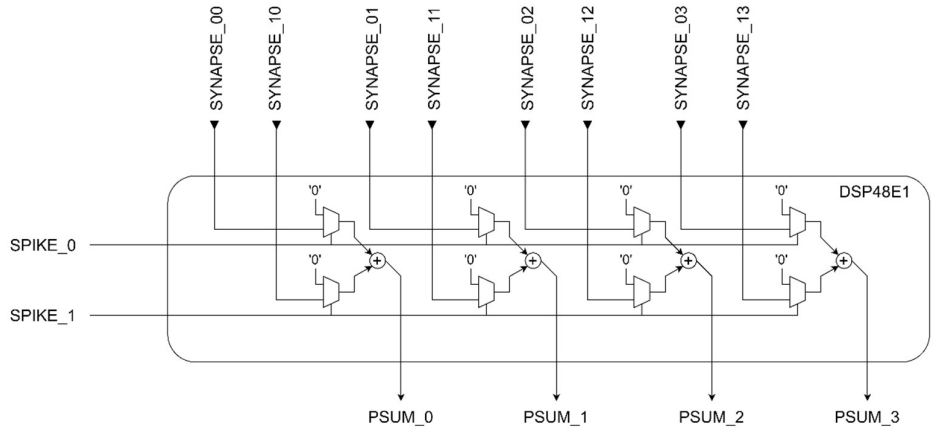
Şekil 5.4 Dsp blok diyagramı [17]

DSP üniteleri sentezleyiciler tarafından aritmetik işlemlerde otomatik olarak kullanılmaktadır. DSP üniteleri istenildiği takdirde komponent olarak HDL koduna eklenebilmektedir. DSP ünitelerinin SIMD modunda kullanılması ile 48-Bitlik bir adet DSP ünitesi 4 adet 12-Bit toplayıcı devreye bölünmektedir. DSP ünitesinin SIMD modunda kullanımı Şekil 5.5'te gösterilmiştir.



Şekil 5.5 Dsp simd modu [17]

XBAR ünitesi 1 adet DSP modülünün SIMD modunda kullanılması ile donanımda gerçekleştirilmiştir. XBAR ünitesinin fonksiyonları DSP modülünün OPMODE [5:0] girişi ile kontrol edilmektedir. OPMODE [5:4] ve OPMODE [1:0] girişleri girişlerdeki W, X,Y ve Z seçicilerini kontrol etmektedir. Şekil 5.4'te seçicilerin sıfırıncı girişler donanımda topraklandığı için giriş 0 olduğu sürece çıkışa 0 değerini vermektedir. Giriş 1 olduğunda ise A, B ve C girişleri toplayıcılara gönderilmektedir. Bu yöntem ile 4 adet basit girişim devresi sadece 1 adet DSP modülü kullanarak donanımda gerçekleştirilmektedir. DSP SIMD modunun 4 adet basit girişim devresi olarak modellenmesine ilişkin temsil Şekil 5.6'da verilmiştir.



Şekil 5.6 Xbar modülünün dsp48e1 simd moduyla oluşturulması

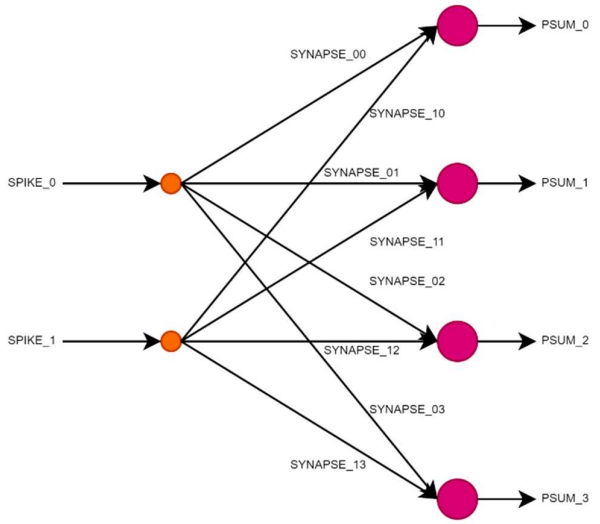
Devrenin giriş ve çıkışları arasındaki ilişki Tablo 5.1'de verilmiştir.

Tablo 5.1 Xbar psum çıkışının spike girişlerine göre değişimi

SPIKE_0	SPIKE_1	PSUM_x
0	0	0
0	1	SYNAPSE_1x
1	0	SYNAPSE_0x
1	1	SYNAPSE_0x + SYNAPSE_1x

Toplayıcı giriş ve çıkışları SIMD modunda 12-bit uzunluğundadır. Girişim devresinde 8-bit ağırlık değerleri donanımda 12-bite genişleterek işleme alınmaktadır.

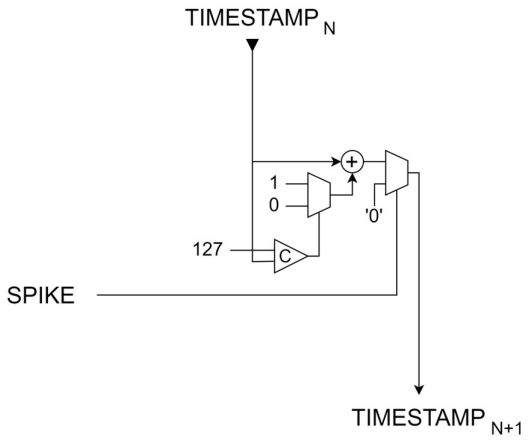
XBAR ünitesinin iki atım girişi, içinde barındırdığı dört adet basit girişim devresi tarafından paylaşılmaktadır. Şekil 5.6'daki aynı iki atım girişini paylaşan devrenin sinir ağı karşılığı Şekil 5.7'de verilmiştir.



Şekil 5.7 Xbar sinir ağı girişim eşdeğeri

5.1.3 Zaman Takip ve Güncelleme Sistemi

Öğrenme kurallarını uygulayabilmek ve ağın zaman uzayında davranışını gözlemlemek için zaman bilgisinin saklanması ve sürekli güncellenmesi gerekmektedir. Donanımda zaman bilgisi 4.3.1'de ifade edildiği gibi 8-Bit uzunluğunda olup [0,127] kapalı aralığında değer almaktadır. Zaman bilgisinin takip edilmesi ve atım gerçekleşip gerçekleşmemesine göre güncellenmesi Şekil 5.8'de verilen basit devre yardımıyla gerçekleştirilmektedir.



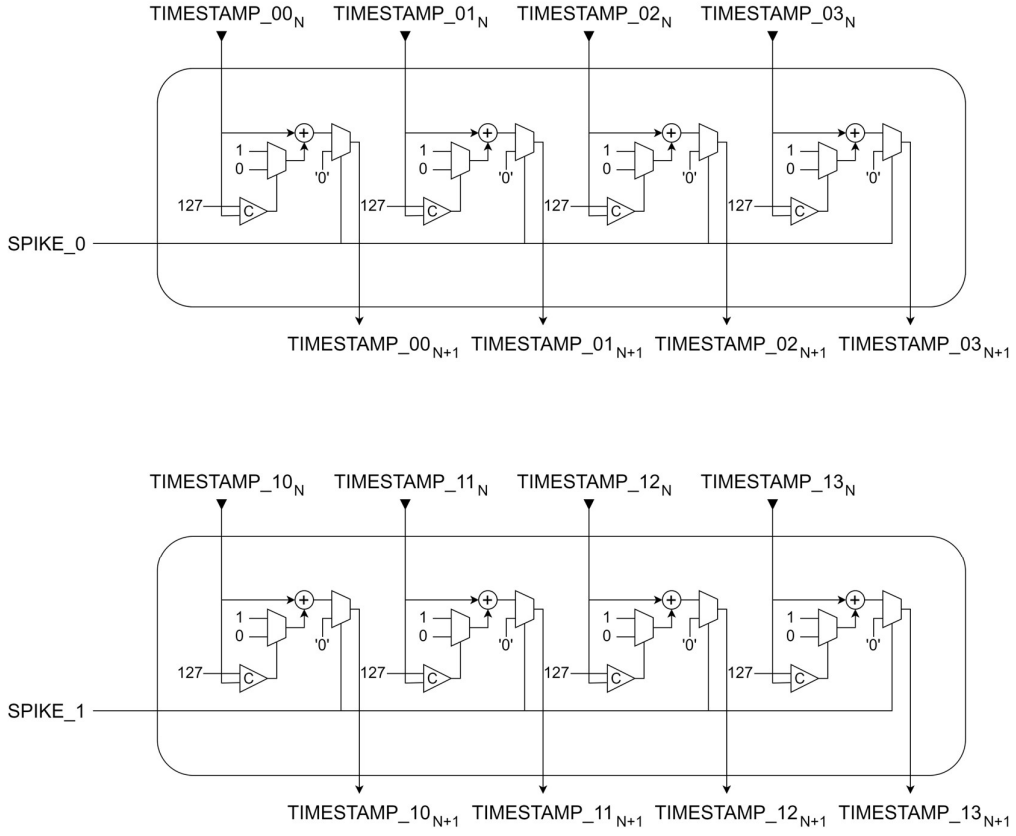
Şekil 5.8 Bir timestamp vektörü için atım zamanı takip ve güncelleme devresi

Zaman bilgisinin güncellenmesi (5.1)'de matematiksel olarak ifade edilmiştir.

$$\mathrm{TIMESTAMP}_{n+1} = \begin{cases} \mathrm{TIMESTAMP}_n + 1\,, & \mathrm{SPIKE} = '0' \\ 0\,, & \mathrm{SPIKE} = '1' \end{cases} \quad (5.1)$$

Herhangi bir zaman diliminde sinapsta bir atım oluşması halinde zaman bilgisi sıfırlanmaktadır. Atım gerçekleşmemesi halinde zaman bilgisi zaman bilgisinin üst sınırına kadar her zaman diliminde 1 artırılmaktadır.

XBAR ünitesi aynı anda 4 farklı nöronun 2 adet sinapsını girişim işlemine tabi tutmaktadır. Dolayısıyla XBAR ünitesi ile paralel çalışacak olan zaman takip sistemi de Şekil 5.8'de verilen basit zaman takip devresinden her SPIKE girişi için 4 adet zaman takip devresi bulunduran bir sistem olarak XBAR ünitesine entegre edilmiştir. Zaman takip sisteminin devre şeması Şekil 5.9'da verilmiştir.



Şekil 5.9 Zaman takip sistemi

5.2 Öğrenme Kuralının Ağırlıklara Uygulanması

5.2.1 Olay Algılama

Öğrenme işlemi sinapsta bir atım gerçekleştiğinde ve/veya nöron ateşlendiğinde uygulanmaktadır. Olay algılama işlemi sinaptik vektördeki zaman bilgisi ve nöronun son ateşlenme bilgisi kontrol edilerek yapılmaktadır. Nöronun son ateşleme bilgisi veya sinapsta gerçekleşen son ateşlemenin zaman bilgisinden herhangi birinin 0 olması durumunda o zaman aralığında olay gerçekleştiği tespit edilir ve ilgili ağırlık değeri öğrenme tablosuna göre güncellenir. Olay türleri ve olay algılama işlemi Tablo 5.2'de gösterilmiştir.

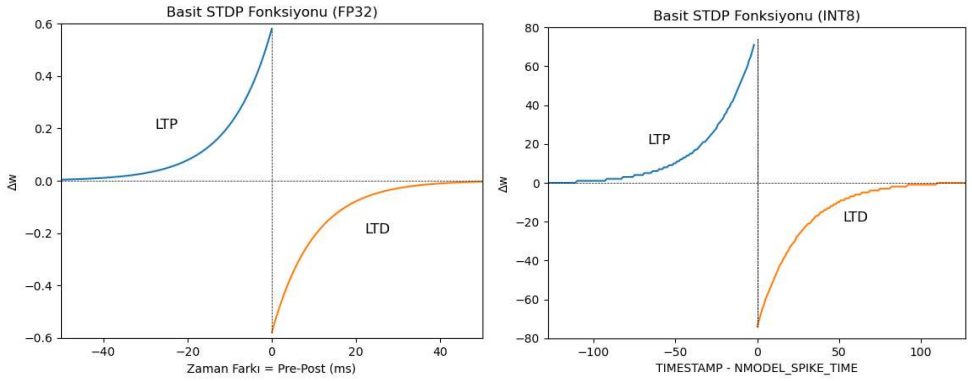
Tablo 5.2 Zaman bilgilerinin değerlerine göre olay tipleri ve alınan aksiyon

TIMESTAMP	NMODEL_SPIKE_TIME	Olay Tipi	Güncelleme
>0	>0	Yok	Hayır
0	>0	Pre-Sinaptik	Evet
>0	0	Post-Sinaptik	Evet
0	0	Eş Zamanlı	Evet

Pre-sinaptik olay türü sadece ilgili sinapsta bir atımın gerçekleşmesi durumudur. Post-sinaptik olay sadece nöronun ateşlenmesi durumu iken eş zamanlı olay türünde hem sinapsta atım gerçekleşmekte olup hem de nöron ateşlenmektedir. Pre-sinaptik olayda sadece ilgili ağırlık değeri güncellenirken, post-sinaptik olayda nöronun bütün ağırlık değerleri güncellenmektedir.

5.2.2 Öğrenme Tablosu ve Öğrenme Parametreleri

Donanım sadece STDP tabanlı öğrenme fonksiyonlarını desteklemektedir. STDP fonksiyonunun 256 noktada örneklenmiş ve INT8 formatında kuantalanmış hali bir tablo haline getirilip donanımda bulunan öğrenme modüllerine verilmektedir. Bölüm 4.3.7'de Şekil 4.6'da gösterilen basit STDP fonksiyonunun 256 noktada örneklenmiş ve INT8 formatında kuantalanmış halinin orjinal hali ile karşılaştırılması Şekil 5.10'da gösterilmektedir.



(a) (b)

Şekil 5.10 Basit stdp fonksiyonunun (a) orjinal eğrisi (b) donanımda kullanılan int8 formatındaki eğrisi

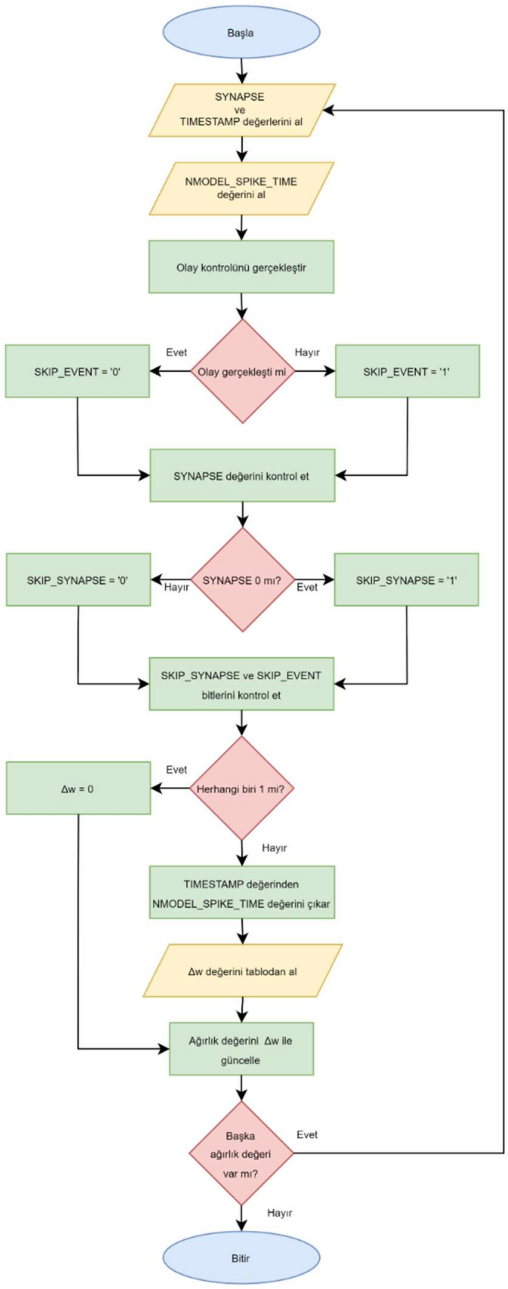
Zaman farkının çözünürlüğü nöron modelinin numerik yönteminde belirlenen adım aralığı ile eşit seçilmelidir. TIMESTAMP-NMODEL_SPIKE_TIME zaman farkı [-127,127] kapalı aralığında değer almaktadır. Numerik yöntemle belirlenen zaman adımı ile Şekil 5.10 (a)'da öğrenme penceresinin yatay ekseni olan Pre - Post arasındaki ilişki (5.2)'de verilmiştir.

$$\text{Pre} - \text{Post} = \text{Adım Aralığı x } [-127,127] \tag{5.2}$$

(5.2) eşitliğine göre adım aralığı 500 µs olarak belirlenen bir nöronun öğrenme penceresi -63.5 ms ve 63.5 ms aralığında olacaktır. Dolayısıyla öğrenme fonksiyonu örneklenirken -63.5 ms ve 63.5 ms arasında aldığı değerler tablo haline getirilecektir.

5.2.3 Öğrenme Algoritması

Donanımda öğrenme algoritması olay kontrolünün yapılmasından sonra olay gerçekleşmesi takdirinde sinapsta gerçekleşen son atımın zamanı olan TIMESTAMP ile nöronun son ateşlenme zamanını olan NMODEL_SPIKE_TIME farkı ile adreslenen öğrenme tablosu katsayısı Δw'nin nöron ağırlık değeri ile (4.7) ve (4.8) ile verilen işlemlere tabi tutulması ile yapılmaktadır. Öğrenme algoritmasının akış diyagramı Şekil 5.11'de verilmiştir.



Şekil 5.11 Öğrenme algoritmasının akış diyagramı

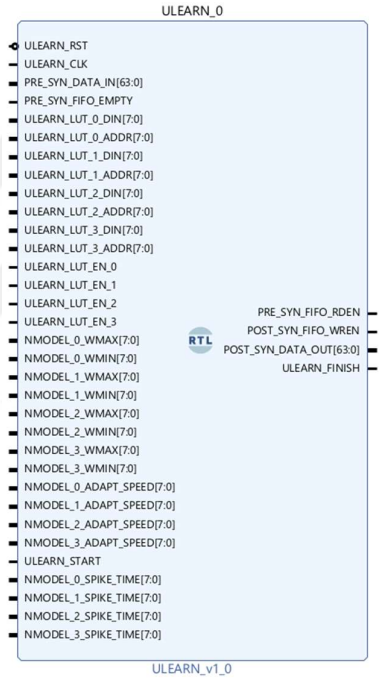
5.2.4 ULEARN Modülü

ULEARN modülü öğrenme işleminin gerçekleştirilmesinden sorumlu olan modüldür. ULEARN modülü 4 farklı nöron modelinin sinapslarını nöronların öğrenme tablolarına göre güncellemektedir. ULEARN modülü çalışırken Şekil 5.11'de akış diyagramı verilen algoritmayı kullanır.

Girişindeki 4 ayrı nöronun sinaps ve zaman bilgisinden oluşan 64-Bit uzunluğundaki veriyi 16 bitlik parçalara ayırarak her ağırlık için sıralı biçimde öğrenme fonksiyonlarını gerçekleştirmektedir.

ULEARN modülü dört farklı nöron için dört farklı öğrenme tablosunu hafızasında tutabilmektedir. Bütün nöronların öğrenme tablosunun aynı olması gibi bir kısıtlama yoktur.

ULEARN modülününün giriş çıkış portlarını gösteren RTL modül görüntüsü Şekil 5.12'de görülmektedir.



Şekil 5.12 Ulearn rtl modül görüntüsü

ULEARN giriş çıkışlarının işlevleri Tablo 5.3'te detaylı bir biçimde verilmiştir. Tabloda x işareti içeren portlar x=0,1,2,3 için aynı işlevi görmektedir.

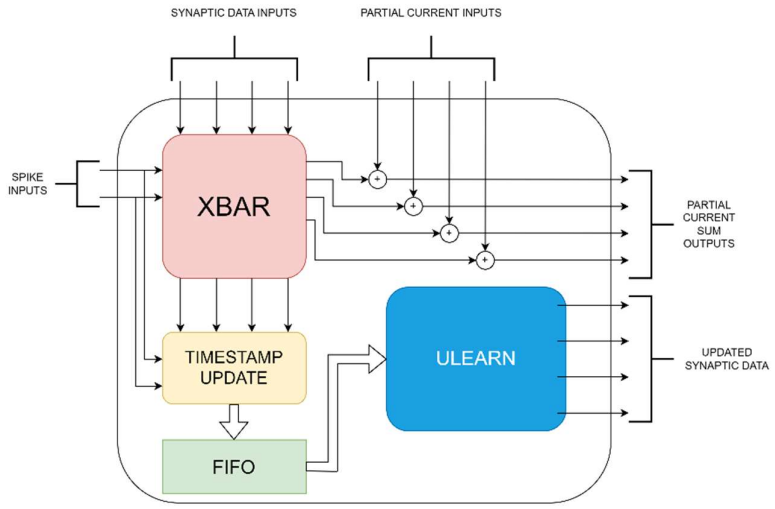
Tablo 5.3 Ulearn portları ve portların işlevleri

Port	İşlevi
ULEARN_CLK	Modülün saat girişidir.
ULEARN_RST	Modülün reset girişidir.
PRE_SYN_DATA_IN [63:0]	Eğitim işleminin uygulanacağı ağırlıkların alındığı giriştir.
PRE_SYN_FIFO_EMPTY	Ağırlıkların saklandığı FIFO'nun boş olup olmadığını bildirir.
ULEARN_LUT_x_DIN [7:0]	Nöronun öğrenme tablosunun verisidir.
ULEARN_LUT_x_ADDR [7:0]	Öğrenme tablosunda verinin yazılacağı adres bilgisidir.
ULEARN_LUT_EN_x	Öğrenme tablosunun girişini aktif eder ve tabloya veri yazılmasına izin verir.
NMODEL_x_WMAX [7:0]	Nöron modellerinin ağırlıklarının alabileceği maksimum değerdir.
NMODEL_x_WMIN [7:0]	Nöron modellerinin ağırlıklarının alabileceği minimum değerdir.
NMODEL_x_ADAPT_SPEED [7:0]	Nöronun öğrenme oranıdır.
ULEARN_START	Modülün çalışmasını başlatan giriştir.
NMODEL_x_SPIKE_TIME [7:0]	Nöron modelinin son ateşlenme zamanıdır.
PRE_SYN_FIFO_RDEN	İşlenmemiş ağırlıkların saklandığı FIFO'nun çıkışını aktif etmektedir. Böylece ULEARN modülü sırayla işlenmemiş sinaptik verileri alır.
POST_SYN_FIFO_WREN	İşlenmiş ağırlıkların saklanacağı FIFO'nun girişini aktif etmektedir. ULEARN modülü işlediği verileri ilgili FIFO'ya gönderir.
POST_SYN_DATA_OUT [63:0]	İşlenmiş sinaptik verilerin modül dışına çıkarıldığı porttur.
ULEARN_FINISH	Modülün öğrenme işlemini bitirdiğini bildirir.

ULEARN_START bitinin aktif olması ile modül çalışmaya başlamaktadır. Modül çalışmaya başladıktan sonra işlenmemiş verileri çektiği FIFO tamamen boşalana kadar çalışmaya devam eder. FIFO boşaldığı anda ULEARN_FINISH bitini '1' yaparak sisteme öğrenme operasyonunu bitirdiğini bildirir.

5.3 Girişim, Atım Zamanı Takibi ve Öğrenme Devrelerinin Birleştirilmesi

Girişim, atım zamanı takibi ve öğrenme işlemleri PROCESSING ELEMENT adı verilen bir modülün içinde birbirine entregre edilmiştir. PROCESSING ELEMENT sistolik dizinin temel yapı taşıdır. PROCESSING ELEMENT içinde XBAR, Atım Zamanı Takip Sistemi ve ULEARN öğrenme motorunun entegrasyonunu gösteren blok diyagram Şekil 5.13'te gösterilmiştir.



Şekil 5.13 Processing element blok diyagramı

PROCESSING ELEMENT ünitesi tek seferde 4 nörona ait 2 adet sinaptik veri tutabilmektedir. Girişim işleminden sonra TIMESTAMP vektörleri güncellenen 8 adet 16-bitlik sinaptik veri iki adet 64-bit uzunluğunda vektör olarak, sıraları bozulmayacak şekilde öğrenme sürecinde ULEARN ünitesinde işlenmek üzere içindeki FIFO'ya kaydetmektedir.

PROCESSING ELEMENT içindeki FIFO'nun derinliği bir PROCESSING ELEMENT ünitesinin içinde depolanması ve eğitilmesi hedeflenen nöron

ağırlıklarının sayısına göre değişmektedir. PROCESSING ELEMENT modülünün giriş çıkışlarını gösteren RTL modül görüntüsü Şekil 5.14'te verilmiştir.

Şekil 5.14 Processing element portlarını gösteren rtl modül görüntüsü

Şekil 5.14'te RTL modüldeki portların işlevleri Tablo 5.4'te verilmiştir.

Tablo 5.4 Processing element portları ve portların işlevleri

PORT	İŞLEVİ
PE_CLK	Modülün saat girişidir.
PE_RST	Modülün reset girişidir.
ROW_0_SPK_IN	Birinci satır için SPIKE BUFFER ünitelerinden atımların alındığı giriştir. Bu girişten alınan atım SYNAPSE_0x ve TIMESTAMP_0x değerleri ile işleme tabi tutulur.
ROW_0_SPK_VALID	Birinci satır için gönderilen atım değerinin geçerli olduğunu bildirir.

Tablo 5.4 Processing element portları ve portların işlevleri (devamı)

ROW_1_SPK_IN	İkinci satır için SPIKE BUFFER ünitelerinden atımların alındığı giriştir. Bu girişten alınan atım SYNAPSE_1x ve TIMESTAMP_1x değerleri ile işleme tabi tutulur.
ROW_1_SPK_VALID	İkinci satır için gönderilen atım değerinin geçerli olduğunu bildirir.
COLN_x_SYN_DIN [15:0]	Sinaptik verilerin modüle alındığı giriş portudur.
PE_SYN_CLK_EN	Sinaptik verilerin tutulduğu registerleri aktif eden giriştir.
ULEARN_LUT_x_DIN [7:0]	Nöronun öğrenme tablosunun verisidir.
ULEARN_LUT_x_ADDR [7:0]	Öğrenme tablosunda verinin yazılacağı adres bilgisidir.
ULEARN_LUT_EN_x	Öğrenme tablosunun girişini aktif eder ve tabloya veri yazılmasına izin verir.
NMODEL_x_WMAX [7:0]	Nöron modellerinin ağırlıklarının alabileceği maksimum değerdir.
NMODEL_x_WMIN [7:0]	Nöron modellerinin ağırlıklarının alabileceği minimum değerdir.
NMODEL_x_ADAPT_SPEED [7:0]	Nöronun öğrenme oranıdır.
NMODEL_x_SPIKE_TIME [7:0]	Nöron modelinin son ateşlenme zamanıdır.
ULEARN_START	ULEARN modülünü başlatır.
COLN_x_SYN_DOUT [15:0]	COLN_x_SYN_DIN [15:0] girişinden alınan sinaptik verinin bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.
PE_FIFO_FULL	PROCESSING ELEMENT içerisinde yer alan FIFO'nun dolu olduğunu bildirir.
PE_FIFO_EMPTY	PROCESSING ELEMENT içerisinde yer alan FIFO'nun boş olduğunu bildirir.
PE_FIFO_RDERR	PROCESSING ELEMENT içerisinde yer alan FIFO'dan veri çekmenin başarısız olduğunu bildiren hata sinyalidir.
PE_FIFO_WRERR	PROCESSING ELEMENT içerisinde yer alan FIFO'ya veri yazmanın başarısız olduğunu bildiren hata sinyalidir.
DOWN_ULEARN_LUT_x_DIN [7:0]	ULEARN_LUT_x_DIN [7:0] girişinden alınan öğrenme tablosu verisinin bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.

Tablo 5.4 Processing element portları ve portların işlevleri (devamı)

DOWN_ULEARN_LUT_x_ADDR [7:0]	ULEARN_LUT_x_DIN [7:0] girişinden alınan öğrenme tablosu adresinin bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.
DOWN_ULEARN_LUT_EN_x	Bir alttaki PROCESSING ELEMENT ünitesinin öğrenme tablosunun girişini aktif eder.
DOWN_NMODEL_x_WMAX [7:0]	NMODEL_x_WMAX [7:0] girişinden alınan nöronun ağırlıklarının alabileceği maksimum değerin bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.
DOWN_NMODEL_x_WMIN [7:0]	NMODEL_x_WMIN [7:0] girişinden alınan nöronun ağırlıklarının alabileceği minimum değerin bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.
DOWN_NMODEL_x_ADAPT_SPEED [7:0]	NMODEL_x_ADAPT_SPEED [7:0] girişinden alınan nöronun öğrenme oranının bir saat atımı süresi sonra bir alttaki PROCESSING ELEMENT modülüne gönderildiği çıkıştır.
POST_SYN_DATA_OUT [63:0]	Öğrenme işlemi esnasında 4 nöronun ağırlıklarının dışarıya gönderildiği çıkıştır.
POST_SYN_FIFO_WREN	Eğitlmiş ağırlıkların depolanacağı FIFO'lara veri gönderiminin kontrolünü gerçekleştiren çıkıştır.
ULEARN_FINISH	PROCESSING ELEMENT içerisindeki ULEARN ünitesinin öğrenme işlemini bitirdiğini bildirir.

5.4 Atımların Depolanması

SPIKE BUFFER ünitesi donanıma verilecek atımların depolandığı ünitedir. SPIKE BUFFER ünitesi sadece bir adet FIFO'dan oluşmaktadır. Hedef platformda 1 BRAM tüketen hazır FIFO'ların uzunlukları 1–72-bit aralığında değişmektedir. FIFO genişliği ve bağlı olduğu maksimum PROCESSING ELEMENT sayısı arasındaki ilişki (5.3)'te ortaya konulmuştur.

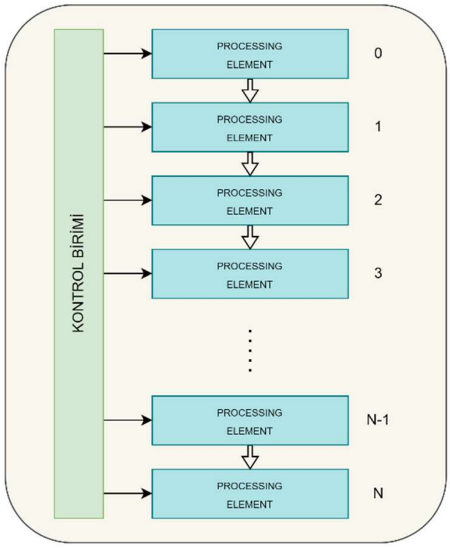
$$\text{Maksimum PE sayısı} = \frac{\text{FIFO genişliği}}{2} \tag{5.3}$$

5.5 Sistolik Dizinin Kontrolü

Sistolik dizinin temel yapı taşı olan PROCESSING ELEMENT ünitelerinin kontrol edilebilmesi için birden fazla PROCESSING ELEMENT ünitesini barındıran ve yöneten bir başka modüle ihtiyaç vardır. HYPERCOLUMN ünitesi bu amaç doğrultusunda istenen sayıda PROCESSING ELEMENT ünitesini kontrol etmek için tasarlanmıştır.

5.5.1 HYPERCOLUMN Ünitesi

HYPERCOLUMN ünitesi birden fazla PROCESSING ELEMENT ünitesinin kaskat bağlanarak sistolik dizi mimarisinin sütunlarının oluşturan ünitedir. Her HYPERCOLUMN ünitesi tek başına Nx4 boyutunda bir sistolik dizi olarak kullanılabileceği gibi, diğer HYPERCOLUMN ünitelerine de aynı atımların gönderilmesiyle NxN boyutunda bir sistolik dizi de oluşturabilmektedir. HYPERCOLUMN ünitesine ait genel blok diyagram Şekil 5.15'te verilmiştir. Donanımda kaç adet HYPERCOLUMN ünitesi bulunabileceği ve HYPERCOLUMN ünitelerinin kaç adet PROCESSING ELEMENT barındırabileceği kullanılan FPGA çipinin ne kadar kaynak içerdiğine bağlı olarak değişmektedir.



Şekil 5.15 Hypercolumn genel blok diyagramı

HYPERCOLUMN ünitesi birden fazla PROCESSING ELEMENT ünitesinin kaskat bağlanarak sistolik dizi mimarisinin sütunlarının oluşturan ünitedir. Her HYPERCOLUMN ünitesi tek başına Nx4 boyutunda bir sistolik dizi olarak

kullanılabileceği gibi, diğer HYPERCOLUMN ünitelerine de aynı atımların gönderilmesiyle NxN boyutunda bir sistolik dizi de oluşturabilmektedir. HYPERCOLUMN ünitesine ait genel blok diyagram Şekil 5.15'te verilmiştir. Donanımda kaç adet HYPERCOLUMN ünitesi bulunabileceği ve HYPERCOLUMN ünitelerinin kaç adet PROCESSING ELEMENT barındırabileceği kullanılan FPGA çipinin ne kadar kaynak içerdiğine bağlı olarak değişmektedir. Şekil 5.16'da 8 adet PROCESSING ELEMENT içeren ve 16x4'lük sistolik dizi oluşturan bir örnek HYPERCOLUMN ünitesinin RTL modül görüntüsü verilmiştir.

Şekil 5.16 16x4'lük hypercolumn ünitesi rtl modül görüntüsü

HYPERCOLUMN ünitesinin giriş çıkışları içinde bulundurduğu PROCESSING ELEMENT ünitelerinin sayısına göre değişmektedir. Sadece HYPERCOLUMN ünitesine özel olan giriş çıkışlar Tablo 5.5'te verildiği gibidir. Tablo 5.5'teki X sembolü HYPERCOLUMN içerisinde bulunan PROCESSING ELEMENT sayısını belirtmektedir.

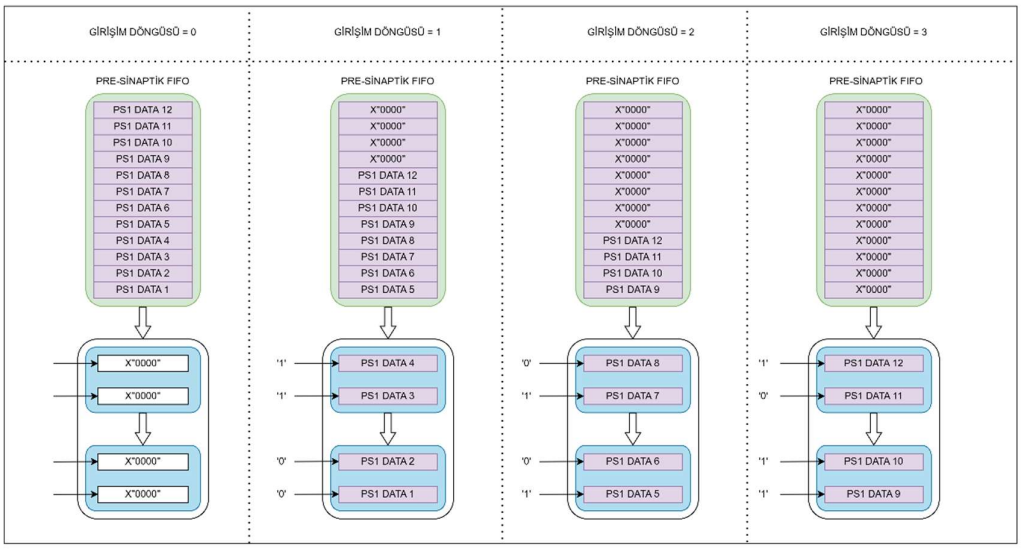
Tablo 5.5 Hypercolumn özel portları ve işlevleri

HP_MID_FIFO_FULL[X:0]	PROCESSING ELEMENT içerisinde bulunan FIFO'ların dolu olduğunu bildirir. Port genişliği PROCESSING ELEMENT sayısına eşittir.
HP_MID_FIFO_EMPTY[X:0]	PROCESSING ELEMENT içerisinde bulunan FIFO'ların boş olduğunu bildirir. Port genişliği PROCESSING ELEMENT sayısına eşittir.
HP_MID_FIFO_RDERR[X:0]	PROCESSING ELEMENT içerisinde bulunan FIFO'lardan veri çekme işleminin başarısız olduğunu bildirir. Port genişliği PROCESSING ELEMENT sayısına eşittir.
HP_MID_FIFO_WRERR[X:0]	PROCESSING ELEMENT içerisinde bulunan FIFO'lara veri gönderme işleminin başarısız olduğunu bildirir. Port genişliği PROCESSING ELEMENT sayısına eşittir.
ROW_SPIKE_VECTOR[2X:0]	SPIKE BUFFER ünitesinden alınan atımların ilgili PROCESSING ELEMENT ünitelerine gönderildiği giriş portudur. Port genişliği PROCESSING ELEMENT sayısının iki katına eşittir.
ROW_SPIKE_VECTOR_VALID[2X:0]	SPIKE BUFFER ünitesinden alınan atımların geçerli olduğunu bildirir. Port genişliği PROCESSING ELEMENT sayısının iki katına eşittir.
PE_x_POST_SYN_DOUT[63:0]	PROCESSING ELEMENT içerisinde bulunan ULEARN ünitesinin güncellenmiş sinaptik veriyi HYPERCOLUMN dışına gönderdiği çıkış portudur. Bu portların sayısı PROCESSING ELEMENT sayısına eşittir.
PE_x_POST_SYN_DOUTVALID	ULEARN ünitesinin gönderdiği verinin geçerli olduğunu bildirir. Bu portların sayısı PROCESSING ELEMENT sayısına eşittir.

Tablo 5.4'te görüldüğü gibi HYPERCOLUMN ünitesinin özel giriş çıkışlarının nitelikleri içinde bulundurduğu PROCESSING ELEMENT ünitelerinin sayısına göre değişmektedir. Sinaptik veriler ve öğrenme tabloları her zaman en üstte bulunan PROCESSING ELEMENT ünitesine verilir ve en yukarıdan başlayarak en alttaki PROCESSING ELEMENT ünitelerine kadar ulaştırılır. Tablo 5.5'teki X sembolü HYPERCOLUMN içerisinde bulunan PROCESSING ELEMENT sayısını belirtmektedir.

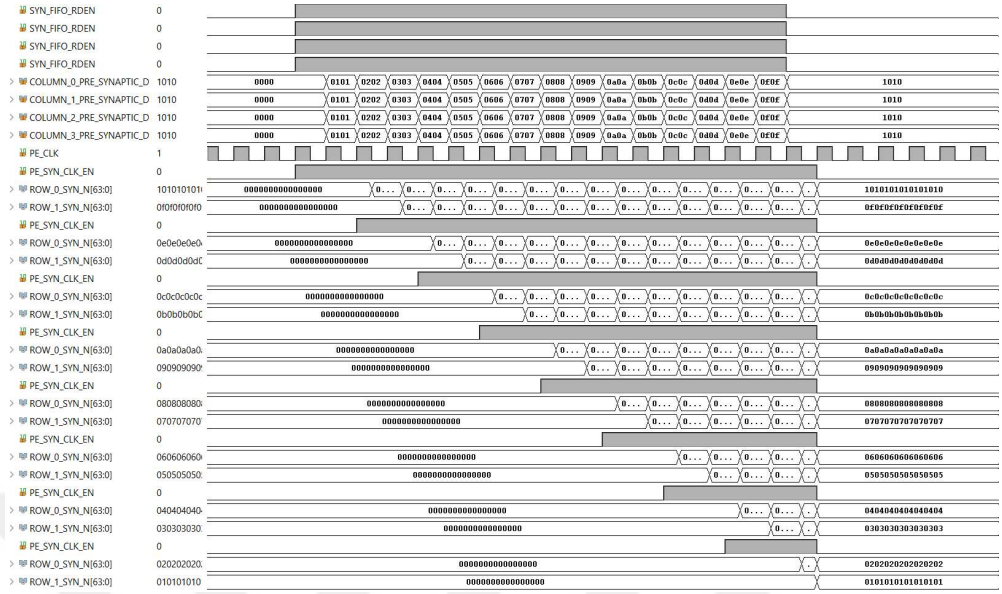
5.5.2 Sinaptik Verilerin PROCESSING ELEMENT Ünitelerine Dağıtılması

HYPERCOLUMN içindeki PROCESSING ELEMENT ünitelerine sinaptik veriler yerleştirilirken her bir PROCESSING ELEMENT ünitesinin PE_SYN_CLK_EN portu, HYPERCOLUMN içerisindeki konumlarına göre belli süre boyunca aktif tutularak PROCESSING ELEMENT içerisine sinaptik veriler FIFO'dan çekildikleri sıraya göre yerleştirilmektedir. 4x4'lük bir HYPERCOLUMN için dağıtım sisteminin 3 girişim döngüsü boyunca temsili Şekil 5.17'de verilmiştir.



Şekil 5.17 4x4'lük hypercolumn ünitesi için 3 girişim döngüsü boyunca fifodan sinaptik veri çekilmesi

FIFO'lardan her döngüde kaç tane veri çekileceği HYPERCOLUMN ünitesinin içindeki PROCESSING ELEMENT sayısı ile doğru orantılıdır. Her bir PROCESSING ELEMENT ünitesi nöron başına 2 adet sinaptik veri tutabilmektedir. Dolayısıyla Nx4'lük bir HYPERCOLUMN ünitesine tek döngüde 2N adet sinaptik veri FIFO'lardan çekilip işleme alınmaktadır. Her PROCESSING ELEMENT ünitesinin PE_SYN_CLK_EN portu konumuna göre 2 saat atımı süresi farkla aktif edilir ve aynı anda sıfıra çekilir. Bu işlemi 8 adet PROCESSING ELEMENT içeren 16x4'lük bir HYPERCOLUMN ünitesi için ayrıntılı bir biçimde gösteren test görüntüsü Şekil 5.18'de verilmiştir.

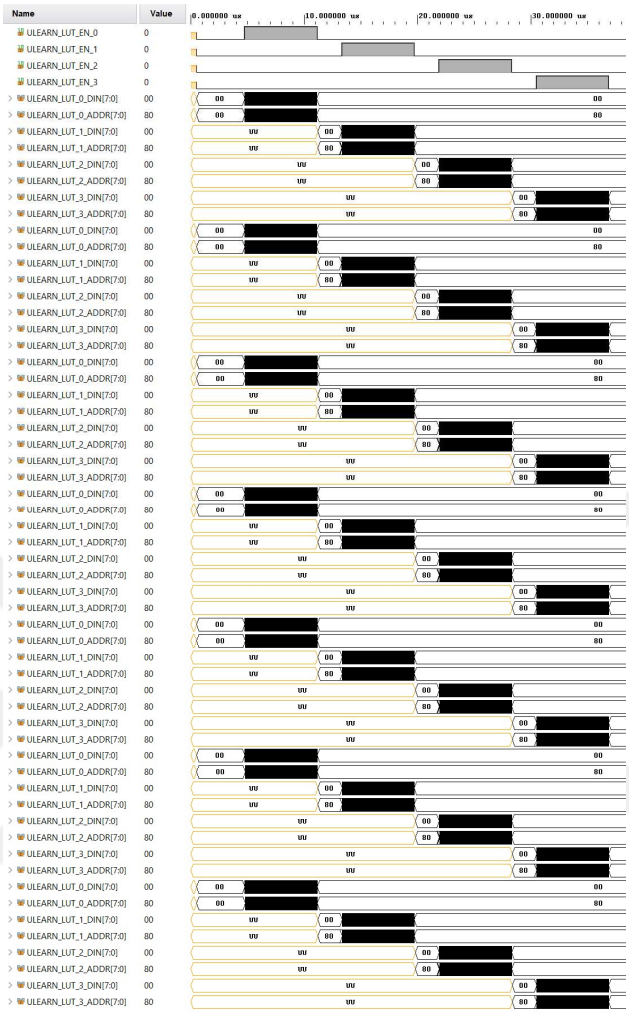


Şekil 5.18 16x4'lük hypercolumn ünitesi içindeki sinaptik veri dağıtım süreci

Şekil 5.18'de dört adet FIFO'dan çekilen 16-bit uzunluğunda sinaptik veriler sıraları bozulmayacak şekilde 64-bit uzunluğunda veriler halinde paketlenmekte ve PROCESSING ELEMENT ünitesi içerisinde işlenmektedir.

5.5.3 Öğrenme Tablolarının PROCESSING ELEMENT Ünitelerine Dağıtılması

HYPERCOLUMN içindeki PROCESSING ELEMENT ünitelerine öğrenme tabloları her seferde tek sütunun verisi gönderilerek yapılmaktadır. HYPERCOLUMN ünitelerinin sütün sayısı her zaman 4 olacak şekilde tasarlanmıştır. Bölüm 5.5.2'de olduğu gibi öğrenme tablosu verileri de en üstte bulunan PROCESSING ELEMENT ünitesinden başlayarak en alttaki üniteye kadar ulaştırılmaktadır. Tabloların tek sütun halinde verilmesinin sağladığı en büyük esneklik her nöron için farklı öğrenme kuralı uygulamaya olanak tanımasıdır. HYPERCOLUMN ünitesi içindeki öğrenme tablosu dağıtımını gösteren test görüntüsü Şekil 5.19'da verilmiştir.



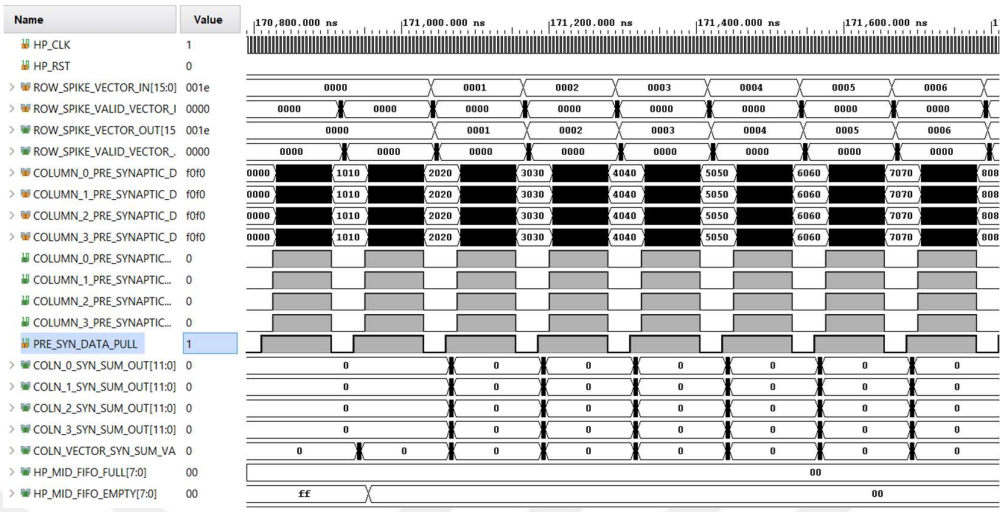
Şekil 5.19 16x4'lük hypercolumn ünitesi içindeki öğrenme tablosu dağıtım süreci

5.5.4 Girişim Sürecinin Yönetilmesi

Girişim sürecinde HYPERCOLUMN ünitesinin satır sayısı kadar sinaptik veri 5.5.2'de gösterildiği gibi FIFO'lardan çekilir ve atımlarla işleme tabi tutulur. Bir nöron için girişim döngüsü nöronun ağırlık sayısı ile HYPERCOLUMN satır sayısı cinsinden ifadesi (5.4)'te verilmiştir.

$$\text{Girişim Döngüsü} = \frac{\text{Nöron Ağırlık Sayısı}}{\text{HYPERCOLUMN Satır Sayısı}} \tag{5.4}$$

Girişim sürecinin yönetilmesini gösteren bir test görüntüsü Şekil 5.20'de verilmiştir.



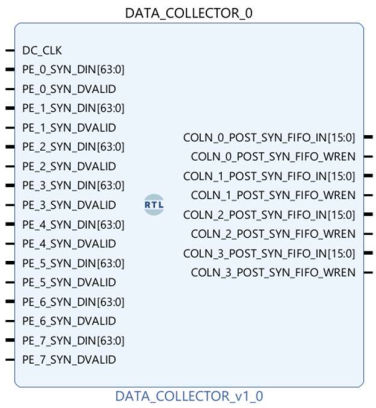
Şekil 5.20 16x4'lük hypercolumn ünitesinin girişim sürecine ait test görüntüsü

Her HYPERCOLUMN ünitesi için farklı girişim döngüsü verilebilmektedir. FIFO'larda ne kadar veri olduğundan bağımsız olarak girişim sürecinde sadece Döngü Sayısı x Satır Sayısı kadar veri FIFO'lardan çekilir. Girişim döngüsü aynı zamanda SPIKE BUFFER ünitesinden kaç adet atım vektörü çekileceğini de belirlemektedir.

5.6 Öğrenme Süreci Boyunca Güncellenen Ağırlıkların Gönderime Hazırlanması

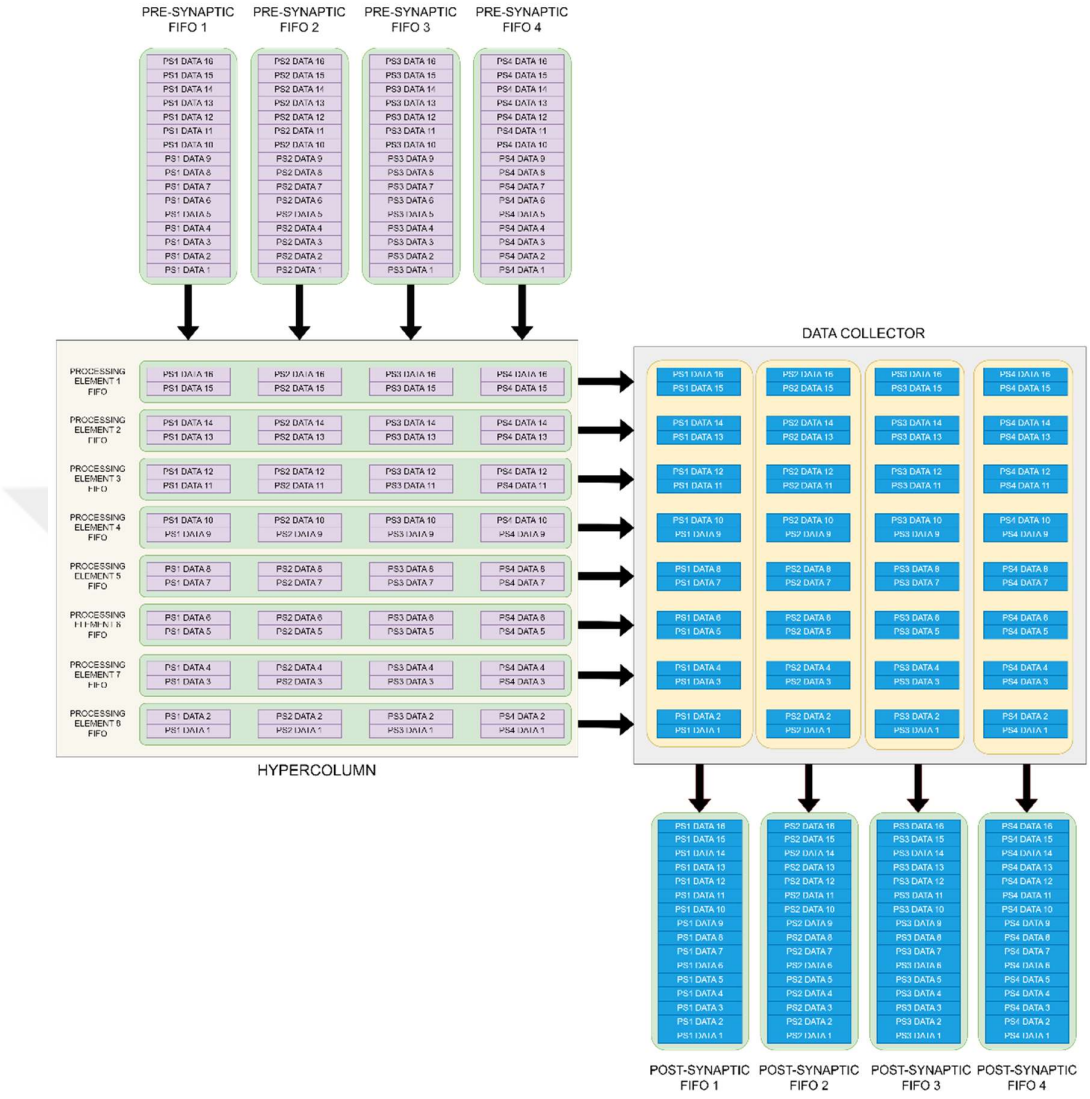
Her HYPERCOLUMN ünitesinde bulunan PROCESSING ELEMENT üniteleri içerisinde sinaptik veriler atımlarla işleme tabi tutulurken 4 adet FIFO'dan çekilen 16-Bit uzunluğunda veriler 64-Bit vektörler halinde PROCESSING ELEMENT içerisindeki FIFO'lara kaydedilmektedir. ULEARN üniteleri öğrenme sürecinde 64-Bit uzunluğundaki verileri tek seferde alıp hepsini işleyerek yine çıkışından 64-Bit uzunluğunda veri olarak dışarıya verir. Bu durum işlenmiş sinaptik verilerin sıraları bozulmadan tekrar 16-Bitlik verilere bölünmesini gerektirmektedir. Bu nedenle DATA COLLECTOR adı verilen yeni bir üniteye ihtiyaç vardır.

DATA COLLECTOR ünitesi eğitim sürecinde ULEARN ünitelerinden çıkan 64-Bit uzunluğundaki 4 nörona ait sinaptik verileri sıraları bozulmadan çıkış FIFO'larına kaydetmek için 4 adet 16-Bit veriye bölerek FIFO'lara gönderilmesinden sorumludur. Her DATA COLLECTOR ünitesi sadece bir adet HYPERCOLUMN ünitesine hizmet etmektedir. 16x4'lük HYPERCOLUMN ünitesine hizmet eden DATA COLLECTOR ünitesine ait RTL modül görüntüsü Şekil 5.21'de verilmiştir.



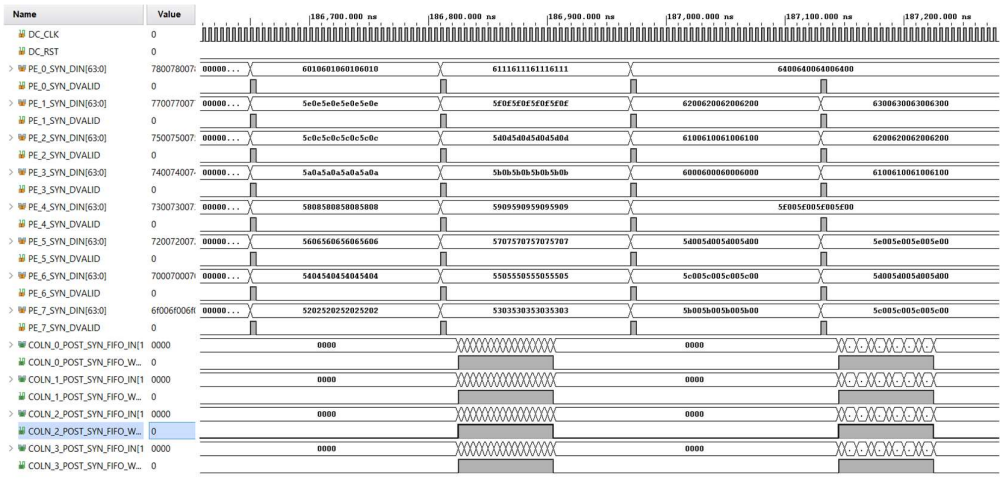
Şekil 5.21 16x4'lük hypercolumn ünitesine hizmet eden data collector ünitesine ait rtl modül görüntüsü

DATA COLLECTOR ünitesinin giriş portu sayısı bağlandığı HYPERCOLUMN ünitesinin içinde bulunan PROCESSING ELEMENT sayısına eşittir. DATA COLLECTOR ünitesinin çıkış portları ise HYPERCOLUMN sütün sayısı olan 4'e eşittir. DATA COLLECTOR ünitesinin yarattığı FIFO hiyerarşisine ait blok diyagram Şekil 5.22'de verilmiştir.



Şekil 5.22 16x4'lük hypercolumn ünitesine hizmet eden data collector ünitesinin yarattığı fifo hiyerarşisi

Şekil 5.22'de hiyerarşisi verilen DATA COLLECTOR ünitesine ait test görüntüsü Şekil 5.23'te verilmiştir.



Şekil 5.23 16x4'lük hypercolumn ünitesine hizmet eden data collector ünitesinin test görüntüsü

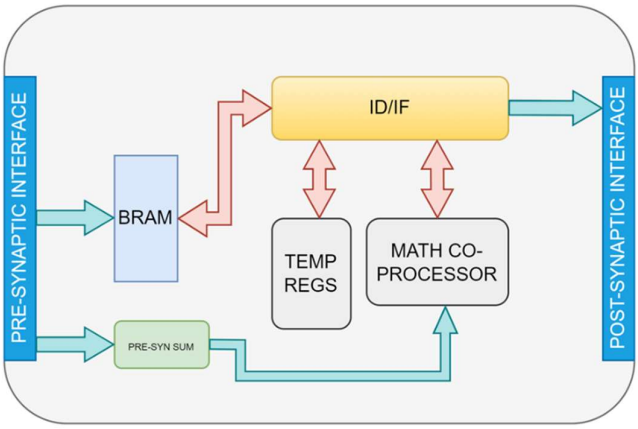
DATA COLLECTOR ünitesi HYPERCOLUMN üniteleri içinde bulunan ULEARN modülleri tarafından yönetilmektedir. ULEARN üniteleri veri göndermeye başladığında DATA COLLECTOR çalışmaya başlar. ULEARN modülleri eğitim sürecini tamamladığı anda DATA COLLECTOR bütün verileri çıkış FIFO'larına göndermiş olur ve çalışmayı durdurur.

5.7 Nöron Modelinin Çalıştırılması

Nöron modelinin çalıştırılması işlemi donanımın temel yapı taşlarından ikincisidir. SNN ağlarında nöron modelleri Bölüm 2'de bahsedildiği gibi çoğunlukla biyolojik nöronlardan ilham alan, zamana bağlı basit diferansiyel denklemlerden oluşmaktadır. Donanımın yeniden kullanılabilirliğini artırabilmek ve değişik problemlerin çözümünde kullanabilmek için sabit bir nöron modeli yerine esnek ve programlanabilir bir modülün tasarlanması ihtiyacı doğmuştur. Bu amaç doğrultusunda tasarlanan NMC ünitesi, herhangi bir SNN nöron modeline uyum sağlayabilecek bir yapıda tasarlanmıştır.

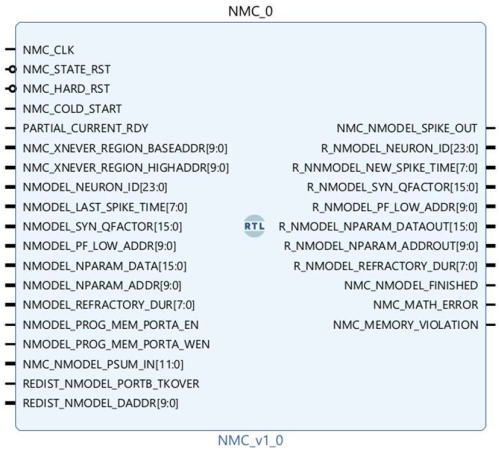
5.7.1 NMC Ünitesi

NMC ünitesi nöron modellerini önceden belirlenen bir program akışı ile sıralı bir biçimde çalıştırmaktadır. NMC ünitesinin basit blok diyagramı Şekil 5.24'te verilmiştir.



Şekil 5.24 Nmc basit blok diyagramı

NMC ünitesine ait RTL modül görüntüsü Şekil 5.25'te verilmiştir.



Şekil 5.25 Nmc ünitesine ait rtl modül görüntüsü

Şekil 5.24'te verilen NMC RTL modül görüntüsünde var olan giriş çıkış portlarının işlevleri Tablo 5.6'da verilmiştir.

Tablo 5.6 Nmc giriş çıkış portları ve işlevleri

PORT	İŞLEVİ
NMC_CLK	NMC saat girişidir.
NMC_STATE_RST	NMC modülünün durumlarını ve özel durum registerlerini sıfırlar.
NMC_HARD_RST	NMC modülünü program hafızası da dahil bütünsel olarak sıfırlar.
NMC_COLD_START	NMC modülünü başlatır.
PARTIAL_CURRENT_RDY	HYPERCOLUMN ünitesinin bir girişim döngüsü için girişim sonucunun alınmaya hazır olduğu bilgisinin alındığı giriştir.
NMC_XNEVER_REGION_BASEADDR [9:0]	NMC modülünün hafızasının veri bölümünün başlangıç adresidir.
NMC_XNEVER_REGION_HIGHADDR [9:0]	NMC modülünün hafızasının veri bölümünün bitiş adresidir.
NMODEL_NEURON_ID [23:0]	NMC modülüne verilen nöronun kimlik bilgisidir. 24-Bitlik bir işaretsiz sayıdır.
NMODEL_LAST_SPIKE_TIME [7:0]	Nöronun en son ateşlenme zamanıdır.
NMODEL_SYN_QFACTOR [15:0]	Nöron modelinin ağırlıklarının kuantalama faktörüdür. FP16 formatında bir sayıdır.
NMODEL_PF_LOW_ADDR [9:0]	Nöron modelinin program akışının başlangıç adresidir.
NMODEL_NPARAM_DATA [15:0]	NMC modülüne program akışının ve nöron durumlarının verildiği giriştir.
NMODEL_NPARAM_ADDR [9:0]	NMC modülünün hafızasına yazılacak verilerin hangi adrese yazılacağını bilgisinin alındığı giriş portudur.
NMODEL_REFRACTORY_DUR [7:0]	Nöron modelinin inert periyodudur.
NMODEL_PROG_MEM_PORTA_EN	NMC modülünün hafızasının A portunun Pre-Sinaptik Arayüz tarafından devralındığı giriş portudur.
NMODEL_PROG_MEM_PORTA_WEN	NMC modülünün hafızasının A portunun yazım kontrolüdür.
NMC_NMODEL_PSUM_IN [11:0]	HYPERCOLUMN girişim sonucunun modüle alındığı giriş portudur.

Tablo 5.6 Nmc giriş çıkış portları ve işlevleri (devamı)

REDIST_NMODEL_PORTB_TKOVER	NMC modülünün hafızasının B portunun Post-Sinaptik Arayüz tarafından devralındığı giriş portudur.
REDIST_NMODEL_DADDR [9:0]	Post-Sinaptik Arayüzün NMC modülünün hafızasının B portundan okuyacağı verilerin adres bilgisinin alındığı giriş portudur.
NMC_NMODEL_SPK_OUT	Çalıştırılan nöron modelinin ateşlenip ateşlenmediğini bildiren çıkış portudur.
R_NMODEL_NEURON_ID [23:0]	Post-Sinaptik Arayüzün çalıştırılan nöron modelinin kimlik bilgisini okuduğu çıkış portudur.
R_NMODEL_LAST_SPIKE_TIME [7:0]	Nöron modelinin güncellenmiş en son ateşlenme zamanının Post-Sinaptik Arayüzden okunduğu çıkış portudur.
R_NMODEL_SYN_QFACTOR [15:0]	Post-Sinaptik Arayüzün çalıştırılan nöron modelinin ağırlıklarının kuantalama faktörünü okuduğu çıkış portudur.
R_NMODEL_PF_LOW_ADDR [9:0]	Post-Sinaptik Arayüzün çalıştırılan nöron modelinin başlangıç adresi bilgisini okuduğu çıkış portudur.
R_NMODEL_NPARAM_DATAOUT [15:0]	Post-Sinaptik Arayüzün çalıştırılan nöron modelinin güncellenen durumlarını okududuğu çıkış portudur.
R_NMODEL_NPARAM_ADDROUT [15:0]	Post-Sinaptik Arayüzün NMC modülünün hafızasından veri çekmek için kullandığı adres giriş portudur.
R_NMODEL_REFRACTORY_DUR [7:0]	Post-Sinaptik Arayüzün nöron modelinin kalan inert periyot zamanını okuduğu çıkış portudur.
NMC_NMODEL_FINISHED	Nöron modelinin çalıştırılmasının bittiğini bildiren çıkış portudur.
NMC_MATH_ERROR	NMC ünitesinde nöron modelinin çalıştırılması esnasında bir aritmetik hata gerçekleştiğini bildiren çıkış portudur.
NMC_MEMORY_VIOLATION	Nöron modelinin program akışında bir hata olduğunu bildiren çıkış portudur.

NMC ünitesi FP16 veri formatıyla çalışmaktadır. NMC içindeki tüm aritmetik işlemler FP16 veri tipiyle yapılmaktadır. FP16 kullanılmasının nedeni nöron modellerinin hesaplanmasında yüksek doğruluk sağlamak, nöron modellerinde var olan bölme işlemlerini fazladan donanım kullanmadan yapmak ve nöron modellerinin istenen numerik yöntemlerle çalıştırılmasını sağlamaktır.

5.7.2 NMC Komut Seti

NMC komut seti 13 adet komuttan oluşmaktadır. Komutların dil yapıları, açıklamaları ve çevrim süreleri Tablo 5.7'de detaylı bir biçimde verilmiştir.

Tablo 5.7 Nmc komut seti

Komut	**Kısaltması**	**Kullanımı**	**Açıklama**	**Çevrim Süresi - CPI**
LOAD WORD	LW	lw x1, M(x)	M(X)+ NMC_XNEVER_REGION_BASEADDR memory adresinde bulunan veriyi x1 registerine kaydet.	4 Periyot
STORE WORD	SW	sw x1, M(x)	x1 Registerindeki değeri M(X)+ NMC_XNEVER_REGION_BASEADDR memory adresine kaydet.	4 Periyot
GET ACCUMMULATOR	GACC	gacc, x1	FMAC16 akümülatöründeki değeri x1 registerine kaydet.	5 Periyot
FP16 MULTIPLY-ACCUMULATE	FMAC	fmac, x1, x2	x1 ve x2 registerlerindeki değerleri çarp ve akmülatördeki değerle topla	4 Periyot
FP16 MULTIPLY-SUBTRACT	SMAC	smac, x1, x2	x1 ve x2 registerlerindeki değerleri çarp ve akmülatördeki değerden çıkar.	4 Periyot
CLEAR ACCUMMULATOR	CLRACC	clracc	Akümülatörü sıfırlar.	2 Periyot
COMPARE	COMP	comp, x1, x2	x1 ve x2 registerindeki değerleri karşılaştırır.	2 Periyot
BRANCH-IF-LESS	BIL	bil,23	LESS bayrağı 1 ise program sayacını 23 artırır.	4 Periyot
BRANCH-IF-EQUAL	BIE	bie,32	EQUAL bayrağı 1 ise program sayacını 32 artırır.	4 Periyot
BRANCH-IF-GREAT	BIG	big,22	GREAT bayrağı 1 ise program sayacını 22 artırır.	4 Periyot
SEND SPIKE	SPK	spk	Nöronun ateşlendiğini bildiren SPK_OUT çıkışını '1' yapar.	4 Periyot
PROGRAM FLOW COMPLT	RETURN	ret	Program akışını sonlandırır	4 Periyot
SET REFRACTORY PERIOD	STRF	strf,21	21 döngü boyunca sürecek refractory periodu oluşturur.	4 Periyot

NMC komut setinde sadece iki adet aritmetik komut bulunmaktadır. “fmac” ve “smac” komutları bütün nöron modellerini hesaplamak için yeterlidir. Nöron modelleri “çarpma ve akümüle etme” komutlarıyla hesaplanmaktadır. “çarpma ve akümüle etme” komutları tezin geri kalanında MAC komutları olarak anılacaktır. (5.4) ve (5.5) denklem çifti Izhikevich nöron modelinin diferansiyel denklemleridir.

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I \tag{5.4}$$

$$\frac{du}{dt} = a(bv - u) \tag{5.5}$$

Diferansiyel denklem çiftinin fark denklemine dönüştürülmesi için birçok numerik yöntem bulunmaktadır. En sık kullanılan ve en az işlem gücü gerektiren numerik yöntem olan Euler yönteminin Izhikevich nöronunun denklem çiftinin diferansiyel ifadelerine uygulanması (5.6) ve (5.7) eşitlikleri kullanılarak gerçekleştirilir.

$$\frac{dv}{dt} = \frac{v[n+1] - v[n]}{h} \tag{5.6}$$

$$\frac{du}{dt} = \frac{u[n+1] - u[n]}{h} \tag{5.7}$$

(5.6) ve (5.7) ifadeleri (5.4) ve (5.5) de yerine koyulup düzenlenerek Izhikevich nöronunun fark denklemleri (5.8) ve (5.9) oluşturulur.

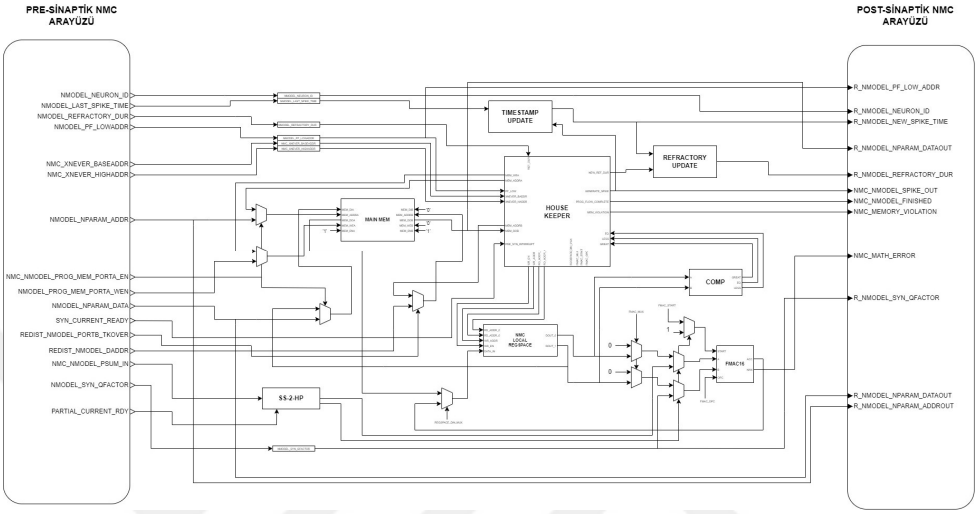
$$v[n+1] = 0.04hv^2[n] + 5hv[n] + 140h - hu[n] + hI[n] + v[n] \tag{5.8}$$

$$u[n+1] = ha(bv[n] - u[n]) + u[n] \tag{5.9}$$

Oluşturulan fark denklemleri MAC komutları kullanılarak hesaplanmaya uygundur. Birden fazla terimin çarpımını içeren ifadeler sıralı çarpım ve registerlere kaydedilme ile, toplama ve çıkarma işlemleri ise toplanacak veya çıkarılacak sayının FP16 formatında 1.0 ile çarpılması ve akümülatörle işleme tabi tutulması ile yapılır.

5.7.3 NMC Mimarisi

NMC ünitesinin, bütün alt modüllerini içeren mimari yapısı Şekil 5.26'da verilmiştir.



Şekil 5.26 Nmc mimari diyagramı

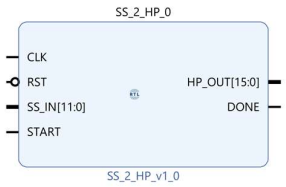
NMC ünitesinin Pre-Sinaptik ve Post-Sinaptik olmak üzere iki adet arayüzü vardır. Pre-Sinapik arayüzden nöron modelinin durum parametreleri, son ateşlenme zamanı, kalan inert periyot durumu, girişim sonucu verileri nöron durumlarını güncellemek için NMC içerisine alınır. NMC güncelleme işlemini bitirdikten sonra Post-Sinaptik arayüzden güncel nöron durumlarının, son atım zamanının ve kalan inert periyot sürecinin güncellenmiş halini ve nöronun ateşlenip ateşlenmediği bilgisi dışarıya göndermek üzere arayüzde hazır tutar.

5.7.4 NMC Alt Modülleri

Şekil 5.26'da görülen NMC mimari yapısının içinde bulunan alt modüller ve işlevleri bu bölümde anlatılmaktadır.

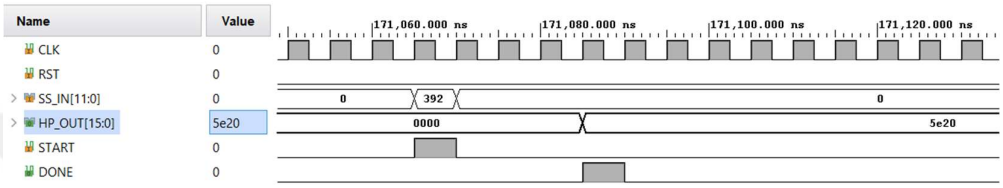
5.7.4.1 SS-2-HP Modülü

SS-2-HP modülü NMC ünitesine gelen girişim döngüsü sonucunu 12-Bit işaretli tamsayı formatından FP16 formatına çevirmektedir. Modülün RTL görüntüsü Şekil 5.27'de verilmiştir.



Şekil 5.27 Ss-2-hp modülü rtl görüntüsü

Tip dönüştürme işlemine ait test görüntüsü Şekil 5.28'de görülmektedir.



Şekil 5.28 Ss-2-hp modülü tip dönüşüm işlemi

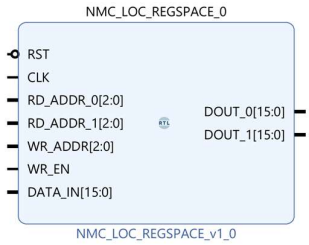
Şekil 5.27'de SS_IN girişinden girişim sonucu 392 olarak modüle verilmiştir. 392 sayısının FP16 formatında karşılığı onaltılık formda X"5E20"'ye karşılık gelmektedir.

5.7.4.2 NMC MAIN MEMORY Modülü

NMC ünitesinde nöron program akışlarının ve nöron durumlarının saklandığı hafızadır. Her NMC ünitesi 16-Bit uzunluğunda kelimelerle çalışır ve toplam 1024 kelimelik bir hafızaya sahiptir. Bu sayede NMC ünitesi birden fazla nöron modelini hafızasında tutup program akışının başlangıç adreslerine göre istenen nöron modelini çalıştırmaktadır. MAIN MEMORY modülü 2 portlu BRAM kullanılarak tasarlanmıştır. Komutlar Port B'den çekilmektedir. Nöron durumları ve sabitler ise Port A kullanılarak yazılıp okunmaktadır.

5.7.4.3 NMC LOCAL REGSPACE Modülü

SS-2-HP, FMAC16, COMP ve NMC_MAIN_MEMORY arasında veri alış verişinin yapıldığı ünitedir. Her NMC ünitesi 16-Bit genişliğinde 8 adet register bulundurmaktadır. Modülün çıkışları FMAC16 ve COMP modüllerine bağlıdır. Modülün RTL görüntüsü Şekil 5.29'da verilmiştir.



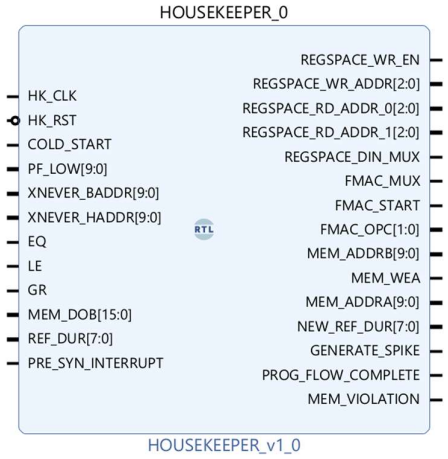
Şekil 5.29 Nmc_local_regspace rtl modül görüntüsü

5.7.4.4 TIMESTAMP UPDATE Modülü

Nöronun son ateşlenme zamanını güncelleyen ünitedir. Nöron ateşlenmezse NMODEL_SPIKE_TIME değerini 1 artırarak dışarı verir. Nöron ateşlenirse 0 değerini döndürür.

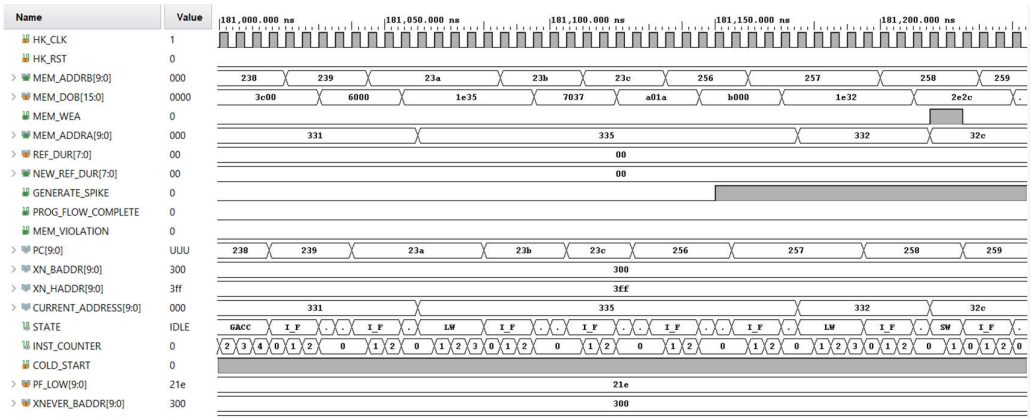
5.7.4.5 HOUSEKEEPER Modülü

NMC ünitesinin en önemli alt modülüdür. NMC hafızasından çekilen tüm komutları hafızadan sırayla çekip işler ve komutların gerektirdiği işlemleri alt modülleri yöneterek yapar. Modülün RTL görüntüsü Şekil 5.30'da görülmektedir.



Şekil 5.30 House keeper rtl modül görüntüsü

HOUSEKEEPER modülüne ait bir test görüntüsü Şekil 5.31'de verilmiştir.



Şekil 5.31 House keeper test görüntüsü

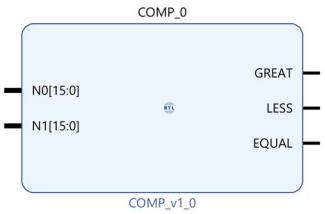
NMC komutları hafızadan her seferinde tek bir komut alınarak işlenmektedir. Bir komutun işlenmesi bitmeden bir sonraki komuta geçilmemektedir. Modül, COLD_START girişinin aktif edilmesi ile Pre-Sinaptik NMC arayüzünden çalışması için gerekli bilgileri alıp çalışmaya başlar ve program akışının sonuna gelindiğinde PROG_FLOW_COMPLETE çıkışını aktif ederek nöron modelinin durumlarının hesaplandığını ve gönderime hazır olduğunu bildirir.

5.7.4.6 REFRACTORY UPDATE Modülü

Nöron modelinin inert periyodunu hesaplayan alt modüldür. Girişteki inert periyot değerini her adımda bir azaltarak dışarıya verir. İnert periyot aktif olduğu sürece SS-2-HP modülünü devre dışı bırakarak girişim sonucunun alınmasını engeller.

5.7.4.7 COMP Modülü

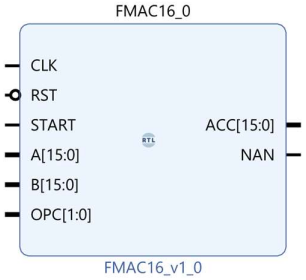
İki adet FP16 formatındaki sayıyı büyüklük bakımından karşılaştıran ünitedir. Tablo 5.6'da verilen "bil", "bie" ve "big" dallanma komutlarında kullanılmaktadır. GREAT, LESS ve EQUAL çıkışları aynı anda aynı değere sahip olamazlar. Modül NMC aktif olduğu sürece çalışmaktadır. Yalnızca dallama komutları işlendiğinde COMP çıkışları dikkate alınır. COMP modülünün RTL görüntüsü Şekil 5.32'de verilmiştir.



Şekil 5.32 Comp modülü rtl görüntüsü

5.7.4.8 FMAC16 Modülü

NMC içerisindeki tüm aritmetik işlemlerin yapıldığı birimdir. FP16 formatında MAC (Multiply-Accummulate/Subtract) işlemi yapmaktadır. Modülün RTL görüntüsü Şekil 5.33'te görülebilmektedir.



Şekil 5.33 Fmac16 modülü rtl görüntüsü

FMAC16 4 saat periyodu gecikmeye sahiptir. Her 4 saat periyodunda bir A ve B girişlerinden verilen FP16 formatındaki sayıları çarpıp OPC portundan verilen komuta göre akümülatördeki değerle toplayıp çıkarabilmektedir. FMAC16 komutları ve komutların gerçekleştirdikleri işlemler Tablo 5.7'de verilmiştir.

Tablo 5.8 Fmac16 komutları ve komutların gerçekleştirdiği işlemler

OPC [1:0]	İşlem
"00"	ACC = ACC + AxB
"01"	ACC = ACC − AxB
"10"	ACC = 0

FMAC16 ünitesinde akümülatördeki değerle bir başka değeri tek başına işleme sokabilmek için nöron modellerinin program akışlarına FP16 formatında 1.0 bir sabit olarak program akışına dahil edilir.

5.7.5 NMC Çalışma Prensibi

NMC Pre-Sinaptik ve Post-Sinaptik arayüzlerinden NMC hafızasına doğrudan ulaşılabilmektedir. NMC ünitesinde nöron durumlarının hesaplanabilmesi için nöron modelinin program akışınının bilinmesi gerekmektedir.

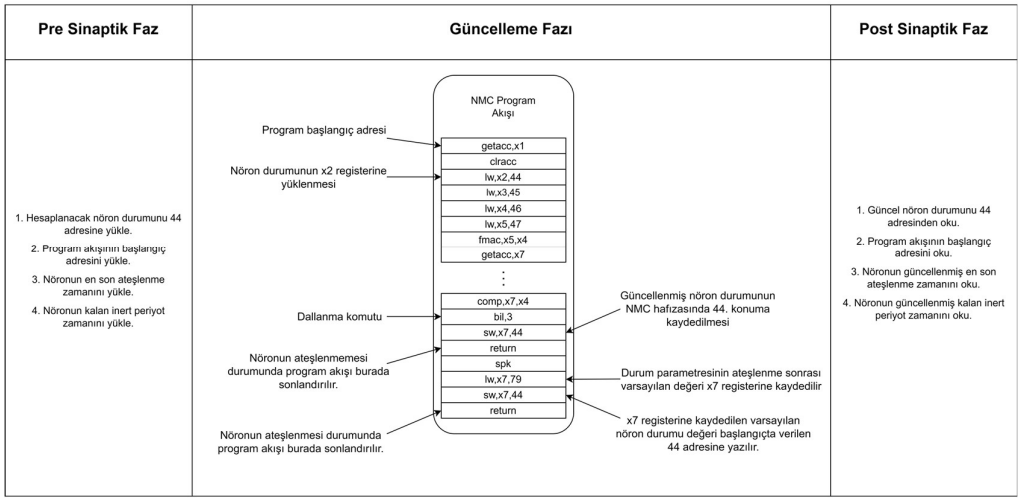
NMC çalışma süreci üç ayrı fazdan oluşmaktadır.

1.Pre-Sinaptik Faz: Nöron durumlarının program akışında hafızadan çekildiği adreslere nöron modelinin başlangıç durumlarının yazıldığı süreçtir. Bu süreçte Pre-Sinaptik arayüz NMC hafızasının A girişini devralır.

2.Güncelleme Fazı: NMC ünitesinin program akışını başlatıp bitirdiği süreçtir.

3.Post-Sinaptik Faz: Güncellenmiş nöron durumlarının ve istenen diğer parametrelerin NMC hafızasından çekildiği süreçtir. Bu süreçte Post-Sinaptik arayüz NMC hafızasının B girişlerini devralarak istenen parametrelerin hafızadan çekilmesini sağlar.

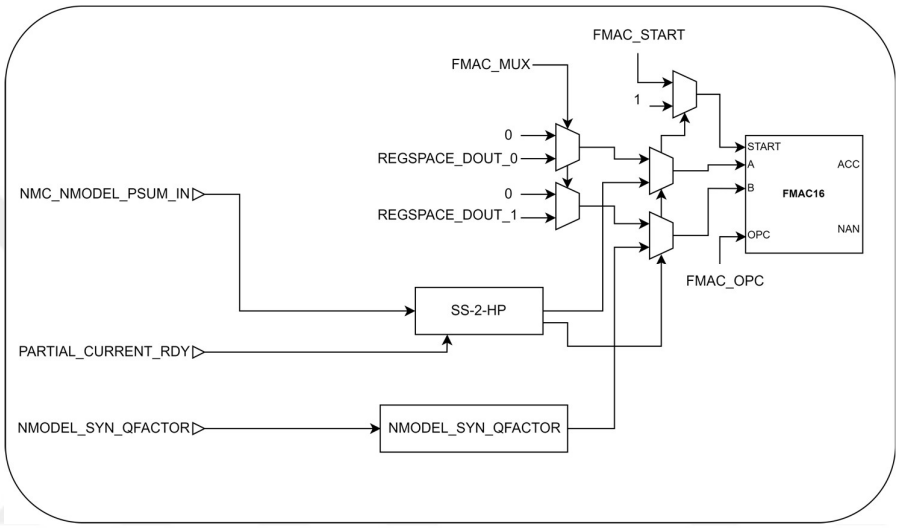
NMC'nin üç çalışma fazının tek durum parametresi bulunan bir nöron modeli için örneklendirilmesi Şekil 5.34'te görülmektedir.



Şekil 5.34 Nmc çalışma fazlarının tek durum parametresi bulunan bir nöron modeli için örneklendirilmesi

5.7.6 Girişim Sonucunun NMC Ünitesi İçinde Girişim Süresi Boyunca Oluşturulması

Bölüm 5.5.3'te detaylandırılan girişim işleminde her girişim döngüsü tamamlandığında girişim sonucu Pre-Sinaptik arayüzden NMC içerisine alınmaktadır. Gönderilen sonucu işleyen NMC mimari parçası Şekil 5.35'te verilmiştir.



Şekil 5.35 Girişim sonucunun nmc içerisinde oluşturulduğu kısmın blok diyagramı

Gönderilen girişim sonucu PARTIAL_CURRENT_RDY girişi aktif edildiğinde SS-2-HP ünitesi tarafından FP16 formatına dönüştürülmektedir. SS-2-HP ünitesi FMAC seçicilerini kontrol ederek girişimin FP16 değerini FMAC16 ünitesinin A portuna yönlendirir.

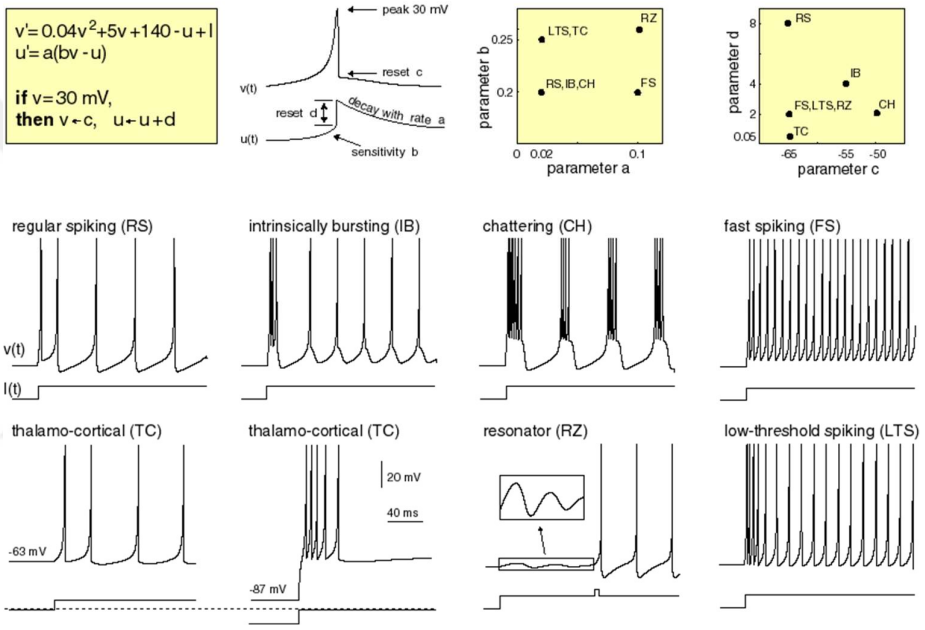
PARTIAL_CURRENT_RDY girişi aynı zamanda NMODEL_SYN_QFACTOR değerini de FMAC16 ünitesinin B portuna yönlendirir. Dönüştürülen girişim sonucu NMODEL_SYN_QFACTOR değeri ile her girişim sonucunda FMAC16 ünitesi içerisinde çarpılarak akümüle edilir. Girişim işlemi bittiğinde FMAC16 akümülatöründe girişim sonucunun FP16 formatındaki kesin değeri NMC ünitesi içerisinde oluşturulur.

Girişim sonucunun NMC içinde oluşturulması nedeniyle her nöron program akışı "getacc" komutu ile başlamak zorundadır. Girişim sonucu getacc komutu ile istenen NMC registerine yazılır ve daha sonra nöron durumlarının hesaplanmasında kullanılır. Girişim sonucu istenildiği takdirde program akışı ile herhangi bir

registere yazıldıktan sonra NMC hafızasına yönlendirilip nöronun giriş akımının zamanla değişimini görüntülemekte kullanılabilir.

5.7.7 Izhikevich Nöron Modeli İçin NMC Program Akışı Örneği

Izhikevich nöron modeli parametrelerinin değer aralıklarına göre oldukça değişken ateşlenme karakteristikleri göstermektedir. Izhikevich nöronunun ateşleme karakteristiklerinden 8 tanesi Şekil 5.36'da verilmiştir.

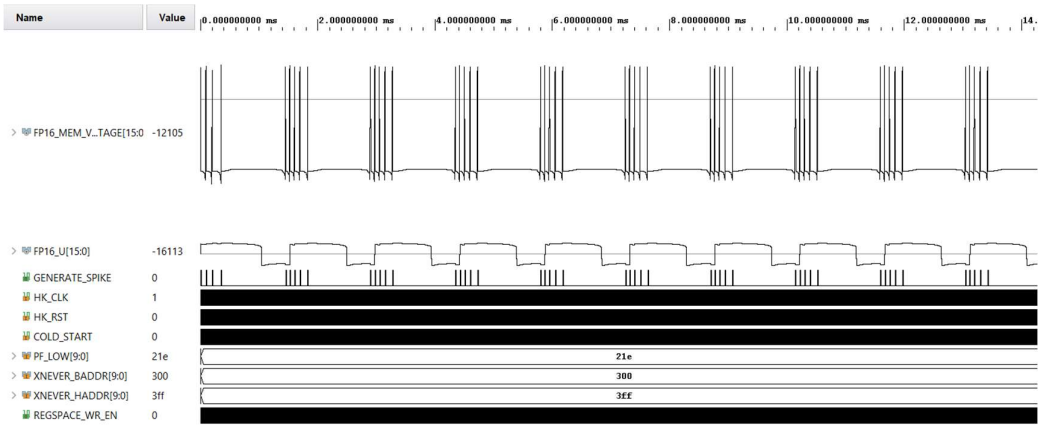


Şekil 5.36 Izhikevich nöronu ateşlenme karakteristikleri [9]

Izhikevich nöronu için oluşturulan NMC komut dizini ve komutların denklemleri adım adım nasıl işlediği Şekil 5.37'de gösterilmektedir. Program akışı (5.8) ve (5.9) ifadeleri kullanılarak oluşturulmuştur. (5.8) ve (5.9) ifadelerinde Izhikevich nöronu Euler yöntemi kullanılarak diferansiyel denklemden fark denklemine dönüştürülmüştür. Programın çıktılarına ait test görüntüleri Şekil 5.38'de gösterilmiştir.

```
1.  "getacc,x1 "  <-I girişim sonucu değerini x1 registerine kaydet.
2.  "clracc    "  <-Akümülatörü sıfırla.
3.  "lw,x2,44  "  <-V durum parametresini NMC hafızasının 44. konumundan al ve x2 registerine kaydet.
4.  "lw,x3,45  "  <-h sabitini NMC hafızasının 45. konumundan al ve x3 registerine kaydet.
5.  "lw,x4,46  "  <-U durum parametresini NMC hafızasının 46. konumundan al ve x4 registerine kaydet.
6.  "lw,x5,47  "  <-140 sabitini NMC hafızasının 47. konumundan al ve x5 registerine kaydet.
7.  "fmac,x2,x0"  <-ACC <= v
8.  "fmac,x1,x3"  <-ACC <= v + I*h
9.  "smac,x4,x3"  <-ACC <= v + I*h - h*u
10. "fmac,x5,x3"  <-ACC <= 140*h - h*u + I*h + v
11. "getacc,x6 "  <-Akümülatördeki 140*h - h*u + I*h + v değerini x6 registerine kaydet.
12. "clracc    "  <-Akümülatörü sıfırla.
13. "fmac,x3,x2"  <-ACC <= h*v
14. "getacc,x7 "  <-Akümülatördeki h*v değerini x7 registerine kaydet.
15. "clracc    "  <-Akümülatörü sıfırla.
16. "lw,x5,48  "  <-5 sabitini NMC hafızasının 48. konumundan al ve x5 registerine kaydet.
17. "fmac,x7,x5"  <-ACC <= 5*h*v
18. "fmac,x6,x0"  <-ACC <= 5*h*v + 140*h - h*u + I*h + v
19. "getacc,x6 "  <-Akümülatördeki 5*h*v + 140*h - h*u + I*h + v değerini x6 registerine kaydet.
20. "clracc    "  <-Akümülatörü sıfırla.
21. "lw,x5,49  "  <-0.04 sabitini NMC hafızasının 49. konumundan al ve x5 registerine kaydet.
22. "fmac,x7,x5"  <-ACC <= 0.04*h*v
23. "getacc,x7 "  <-Akümülatördeki 0.04*h*v değerini x7 registerine kaydet.
24. "clracc    "  <-Akümülatörü sıfırla.
25. "fmac,x7,x2"  <-ACC <= 0.04*h*v*v
26. "fmac,x6,x0"  <-ACC <= 0.04*h*v*v + 5*h*v + 140*h - h*u + I*h + v
27. "getacc,x6 "  <-Akümülatördeki 0.04*h*v*v + 5*h*v + 140*h - h*u + I*h + v değerini x6 registerine kaydet.
28. "clracc    "  <-Akümülatörü sıfırla.
29. "lw,x7,53  "  <-V_th sabitini NMC hafızasının 53. konumundan al ve x7 registerine kaydet.
30. "comp,x6,x7"  <-x6 registerindeki yeni V değeri ile x7 registerindeki V_th değerini karşılaştır.
31. "big,26    "  <-Eğer V > V_th ise bu konumdan 26 komut sonrasına atla. Eğer şart sağlanmadıysa sonraki komuttan devam et.
32. "sw,x6,44  "  <-Güncellenmiş V durum değerini x6 registerinden al ve NMC hafızasının 44. konumuna kaydet.
33. "lw,x5,51  "  <-a sabitini NMC hafızasının 51. konumundan al ve x5 registerine kaydet.
34. "lw,x6,52  "  <-b sabitini NMC hafızasının 52. konumundan al ve x6 registerine kaydet.
35. "fmac,x5,x6"  <-ACC <= a*b
36. "getacc,x7 "  <-Akümülatördeki a*b değerini x7 registerine kaydet.
37. "clracc    "  <-Akümülatörü sıfırla.
38. "fmac,x2,x3"  <-ACC <= v*h
39. "getacc,x1 "  <-Akümülatördeki v*h değerini x1 registerine kaydet.
40. "clracc    "  <-Akümülatörü sıfırla.
41. "fmac,x1,x7"  <-ACC <= v*h*a*b
42. "fmac,x4,x0"  <-ACC <= v*h*a*b + u
43. "getacc,x1 "  <-Akümülatördeki v*h*a*b + u değerini x1 registerine kaydet.
44. "clracc    "  <-Akümülatörü sıfırla.
45. "fmac,x3,x5"  <-ACC <= h*a
46. "getacc,x7 "  <-Akümülatördeki h*a değerini x7 registerine kaydet.
47. "clracc    "  <-Akümülatörü sıfırla.
48. "fmac,x7,x4"  <-ACC <= h*a*u
49. "getacc,x7 "  <-Akümülatördeki h*a*u değerini x7 registerine kaydet.
50. "clracc    "  <-Akümülatörü sıfırla.
51. "fmac,x0,x1"  <-ACC <= v*h*a*b + u
52. "smac,x0,x7"  <-ACC <= v*h*a*b + u - h*a*u
53. "getacc,x7 "  <-Akümülatördeki v*h*a*b + u - h*a*u değerini x7 registerine kaydet.
54. "clracc    "  <-Akümülatörü sıfırla.
55. "sw,x7,46  "  <-Güncellenmiş u durum değerini x7 registerinden al ve NMC hafızasının 46. konumuna kaydet.
56. "return    "  <-Program akışını sonlandır.
57. "spk       "  <-Nöronun ateşlendiğini bildir.
58. "lw,x7,50  "  <-c sabitini NMC hafızasının 50. konumundan al ve x7 registerine kaydet.
59. "sw,x7,44  "  <-c sabitini x7 registerinden al NMC hafızasının 44. konumuna kaydet.
60. "fmac,x0,x4"  <-ACC <= u
61. "lw,x2,54  "  <-d sabitini NMC hafızasının 54. konumundan al ve x2 registerine kaydet.
62. "fmac,x2,x0"  <-ACC <= u + d
63. "getacc,x7 "  <-Akümülatördeki u + d değerini x7 registerine kaydet.
64. "sw,x7,46  "  <-Güncellenmiş u durum değerini x7 registerinden al ve NMC hafızasının 46. konumuna kaydet.
65. "return    "  <-Program akışını sonlandır.
```

Şekil 5.37 Izhikevich nöronu için nmc program akışı ve komutların yaptığı işlemler



Şekil 5.38 Ch izhikevich nöronunun membran gerilimi (v) ve membran toparlanma hızı (u) durumlarının nmc tarafından hesaplanmasına ilişkin test görüntüsü

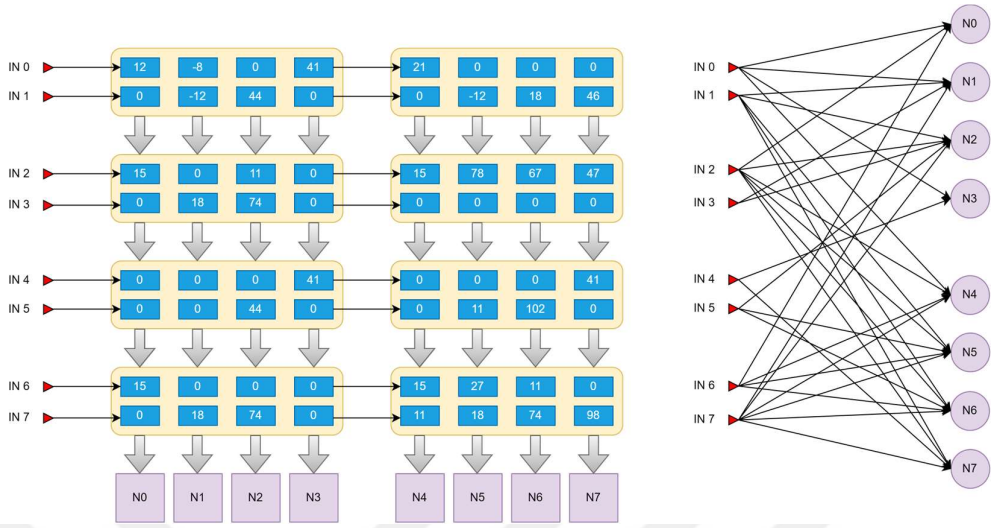
Şekil 5.38'deki test görüntüsü 200 MHz saat frekansında çalışan bir NMC ünitesinin test sonucudur. NMC ünitesi Şekil 5.35'de verilen program akışını hesaplamayı 3.556 µs sürede bitirmektedir. 5.37'de verilen test görüntüsü toplamda 4096 adet örnek içermekte olup, adım aralığı 0.1 ms olarak belirlenmiştir. Şekil 5.37 Izhikevich nöronunun 409.6 ms süre boyuncaki aktivitesini göstermektedir. Bütün hesaplamalar 14.476 ms sürede tamamlanmıştır. NMC ünitesi 200 MHz saat frekansında gerçek zamanlı bir nörona kıyasla 28 kat daha hızlı çalışmaktadır. NMC ünitesinin program akışını tamamlama süreci program akışının uzunluğuna, dolayısıyla nöron modelinin doğrudan kendisine bağlıdır.

5.8 Sinaptik Plastisitenin Sağlanması

SNN ağlarının biyolojik sinir ağlarından ilham alması SNN ağlarını çalıştıracak herhangi bir nöromorfik donanımın nöronlar arasında çok çeşitli bağlantı topolojilerinin desteklenmesini gerektirmektedir.

5.8.1 Sıfır Değerli Ağırlıklar

Tasarlanan donanımda XBAR ünitelerinin 2x4'ten daha küçük parçalara ayrılamaması sinaptik plastisitenin tam olarak sağlanabilmesini olumsuz etkilemektedir. Bu olumsuzluğu giderebilmek için nöronların ağırlıkları arasına seyrek bağlantıları da sağlayabilmek açısından sıfır değerli ağırlıklar enjekte edilmektedir. Sıfır değerli ağırlıkların enjekte edilmesi ile sinaptik plastisitenin sağlanmasına ilişkin bir görsel Şekil 5.39'da verilmiştir.



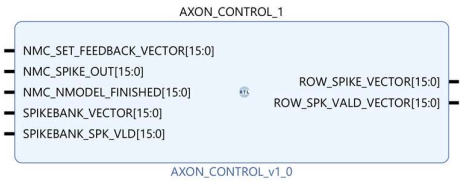
Şekil 5.39 Sıfır değerli ağırlıkların enjeksiyonu ile sinaptik plastisitenin sağlanması

Bölüm 5.2.4'te anlatıldığı üzere öğrenme motorları olan ULEARN üniteleri eğitim sürecinde sıfır değerli ağırlıkları dikkate almamaktadır.

5.8.2 Geri Beslemeli Nöronlar için AXON CONTROL Ünitesi

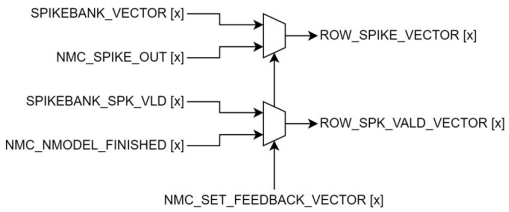
Donanımın geri beslemeli nöronları çalıştırabilmesi için NMC ünitelerinin atım gönderdiği çıkışlar olan NMC_SPK_OUT çıkışları HYPERCOLUMN ünitelerinin atım girişlerine bağlanmalıdır. NMC – HYPERCOLUMN atım bağlantısını yönetmek için AXON CONTROL adında bir ünite donanıma yerleştirilmiştir.

16 atım girişli bir HYPERCOLUMN ünitesinin atım girişleri ile 16 adet NMC ünitesinin atım çıkışlarını birbirine bir AXON CONTROL ünitesine ait RTL modül görüntüsü Şekil 5.40'ta verilmiştir.



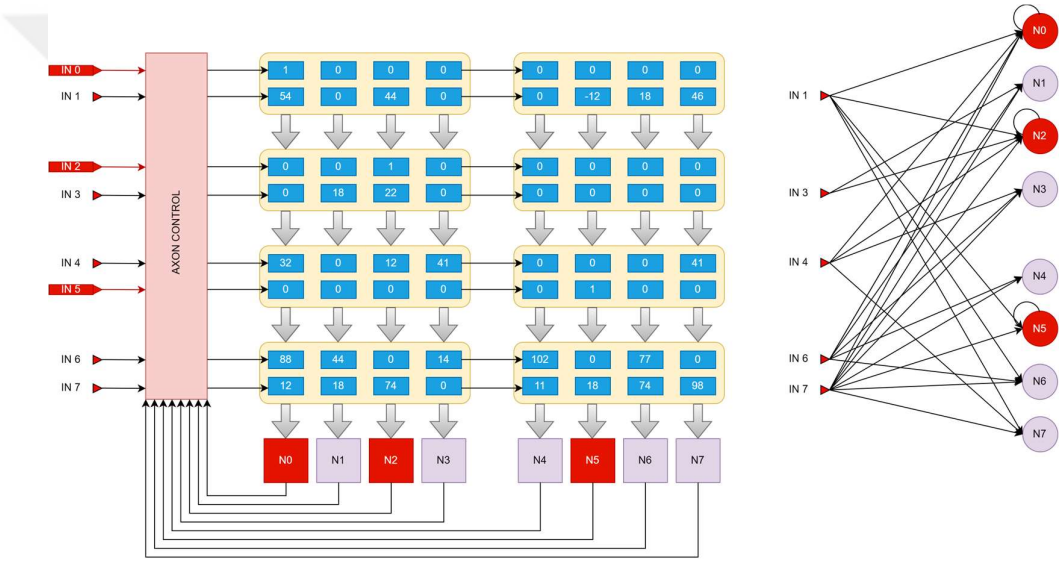
Şekil 5.40 16 girişli axon control ünitesi

AXON CONTROL ünitesi sadece seçici devrelerden oluşmaktadır. AXON CONTROL ünitesinin bir adet NMC için içerdiği seçici devresini gösteren blok diyagram Şekil 5.41'de görülmektedir.



Şekil 5.41 Bir adet nmc için axon control devresi

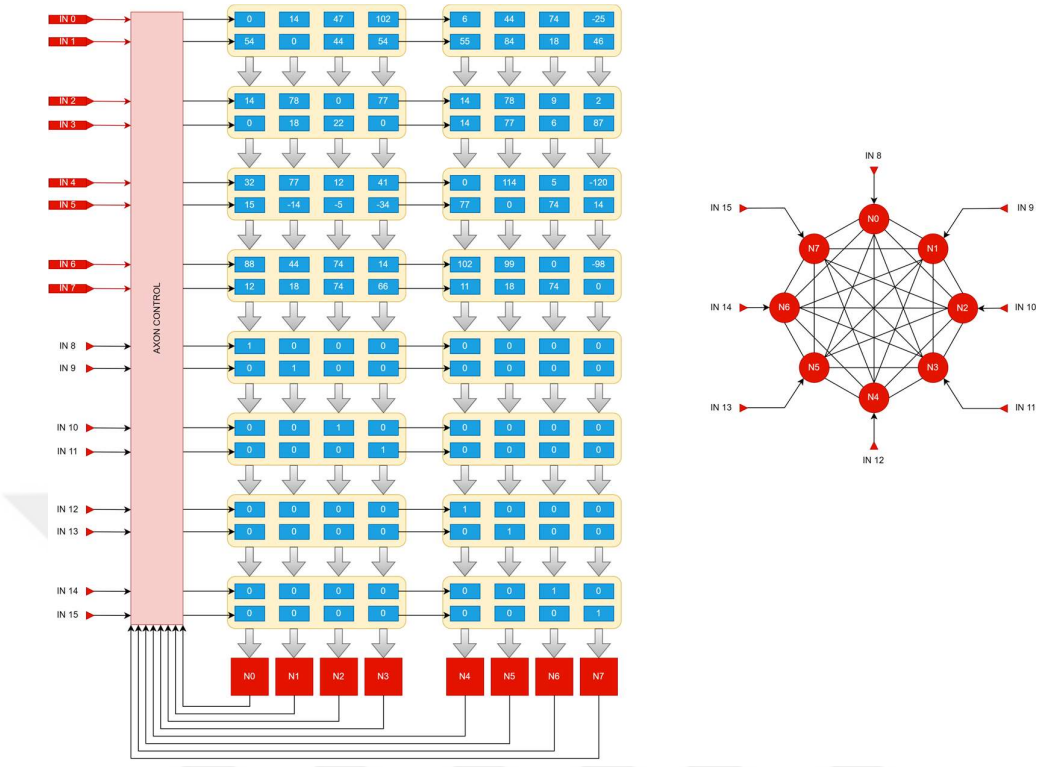
Geri beslemeli nöronlar oluşturulurken sıfır değerli ağırlıklar da kullanılır. Şekil 5.42'de sadece 3 adet geri beslemeli nöron içeren ve seyrek bağlantılı bir SNN ağının AXON CONTROL ünitesi ile donanıma haritalanması temsil edilmiştir.



Şekil 5.42 Axon control ünitesi ile geri beslemeli nöronlara ve seyrek bağlantıya sahip bir ağın donanıma haritalanması

Şekil 5.41'de geri beslemeye sahip N0, N2 ve N5 nöronlarının HYPERCOLUMN ünitelerine bağlanması nedeniyle IN 0, IN 2 ve IN 5 girişlerinden gelen atımlar NMC ünitelerinden, IN 1, IN 3, IN 4, IN 6 ve IN 7 girişlerinden gelen atımlar SPIKE BUFFER ünitesinden gelmektedir. Kırmızı renkli nöronlar geri beslemeye sahip nöronlardır.

AXON CONTROL ünitesi sadece geri beslemeli nöronların değil, farklı bağlantı türlerine sahip ağların da donanıma haritalanabilmesini mümkün kılmaktadır. Şekil 5.43'de 8 nöronlu bir Hopfield ağının AXON CONTROL ile donanıma haritalanması gösterilmektedir.

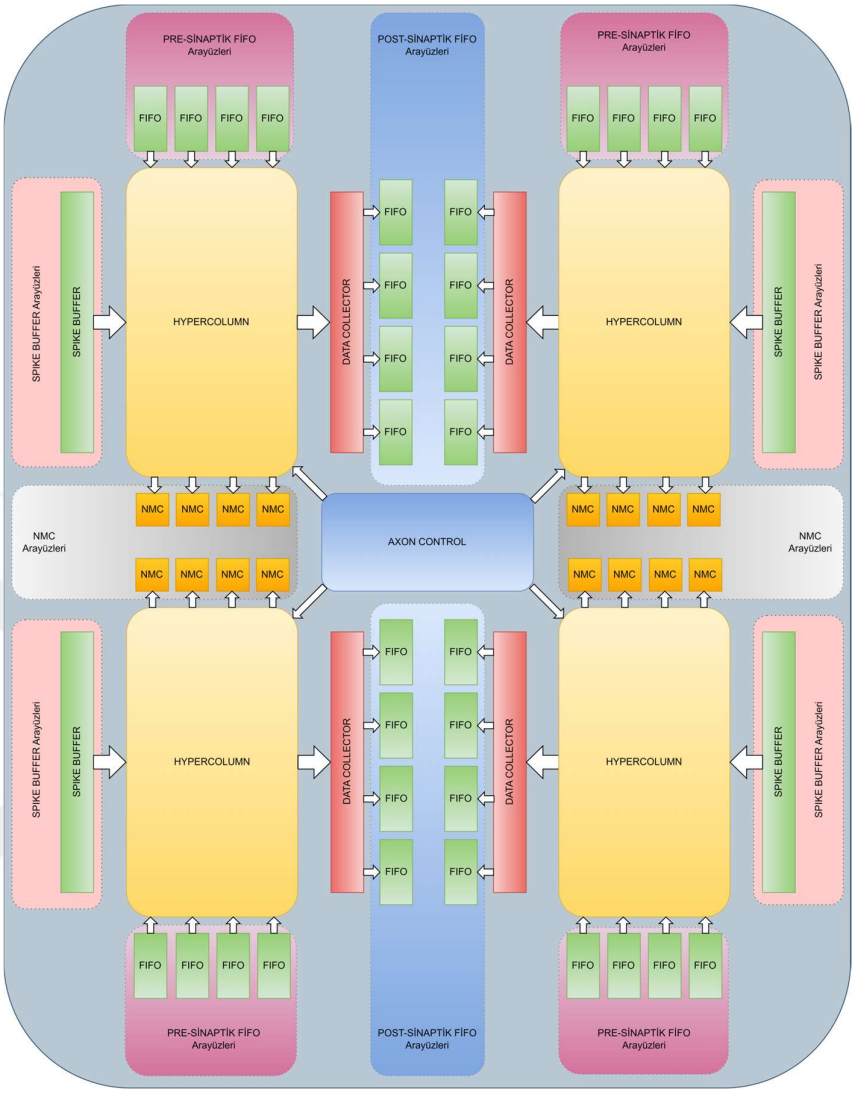


Şekil 5.43 8 Nöronlu bir hopfield ağının axon control ünitesi ile donanıma haritalanması

5.9 Hedef Platformda Donanım Sentezi

Önceki bölümlerde donanımın temel yapı taşları ve çalışma prensipleri ele alınmıştır. Donanımın ne kadar NMC, HYPERCOLUMN, PROCESSING ELEMENT, ULEARN ve FIFO içereceği uygulamanın ihtiyaçlarına göre değişmektedir.

Bu bölümden sonra işlenecek olan donanımın çekirdek kısmı, her biri 8 adet PROCESSING ELEMENT bulunduran 4 adet HYPERCOLUMN, HYPERCOLUMN başına 1 tane olmak üzere 4 adet SPIKE BUFFER, her HYPERCOLUMN ünitesi başına 1 tane olmak üzere toplam 4 adet DATA COLLECTOR, her HYPERCOLUMN başına 4 adet olmak üzere toplam 16 NMC ve her NMC başına 1 adet Pre-Sinaptik, bir adet de Post-Sinaptik adında ağırlıkları depolamak ve işledikten sonra dış dünyaya vermek için kullanılan, toplamda 32 adet FIFO, geri beslemeli ağlara uyum sağlamak için de 1 adet AXON CONTROL ünitesi içermektedir. Donanımın çekirdek kısmına ait mimari blok diyagramı Şekil 5.44'te verilmiştir.



Şekil 5.44 Çekirdek donanım blok diyagramı

Şekil 5.41, sadece donanımın çekirdek kısmını oluşturmakta olup, dış dünya ile ve hedef platformun CPU'su ile haberleşmesini sağlayacak AXI arayüzleri dahil edilmemiştir. AXI arayüzlerinin Bölüm 5'e dahil edilmemesinin nedeni donanımın tasarımının sabit miktarda kaynak hedeflemeyip, herhangi bir SoC veya FPGA donanımında farklı sayıda alt modüllerle sentezlenip kullanabilmesine olanak tanıyan bir tasarım yaklaşımını sergilemektir. AXI arayüzleri 6. Bölüm'de, Şekil 5.41'de blok diyagramı verilen donanım için tasarlanacaktır. Farklı sayıda alt modül içeren donanımlar için AXI arayüzleri değişkenlik gösterebilir.

Şekil 5.41'de blok diyagramı verilen donanımın içerdiği modül ve kaynak sayıları Tablo 5.8'de verilmiştir. Modüllerin barındırdıkları FIFO ve RAM'lerin derinlikleri ve genişlikleri Tablo 5.9'da yer almaktadır.

Tablo 5.9 Çekirdek donanımın modül sayıları

Donanım Kaynakları	**Miktar**
Sistolik Dizi	16x4x4
NMC	16
ULEARN	32
PROCESSING ELEMENT	32
HYPERCOLUMN	4
DATA COLLECTOR	4
AXON CONTROL	1
PRE-SİNAPTİK FIFO	16
POST-SİNAPTİK FIFO	16
SPIKE BUFFER	4

Tablo 5.10 Çekirdek donanımın modüllerinin fifo ve ram bilgileri

FIFO ve RAM Türleri	**Derinlik**	**Genişlik (Bit)**
PRE-SİNAPTİK FIFO	4096	16
POST-SİNAPTİK FIFO	4096	16
SPIKE BUFFER FIFO	2048	16
PROCESSING ELEMENT FIFO	512	64
NMC RAM	1024	16

6
DONANIM ARAYÜZLERİ

6.1 Pre-Sinaptik AXI Stream Arayüzü

Pre-Sinaptik AXI Stream arayüzü nöron ağırlıklarının CPU'dan çekirdek donanıma gönderildiği arayüzdür. Pre-Sinaptik AXI Stream arayüzü Şekil 5.43'de gösterilen Pre-Sinaptik FIFO Arayüzlerine bağlıdır ve CPU'dan gelen sinaptik verileri Pre-Sinaptik FIFO ünitelerine depolar.

Pre-Sinaptik AXI Stream arayüzü 64-Bit genişliğindedir. Her bir Pre-Sinaptik FIFO 16-Bit genişliğinde olduğundan ötürü gelen 64-Bit uzunluğundaki AXI stream verisi 4 FIFO'ya gönderilecek veriyi içermektedir. Dolayısıyla aynı anda en fazla 4 FIFO'ya veri gönderilebilmektedir. Pre-Sinaptik AXI Stream arayüzü

6.2 Atım AXI Stream Arayüzü

Atım AXI Stream arayüzü nöronlara verilecek atımların SPIKE BUFFER ünitelerine depolamakla görevli olan arayüzdür. 64-Bit genişliğinde olan Atım AXI Stream arayüzü 16-Bit genişliğinde FIFO içeren SPIKE BUFFER ünitelerinin hepsine aynı anda atım gönderebilmektedir.

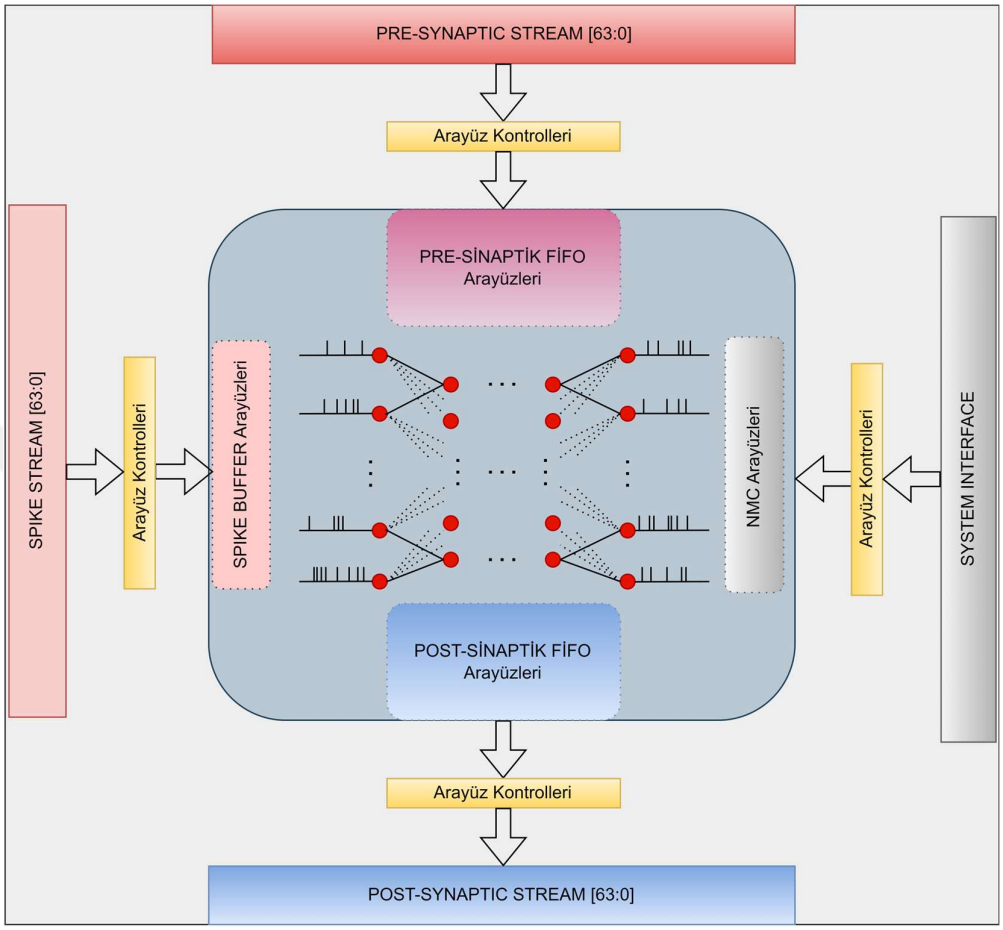
6.3 Post-Sinaptik AXI Stream Arayüzü

Post-Sinaptik AXI Stream arayüzü öğrenme işleminden sonra Post-Sinaptik FIFO'larda depolanmış ağırlıkların CPU'ya geri gönderilmesine görevli olan arayüzdür. 64-Bit genişliğinde veri göndermektedir. Bu nedenle aynı anda en fazla 4 adet Post-Sinaptik FIFO'dan veri çekebilmektedir.

6.4 Sistem AXI4-Lite Arayüzü

Sistem AXI4-Lite donanımın programlanması ve diğer ara yüzlerin veri yazdığı veya aldığı modülleri yöneten arayüzdür. Şekil 5.43'de blok diyagramı verilen çekirdek donanımın yönetimi için Sistem AXI4-Lite arayüzünde 62 adet register tanımlanmıştır. Register haritası ve fonksiyonları 7. Bölümde incelenmiştir.

Bütün arayüzlerin dahil edildiği blok diyagram Şekil 6.1'te verilmiştir.



Şekil 6.1 Çekirdek donanımın veri transfer ve kontrol arayüzleri ile birleştirilmesini gösteren blok diyagram

7
DONANIMIN PROGRAMLANMASI

7.1 Register Haritası ve Register İşlevleri

Donanımın programlanabilmesi için bir önceki bölümde Şekil 6.1'de gösterilen blok diyagramda yer alan arayüzler kullanılmaktadır. Donanımı programlamak için kullanılan tüm girişler Sistem AXI4-Lite arayüzüne bağlanmış olup, söz konusu arayüz Atım, Pre-Sinaptik ve Post-Sinaptik AXI Stream arayüzlerinin de veri yönlendirmelerini kontrol etmektedir.

Hedef platform Zynq-7000 SoC'de AXI4-Lite arayüzünde en az 32-Bit genişliğinde register tanımlanabilmektedir. Şekil 6.1'de blok diyagramı gösterilen çekirdek donanımın 62 adet registeri vardır. Register isimleri, adresleri ve işlevleri Tablo 7.1'de ele alınmıştır. Tablonun sütunlarındaki R/W ifadesi, registerin yazılabilir ve okunabilir olup olmadığını göstermektedir. RW yazılabilir ve okunabilir, R ise sadece okunabilir anlamına gelmektedir. Her register 32-Bit uzunluğundadır.

Tablo 7.1 Register haritası ve işlevleri

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
PRE_SYNAPTIC_STREAM_ROUTER	RW	"E0"	Pre-Sinaptik AXI Stream verilerini Pre-Sinaptik FIFO'lara yönlendirir.
POST_SYNAPTIC_STREAM_ROUTER	RW	"E4"	Post-Sinaptik FIFO çıkışlarını Post-Sinaptik AXI Stream arayüzüne yönlendirir.
PRE_SYNAPTIC_STREAM_STROBE	RW	"E8"	Pre-Sinaptik FIFO'lara veri yazımını kontrol eder.
POST_SYNAPTIC_STREAM_STROBE	RW	"EC"	Post-Sinaptik FIFO'lardan veri çekilmesini kontrol eder.

Tablo 7.1 Register haritası ve işlevleri (devamı)

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
POST_SYNAPTIC_STREAM_DATACOUNT	RW	“F4”	Post-Sinaptik FIFO’lardan kaç tane veri çekileceğini kontrol eder.
SPIKE_STREAM_STROBE	RW	“F0”	Atımların SPIKE BUFFER ünitelerine gönderimini kontrol eder.
POST_SYN_FIFO_RST	RW	“00”	Post-Sinaptik FIFO’ları resetler.
HYPERCOLUMN_RST	RW	“04”	HYPERCOLUMN ünitelerini resetler.
NMC_STATE_RST	RW	“08”	NMC ünitelerini resetler.
NMC_HARD_RST	RW	“0C”	NMC ünitelerinin hafızasını resetler.
PRE_SYN_FIFO_RST	RW	“10”	Pre-Sinaptik FIFO’ları resetler.
DC_RST_VECTOR	RW	“14”	DATA COLLECTOR ünitelerini resetler.
SPIKE_BUFFER_RST_VECTOR	RW	“18”	SPIKE BUFFER ünitelerini resetler.
POST_SYN_FIFO_EMPTY	R	“1C”	Post-Sinaptik FIFO’ların boş olduğunu bildirir.
POST_SYN_FIFO_FULL	R	“20”	Post-Sinaptik FIFO’ların dolu olduğunu bildirir.
POST_SYN_FIFO_WRERR	R	“24”	Post-Sinaptik FIFO’lara veri yazımının başarısız olduğunu bildirir.
POST_SYN_FIFO_RDERR	R	“28”	Post-Sinaptik FIFO’lardan veri çekilmesinin başarısız olduğunu bildirir.
PRE_SYN_FIFO_EMPTY	R	2C”	Pre-Sinaptik FIFO’ların boş olduğunu bildirir.

Tablo 7.1 Register haritası ve işlevleri (devamı)

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
PRE_SYN_FIFO_FULL	R	“30”	Pre-Sinaptik FIFO’ların dolu olduğunu bildirir.
PRE_SYN_FIFO_WRERR	R	“34”	Pre-Sinaptik FIFO’lara veri yazımının başarısız olduğunu bildirir.
PRE_SYN_FIFO_RDERR	R	“38”	Pre-Sinaptik FIFO’lardan veri çekilmesinin başarısız olduğunu bildirir.
SPIKE_BUFFER_FULL	R	“3C”	SPIKE BUFFER ünitelerininin dolu olduğunu bildirir.
SPIKE_BUFFER_EMPTY	R	“40”	SPIKE BUFFER ünitelerininin boş olduğunu bildirir.
NMC _SPIKE_FLAGS	R	“44”	Nöron modelinin ateşlendiğini bildirir.
NMC_MEM_VIOLATION	R	“48”	Nöron modelinin çalıştırılmasında bir program hatası gerçekleştiğini bildirir.
NMC_MATH_ERROR	R	“4C”	Nöron modelinin çalıştırılmasında bir aritmetik hata gerçekleştiğini bildirir.
NMC_FINISHED	R	“50”	Nöron modelinin hesaplanmasının tamamlandığını bildirir.
NMC_XN_LOW	RW	“54”	NMC ünitelerinin veri hafızasının başlangıç adresidir.
NMC_XN_HIGH	RW	“58”	NMC ünitelerinin veri hafızasının bitiş adresidir.
NMC _COLD_START	RW	“5C”	NMC ünitelerini çalıştırarak nöron modelinin hesaplanmasını başlatır.

Tablo 7.1 Register haritası ve işlevleri (devamı)

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
HYPERCOLUMN_0_INFERENCE_CYCLES	RW	“60”	HYPERCOLUMN_0 ünitesinin girişim döngüsüdür.
HYPERCOLUMN_1_INFERENCE_CYCLES	RW	“64”	HYPERCOLUMN_1 ünitesinin girişim döngüsüdür.
HYPERCOLUMN_2_INFERENCE_CYCLES	RW	“68”	HYPERCOLUMN_2 ünitesinin girişim döngüsüdür.
HYPERCOLUMN_3_INFERENCE_CYCLES	RW	“6C”	HYPERCOLUMN_3 ünitesinin girişim döngüsüdür.
HYPERCOLUMN_INFERENCE_START	RW	“70”	Atım-ağırlık girişimini başlatır.
HYPERCOLUMN_INFERENCE_DONE	R	“74”	Atım-ağırlık girişiminin bittiğini bildirir.
HYPERCOLUMN_LEARNING_START	RW	“78”	Sinaptik ağırlıkların eğitimi işlemini başlatır.
HYPERCOLUMN_LEARNING_DONE	R	“7C”	Sinaptik ağırlıkların eğitimi işleminin bittiğini bildirir.
AXON_CONTROLS_SET_FEEDBACK	RW	“80”	NMC ünitelerinin atım çıkışlarını HYPERCOLUMN atım girişlerine bağlayarak geri besleme oluşturur.
PRE_SYN_MUX	RW	“84”	AXI4-Lite arayüzündeki ilgili registerleri NMC Pre-Sinaptik arayüzlerine yönlendirir.
NMC_PRE_SYN_INTERFACE_CNTRL	RW	“88”	NMC ünitelerinin Pre-Sinaptik arayüzlerini aktif eder.
NMC_PROG_MEM_WENABLE	RW	“8C”	NMC ünitelerinin program hafızasına dışardan erişimi kontrol eder.

Tablo 7.1 Register haritası ve işlevleri (devamı)

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
NMC_XX_NEURON_ID	RW	“90”	NMC ünitesinde çalışan nöronun kimlik bilgisidir.
NMC_XX_LAST_SPIKE_TIME	RW	“94”	Nöronun son ateşlenme zamanıdır.
NMC_XX_SYN_QFACTOR	RW	“98”	Nöronun ağırlıklarının kuantalama faktörüdür.
NMC_XX_PF_LOW_ADDR	RW	“9C”	Nöronun program akışının başlangıç adresidir.
NMC_XX_NMODEL_REFRACTORY_DUR	RW	“A0”	Nöronun kalan inert periyodudur.
NMC_XX_NMODEL_NPARAM_DATA_ADDR	RW	“A4”	Nöron durum parametresidir ve hafızada yazılacağı adrestir
NMC_XX_NMODEL_ULEARN_LUT_DIN_ADDR	RW	“AC”	Nöronun öğrenme tablosunun verisi ve LUT adresidir.
NMC_XX_NMODEL_ULEARN_LUT_EN	RW	“B4”	ULEARN ünitelerinin tablolarının tutulduğu hafızaya veri yazılmasını kontrol eder.
NMC_XX_NMODEL_WMAX_WMIN_LR	RW	“B8”	Nöronun ağırlıklarının alabileceği en yüksek değer, en küçük değer ve nöronun öğrenme oranıdır.
POST_SYN_MUX	RW	“BC”	NMC Post-Sinaptik çıkışlarını ilgili AXI4-Lite registerlerine yönlendirir.

Tablo 7.1 Register haritası ve işlevleri (devamı)

Register İsmi	**R/W**	**Adres (Hex)**	**İşlevi**
NMC_POST_SYN_INTERFACE_CNTRL	RW	"C0"	NMC ünitelerinin Post-Sinaptik arayüzlerini aktif eder.
NMC_XX_R_NMODEL_NEURON_ID	R	"C4"	NMC ünitesinde çalışan nöronun kimlik bilgisidir.
NMC_XX_R_NNMODEL_NEW_SPIKE_TIME	R	"C8"	Nöronun güncellenmiş son ateşlenme zamanıdır.
NMC_XX_R_NMODEL_SYN_QFACTOR	R	"CC"	Nöronun ağırlıklarının kuantalama faktörüdür.
NMC_XX_R_NMODEL_PF_LOW_ADDR	R	"D0"	Nöronun program akışının başlangıç adresidir.
NMC_XX_R_NMODEL_REFRACTORY_DUR	R	"D4"	Nöronun güncellenmiş kalan inert periyodudur.
NMC_XX_R_NMODEL_NPARAM_DATAOUT	R	"D8"	Nöronun güncellenmiş durum parametresidir.
NMC_XX_REDIST_NMODEL_DADDR	RW	"DC"	Nöronun güncellenmiş durum parametresinin NMC hafızasından çekileceği adrestir.

7.2 Nöron Program Akışlarının NMC Ünitelerine Yüklenmesi

NMC ünitelerine program akışlarının gönderilmesi için NMC ünitelerinin Pre-Sinaptik arayüzleri aktif edilmelidir. NMC_PRE_SYN_INTERFACE_CNTRL registeri NMC ünitelerinin Pre-Sinaptik arayüzlerini kontrol etmektedir. NMC_PRE_SYN_INTERFACE_CNTRL registerinin hangi bitlerinin hangi NMC ünitelerinin Pre-Sinaptik Arayüzünü aktif ettiği Tablo 7.2'de verilmiştir.

Tablo 7.2 Nmc_pre_syn_interface_cntrl bitlerinin pre-sinaptik arayüzünü aktif ettiği nmc üniteleri

Register Bitleri	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_PRE_SYN_INTERFACE_CNTRL [15:8]	NMC 15	NMC 14	NMC 13	NMC 12	NMC 11	NMC 10	NMC 09	NMC 08
Register Bitleri	7	6	5	4	3	2	1	0
NMC_PRE_SYN_INTERFACE_CNTRL [7:0]	NMC 07	NMC 06	NMC 05	NMC 04	NMC 03	NMC 02	NMC 01	NMC 00

NMC Pre-Sinaptik Arayüzlerinin aktif edilmesinden sonraki adım program verilerinin yüklenebilmesi için NMC hafızasına yazma erişiminin açılmasıdır. NMC hafızasına dışardan erişebilmek ve veri yazabilmek için NMC_PROG_MEM_WENABLE [15:0] registeri kullanılmaktadır. Registerin bitlerinin durumuna göre hangi NMC ünitesinin hafızasına erişildiği Tablo 7.3'te gösterilmiştir.

Tablo 7.3 Nmc_prog_mem_wenable bitlerinin hafıza erişimini sağladığı nmc üniteleri

Register Bitleri	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_PROG_MEM_WENABLE [15:8]	NMC 15	NMC 14	NMC 13	NMC 12	NMC 11	NMC 10	NMC 09	NMC 08
Register Bitleri	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NMC_PROG_MEM_WENABLE [7:0]	NMC 07	NMC 06	NMC 05	NMC 04	NMC 03	NMC 02	NMC 01	NMC 00

NMC Pre-Sinaptik Arayüzlerinin aktif edilmesinden sonraki adım PRE_SYN_MUX [3:0] ile genel NMC registerleri olan NMC_XX ile gösterilen registerlerin programlanacak NMC ünitelerinin portlarına yönlendirilmesidir. PRE-SYN_MUX [3:0] ile NMC_XX registerlerinin hangi NMC Pre-Sinaptik arayüzüne bağlandığı her bir register için Tablo 7.4'te verilmiştir.

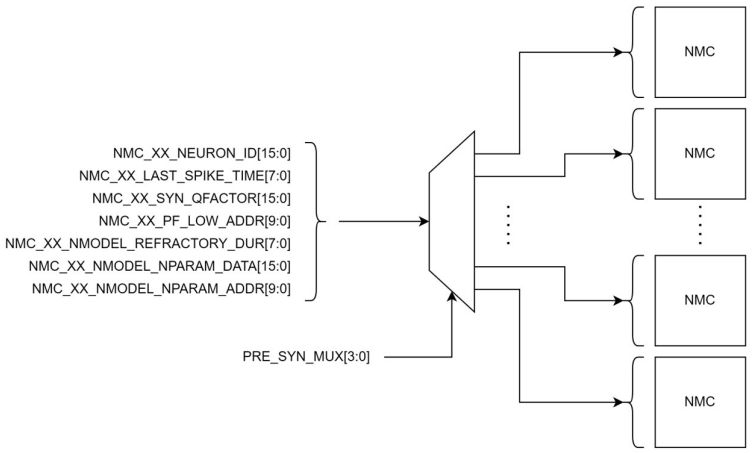
Tablo 7.4 Pre_syn_mux registerinin aldığı değere göre nmc_xx registerinin nmc ünitelerinin pre-sinaptik arayüzlerine yönlendirilmesi

PRE_SYN_MUX [3:0]	**X"0"**	**X"1"**	**X"2"**	**X"3"**	**X"4"**	**X"5"**	**X"6"**	**X"7"**
NMC_XX_NEURON_ID [23:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_LAST_SPIKE_TIME [7:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_SYN_QFACTOR [15:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_PF_LOW_ADDR [9:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_NMODEL_REFRACTORY_DUR [7:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_NMODEL_NPARAM_DATA [15:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_NMODEL_NPARAM_ADDR [9:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
PRE_SYN_MUX [3:0]	**X"8"**	**X"9"**	**X"A"**	**X"B"**	**X"C"**	**X"D"**	**X"E"**	**X"F**
NMC_XX_NEURON_ID [23:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_LAST_SPIKE_TIME [7:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_SYN_QFACTOR [15:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15

Tablo 7.4 Pre_syn_mux registerinin aldığı değere göre nmc_xx registerinin nmc ünitelerinin pre-sinaptik arayüzlerine yönlendirilmesi (devamı)

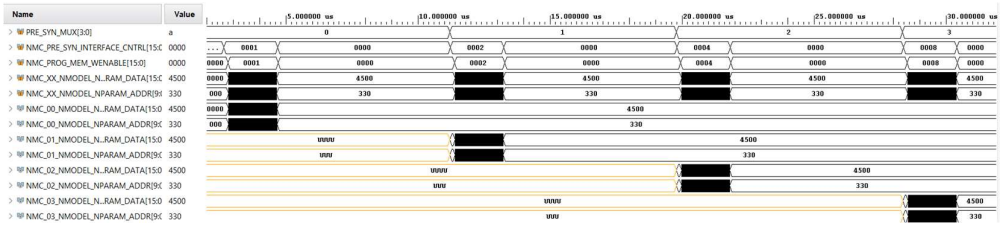
PRE_SYN_MUX [3:0]	**X"8"**	**X"9"**	**X"A"**	**X"B"**	**X"C"**	**X"D"**	**X"E"**	**X"F**
NMC_XX_PF_LOW_ ADDR [9:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_NMODEL_ REFRACTORY_DUR [7:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_NMODEL_ NPARAM_DATA [15:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_NMODEL_ NPARAM_ADDR [9:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15

Tablo 7.4'te gösterilen PRE_SYN_MUX ile NMC pre-sinaptik arayüzleri arasındaki ilişki Şekil 7.1'de görselleştirilmiştir.



Şekil 7.1 Pre_syn_mux registerinin nmc ünitelerinin pre-sinaptik arayüzlerine bağlanması

NMC ünitelerine nöron program akışları ve nöron parametreleri NMC_XX_NMODEL_NPARAM_DATA ve NMC_XX_NMODEL NPARAM_ADDR registerleri üzerinden transfer edilmektedir. NMC00, NMC01, NMC02 ve NMC03 ünitelerine program akışının gönderildiği örnek bir test görüntüsü Şekil 7.2'de gösterilmiştir.



Şekil 7.2 Nmc_xx_nmodel_nparam_data ve nmc_xx_nmodel_nparam_addr registerleri üzerinden nöron program akışı ve nöron parametrelerinin nmc00, nmc01, nmc02 ve nmc03 ünitelerine aktarılması

Nöron modellerine ait program akışları donanımın çalışma süresi boyunca sürekli değiştirilmesi gereken parametreler değildir. Nöron durumu sayısı ise herhangi bir SNN nöron modeli için 1-7 arasında değişmektedir. Bu kadar az veriyi donanıma göndermek için en uygun arayüz AXI4-Lite arayüzüdür.

7.3 Öğrenme Motorlarının Tablolarının Yüklenmesi

ULEARN modüllerine veri yazılabilmesi için ULEARN tablolarının aktif hale getirilmesi gerekmektedir. NMC_XX_NMODEL_ULEARN_LUT_EN [15:0] registeri ile tabloların yazıldığı hafıza kontrol edilmektedir. Tablo 7.5'te NMC_XX_NMODEL_ULEARN_LUT_EN [15:0] registerinin bitlerinin hangi HYPERCOLUMN ünitesinin hangi sütunundaki ULEARN öğrenme tablosuna veri yazdığı ortaya gösterilmiştir.

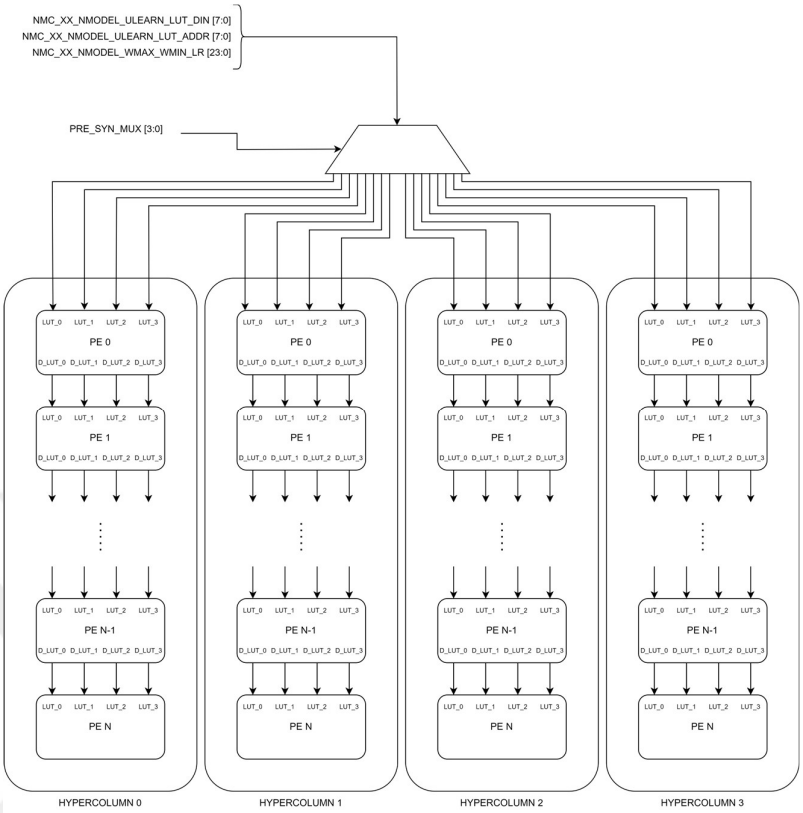
Tablo 7.5 Nmc_xx_nmodel_ulearn_lut_en registerinin bitleri hypercolumn sütunları arasındaki ilişki

Register Bitleri	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_XX_NMODEL_ULEARN_LUT_EN [15:8]	HP3_3	HP3_2	HP3_1	HP3_0	HP2_3	HP2_2	HP2_1	HP2_0
Register Bitleri	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NMC_XX_NMODEL_ULEARN_LUT_EN [7:0]	HP1_3	HP1_2	HP1_1	HP1_0	HP0_3	HP0_2	HP0_1	HP0_0

Öğrenme motoru programlama verileri HYPERCOLUMN ünitelerine iletilmektedir. Bu iletimi yine PRE_SYN_MUX registeri kontrol etmektedir. PRE_SYN_MUX [3:0] registerinin değerine göre ilgili registerlerin hangi HYPERCOLUMN ünitelerinin hangi sütunlarına yönlendirildiği Tablo 7.6'da gösterilmiştir. Tablodaki bilgiler blok diyagram olarak Şekil 7.3'te gösterilmiştir.

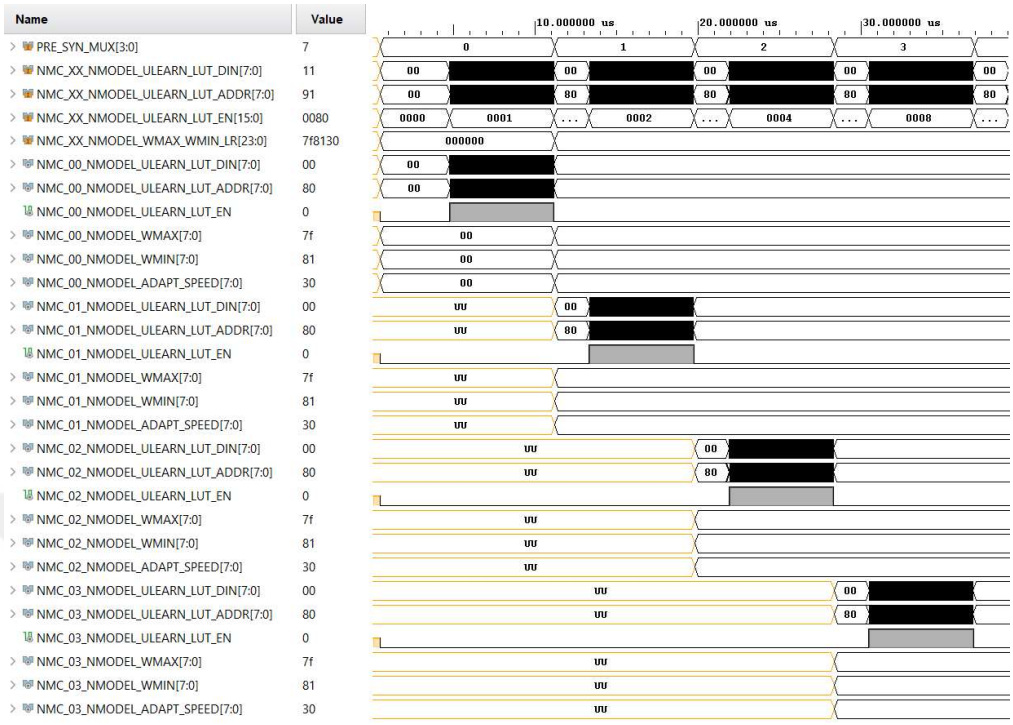
Tablo 7.6 Pre_syn_mux registerinin aldığı değere göre hypercolumn sütunlarına öğrenme motorlarını programlayan registerlerin yönlendirilmesi

PRE_SYN_MUX [3:0]	**X"0"**	**X"1"**	**X"2"**	**X"3"**	**X"4"**	**X"5"**	**X"6"**	**X"7"**
NMC_XX_NMODEL_ULEARN_LUT_DIN [7:0]	HP0_0	HP0_1	HP0_2	HP0_3	HP1_0	HP1_1	HP1_2	HP1_3
NMC_XX_NMODEL_ULEARN_LUT_ADDR [7:0]	HP0_0	HP0_1	HP0_2	HP0_3	HP1_0	HP1_1	HP1_2	HP1_3
NMC_XX_NMODEL_WMAX_WMIN_LR [23:0]	HP0_0	HP0_1	HP0_2	HP0_3	HP1_0	HP1_1	HP1_2	HP1_3
PRE_SYN_MUX [3:0]	**X"8"**	**X"9"**	**X"A"**	**X"B"**	**X"C"**	**X"D"**	**X"E"**	**X"F**
NMC_XX_NMODEL_ULEARN_LUT_DIN [7:0]	HP2_0	HP2_1	HP2_2	HP2_3	HP3_0	HP3_1	HP3_2	HP3_3
NMC_XX_NMODEL_ULEARN_LUT_ADDR [7:0]	HP2_0	HP2_1	HP2_2	HP2_3	HP3_0	HP3_1	HP3_2	HP3_3
NMC_XX_NMODEL_WMAX_WMIN_LR [23:0]	HP2_0	HP2_1	HP2_2	HP2_3	HP3_0	HP3_1	HP3_2	HP3_3



Şekil 7.3 Pre_syn_mux üzerinden ulearn ünitelerine öğrenme tabloları ve diğer verilerin ulaştırılması

Öğrenme motoru programlama verileri her seferde tek HYPERCOLUMN ünitesindeki tek bir sütuna veri yazılarak gerçekleştirilir. Her nöronun farklı bir öğrenme tablosu kullanabilmesine imkan tanımak için bu yöntem tercih edilmiştir. HYPERCOLUMN 0 ünitesinin içinde bulunan ve NMC00, NMC01, NMC02 ve NMC03 ünitelerine hizmet eden ULEARN ünitelerine öğrenme tablolarının gönderildiği bir test görüntüsü Şekil 7.4'te verilmiştir.



Şekil 7.4 Pre_syn_mux üzerinden nmc00, nmc 01, nmc 02 ve nmc 03 ünitelerine hizmet eden ulearn modüllerine öğrenme tabloları ve diğer verilerin ulaştırılması

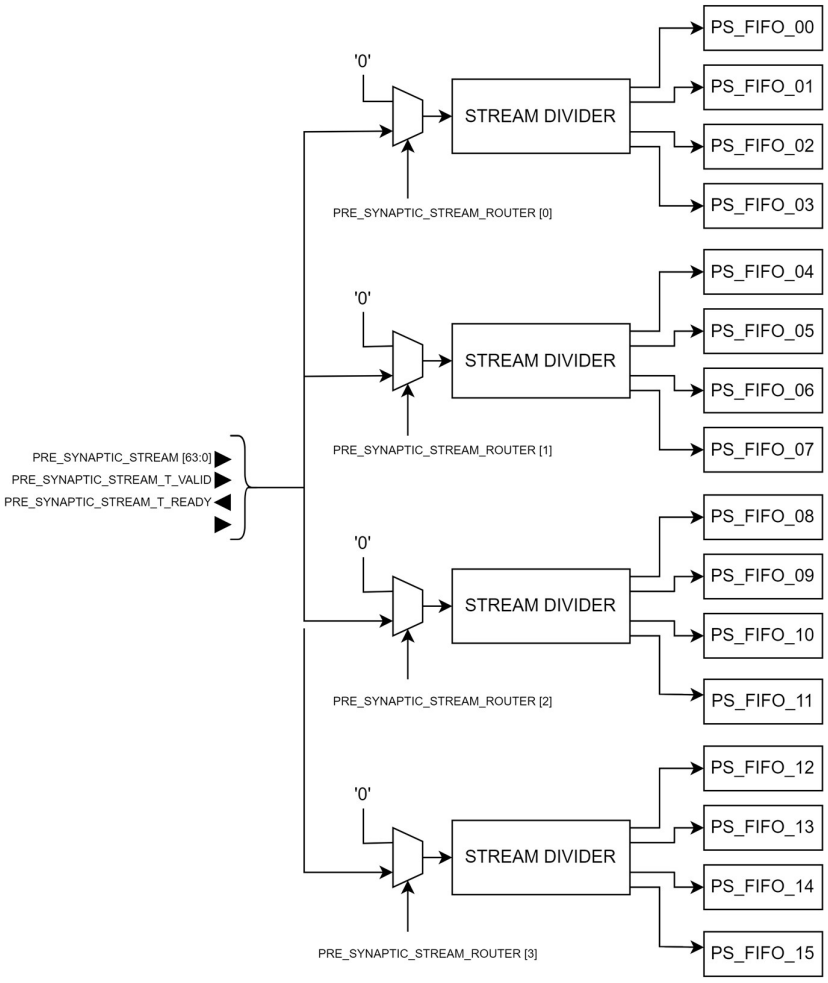
7.4 Sinaptik Veri Dağıtımı

Nöron ağırlıkları ve zaman bilgileri girişim işlemindens önce Pre-Sinaptik FIFO adı verilen 16-Bit genişliğinde FIFO'larda tutulmaktadır. FIFO'lara veri gönderimi Pre-Sinaptik AXI Stream arayüzünden yapılmaktadır. 64-Bit'lik AXI Stream verisi 16-Bitlik parçalara ayrılarak her seferde bir HYPERCOLUMN ünitesine bağlı en fazla 4 adet Pre-Sinaptik FIFO'ya eş zamanlı olarak sinaptik veri göndermekte kullanılmaktadır. Bu PRE_SYNAPTIC_STREAM_ROUTER [3:0] ve PRE_SYNAPTIC_STREAM_STROBE [3:0] görev almaktadır. İlgili registerlerin aldığı değerler ile hangi Pre-Sinaptik FIFO'lara değer gönderildiği Tablo 7.7'de gösterilmiştir.

Tablo 7.7 Pre_synaptic_stream_router [3:0] ve pre_synaptic_stream_strobe [3:0] registerlerinin aldığı değerlere göre axi stream verisinin yönlendirileceği pre-sinaptik fifolar

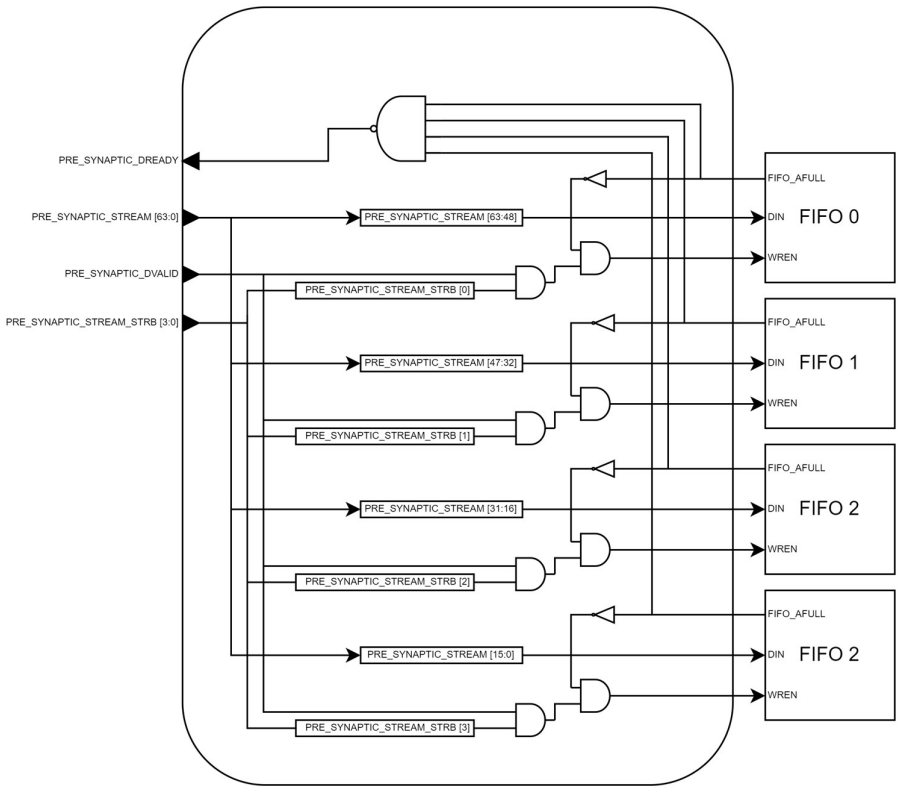
PRE_SYNAPTIC_STREAM_ ROUTER [3:0]	**"0000"**			
PRE_SYNAPTIC_STREAM_ STROBE [3:0]	**STROBE [3]**	**STROBE [2]**	**STROBE [1}**	**STROBE [0]**
	PS_FIFO_03	PS_FIFO_ 02	PS_FIFO_01	PS_FIFO_ 00
PRE_SYNAPTIC_STREAM_ ROUTER [3:0]	**"0010"**			
PRE_SYNAPTIC_STREAM_ STROBE [3:0]	**STROBE [3]**	**STROBE [2]**	**STROBE [1]**	**STROBE [0]**
	PS_FIFO_07	PS_FIFO_ 06	PS_FIFO_05	PS_FIFO_ 04
PRE_SYNAPTIC_STREAM_ ROUTER [3:0]	**"0100"**			
PRE_SYNAPTIC_STREAM_ STROBE [3:0]	**STROBE [3]**	**STROBE [2]**	**STROBE [1]**	**STROBE [0]**
	PS_FIFO_11	PS_FIFO_ 10	PS_FIFO_09	PS_FIFO_ 08
PRE_SYNAPTIC_STREAM_ ROUTER [3:0]	**"1000"**			
PRE_SYNAPTIC_STREAM_ STROBE [3:0]	**STROBE [3]**	**STROBE [2]**	**STROBE [1]**	**STROBE [0]**
	PS_FIFO_15	PS_FIFO_ 14	PS_FIFO_13	PS_FIFO_ 12

PRE_SYNAPTIC_STREAM_ROUTER [3:0] registeri kullanılarak gelen AXI Stream verisinin hangi HYPERCOLUMN ünitesine bağlı 4 Pre-Sinaptik FIFO'ya gönderileceği belirlenmektedir. PRE_SYNAPTIC_STREAM_STROBE [3:0] registeri ile de veri yazılmak istenen dört tane FIFO'dan hangilerinin gelen verileri alacağı belirlenerek herhangi bir FIFO'ya diğerlerinden bağımsız olarak veri yazma işlemi gerçekleştirilir. Sinaptik veri akışı kontrolünün görselleştirilmiş şekli Şekil 7.5'te verilmiştir.



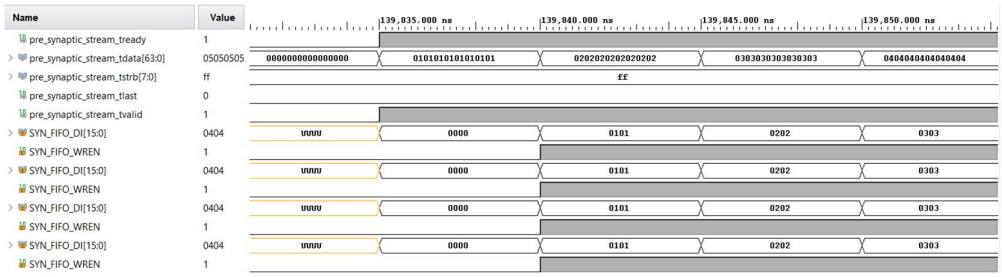
Şekil 7.5 Pre_synaptic_stream_router ile sinaptik veri akışının pre-sinaptik fifolara yönlendirilmesi

Şekil 7.4'te görülen 64-Bitlik AXI Stream verisini 16-Bit'lik parçalara ayırıp FIFO'lara gönderen STREAM DIVIDER devresinin blok diyagramı Şekil 7.6'da verilmiştir.



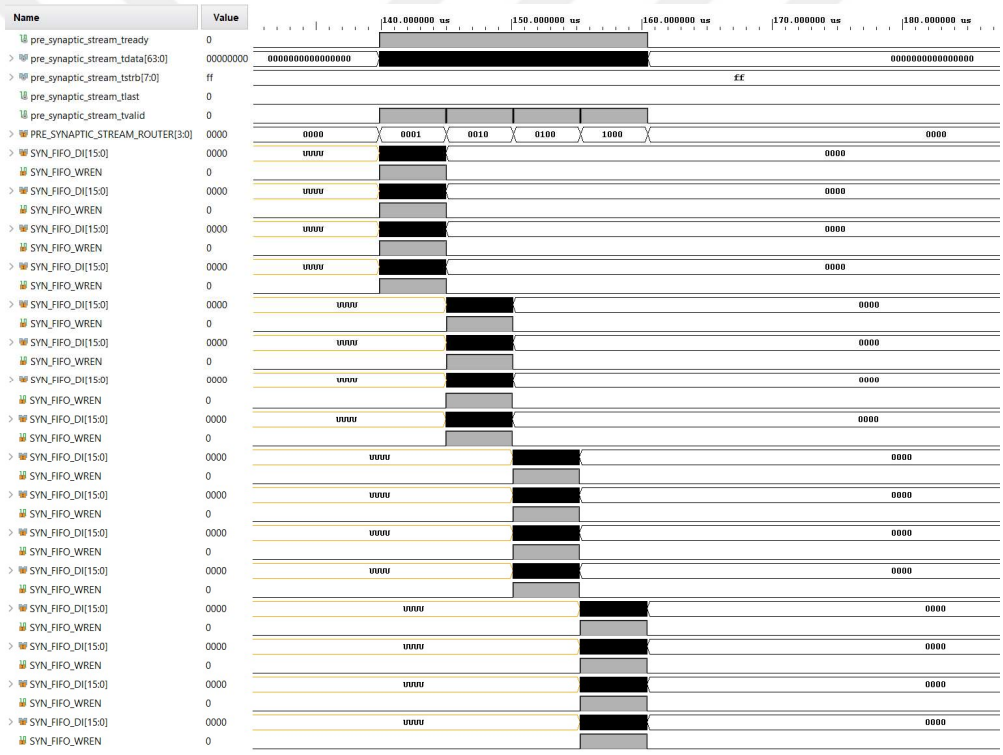
Şekil 7.6 Stream divider blok diyagramı

STREAM DIVIDER devresi kendisine bağlı olan herhangi bir FIFO'ya diğerlerinden bağımsız olarak veri yazılmasını mümkün kılmaktadır. Herhangi bir FIFO boş olduğu sürece PRE_SYNAPTIC_DREADY çıkışı '1' olmaktadır. PRE_SYNAPTIC_DREADY çıkışı AXI Stream arayüzünün TREADY çıkışına bağlanmıştır. Bu sayede FIFO boş olduğu sürece istenildiği zaman FIFO'ya AXI stream arayüzü üzerinden veri gönderilebilmektedir. PRE_SYNAPTIC_STREAM_STROBE [3:0] registeri gelen 64-Bit'lik AXI Stream verisinin 16-Bit'lik parçasının ilgili FIFO'lara yazılıp yazılmayacağı kontrol etmektedir. Veri yazılacak FIFO'nun dolu olması durumunda PRE_SYNAPTIC_STREAM_STROBE [3:0] değerinden bağımsız olarak FIFO'nun yazım kontrol girişi olan WREN sıfıra çekilir ve FIFO yazım işlemlerine kapatılır. STREAM DIVIDER devresinin gelen 64-Bitlik veriyi nasıl böldüğü Şekil 7.7'de görülmektedir.



Şekil 7.7 Stream divider ile gelen 64-bitlik sinaptik verinin 4 adet fifoya eş zamanlı yazılması

16 adet Pre-Sinaptik FIFO'ya veri yazılmasını gösteren süreç Şekil 7.8'de test görüntüsü olarak verilmiştir.

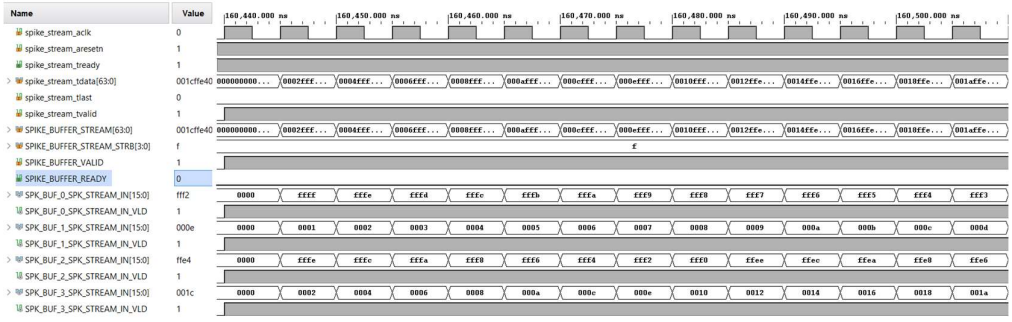


Şekil 7.8 16 adet pre-sinaptik fifoya axi stream üzerinden veri gönderilmesi

7.5 Atımların Donanıma Yüklenmesi

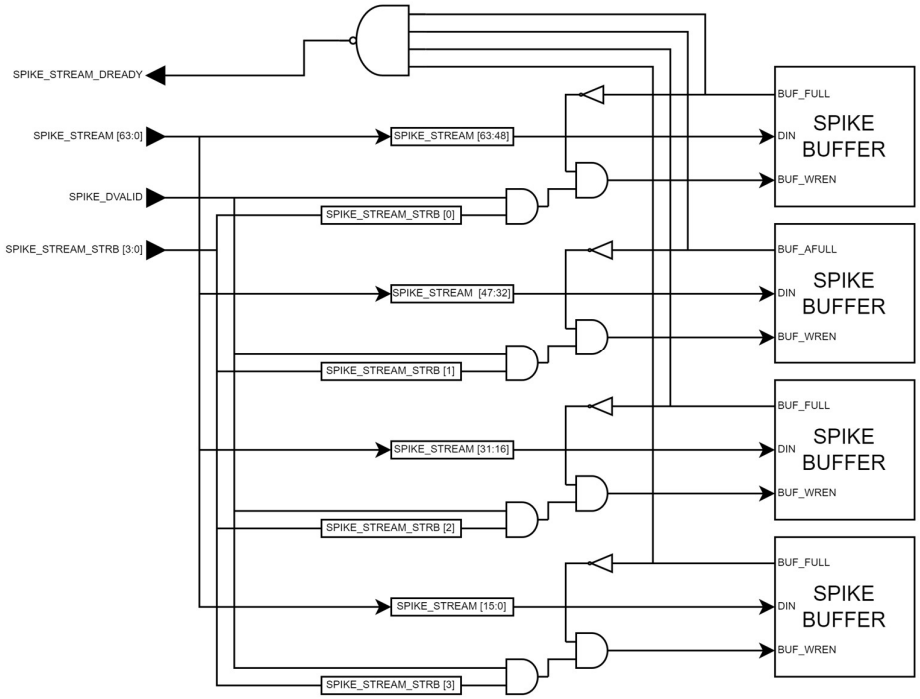
Atımların donanıma yüklenmesi süreci 7.4'te anlatılan Sinaptik Veri Dağıtımı süreci ile temel olarak aynı şekilde işlemektedir. Gelen 64-Bit'lik AXI Stream verisi HYPERCOLUMN ünitelerine bağlı SPIKE BUFFER ünitelerine 16-Bit'lik parçalara bölünerek eş zamanlı olarak gönderilir. Donanımda 4 adet SPIKE BUFFER bulunduğu için bütün ünitelere aynı anda veri yazılması mümkündür. Aynı anda aynı atımların yazılabilmesi 4 ayrı 16x4'lük HYPERCOLUMN ünitesinin istenildiği takdirde 16x16 boyutunda 1 adet, 16x8 boyutunda 2 adet veya 16x4 boyutunda 4 adet ayrı sistolik dizi olarak kullanılabilmesini sağlamaktadır.

Atımların donanıma yüklenmesi Atım AXI Stream arayüzü ile gerçekleştirilmektedir. AXI Stream verisinin kontrolünde SPIKE_STREAM_STROBE [3:0] registeri görev almaktadır. Register bitlerinin değerine göre 64-Bit'lik verinin ilgili parçası SPIKE BUFFER ünitesine gönderilir. Bu sürece ait bir test görüntüsü Şekil 7.9'da verilmiştir.



Şekil 7.9 Atımların donanıma gönderilmesine ait bir test görüntüsü

Atımların donanıma yüklenmesini gerçekleştiren Atım AXI Stream arayüzü ile SPIKE BUFFER üniteleri arasındaki kontrol devresinin blok diyagramı Şekil 7.10'da görülmektedir. SPIKE_STREAM_STRB [3:0] registeri ile gelen 64-Bitlik verilerin ilgili 16-Bit'lik parçasının SPIKE BUFFER ünitesine yazılıp yazılmayacağı kontrol edilir her bir SPIKE BUFFER ünitesine diğer ünitelerden bağımsız olarak veri yazılabilir.



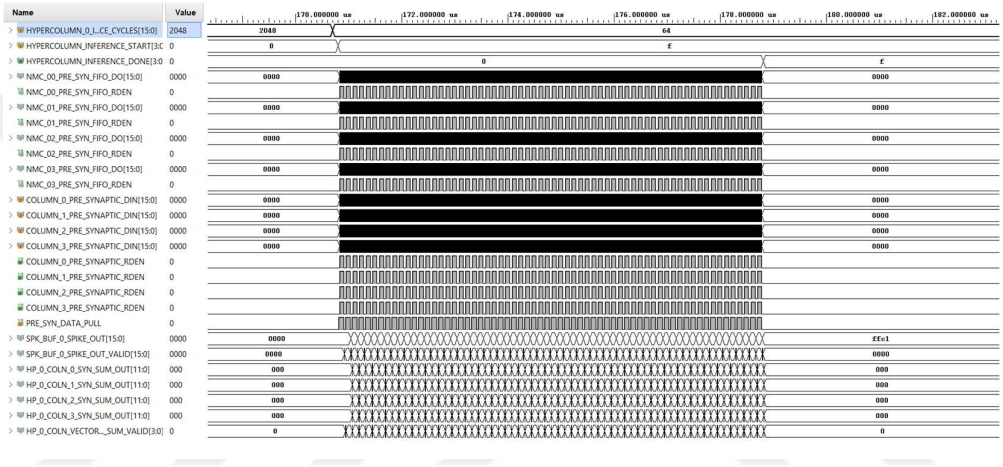
Şekil 7.10 Atımların donanıma gönderilmesinin temsili görseli

7.6 Girişim Döngüsünün Belirlenmesi

Girişim döngüsü, HYPERCOLUMN ünitelerinin Pre-Sinaptik FIFO'lardan sinaptik verileri ve o sinaptik verilere karşılık gelen atım değerlerinin SPIKE BUFFER ünitelerinden çekildiği ve PROCESSING ELEMENT ünitelerinde işleme sokulduğu süreçtir. Çekirdek donanımda HYPERCOLUMN başına 8 adet PROCESSING ELEMENT düştüğü için her girişim döngüsünde FIFO'lardan 16 adet sinaptik veri ve SPIKE BUFFER ünitesinden 16-Bit'lik sadece bir atım vektörü çekilip işleme alınmaktadır. Bu sürecin kaç kere tekrarlanacağı nöronun toplam sinaptik veri sayısının 16'ya bölümünden elde edilmektedir. Girişim döngüsünün alabileceği en büyük değer 65536 olarak atanmıştır.

7.7 Girişim Sürecinin Başlatılması

Girişim süreci HYPERCOLUMN_INFERENCE_START [3:0] registeri ile kontrol edilmektedir. HYPERCOLUMN_INFERENCE_START [3:0] registerinin bitlerinin ‘1’ olması durumunda girişim işlemi ilgili HYPERCOLUMN ünitesinde başlatılır. 1024 adet ağırlık için 1024/16=64 girişim döngüsünün bir HYPERCOLUMN ünitesinde gerçekleştirilmesine ilişkin test görüntüsü Şekil 7.11’de verilmiştir.



Şekil 7.11 1024 adet ağırlığın 64 girişim döngüsü ile atımlarla girişime tabi tutulması

Girişim süreci tamamlandığı zaman HYPERCOLUMN_INFERENCE_DONE [3:0] registerinin bitleri ilgili HYPERCOLUMN ünitesi için ‘1’ olmaktadır. Bu registerin değeri okunarak girişim işleminin bitip bitmediği kontrol edilebilir.

7.8 Nöron Modelinin Çalıştırılması

Nöron modelini çalıştıracak olan NMC üniteleri NMC_COLD_START [15:0] registeri ile çalıştırılmaktadır. NMC_COLD_START [15:0] registerinin her bir biti bir adet NMC ünitesini kontrol etmektedir. NMC üniteleri çalışmaya başladıktan sonra NMC_FINISHED [15:0] registeri kontrol edilerek NMC ünitesinin nöron modelini hesaplamayı bitirip bitirmediği kontrol edilir. Nöron modelinde bir program hatası oldup olmadığı NMC_MEM_VIOLATION [15:0] registerine yazılırken nöron modelinin çalıştırılmasında bir aritmetik hata yapılması durumunda NMC_MATH_ERROR [15:0] registerindeki ilgili NMC ünitesine karşılık gelen bit ‘1’ yapılarak nöron modelinin doğru bir biçimde işlenmediğini dış dünyaya bildirir. NMC_COLD_START [15:0] registerinin bitlerinin hangi NMC

ünitelerini çalıştırdığı ve NMC_FINISHED [15:0] bitlerinin hangi NMC ünitelerinden alındığı Tablo 7.8'de gösterilmiştir.

Tablo 7.8 Nmc_cold_start ve nmc_finished registerlerinin bitlerinin bağlı olduğu nmc üniteleri

Register	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_COLD_START [15:8]	NMC 15	NMC 14	NMC 13	NMC 12	NMC 11	NMC 10	NMC 09	NMC 08
Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NMC_COLD_START [7:0]	NMC 07	NMC 06	NMC 05	NMC 04	NMC 03	NMC 02	NMC 01	NMC 00
Register	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_FINISHED [15:8]	NMC 15	NMC 14	NMC 13	NMC 12	NMC 11	NMC 10	NMC 09	NMC 08
Register	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
NMC_FINISHED [7:0]	NMC 07	NMC 06	NMC 05	NMC 04	NMC 03	NMC 02	NMC 01	NMC 00

7.9 Öğrenme İşleminin Başlatılması

Öğrenme işlemi HYPERCOLUMN_LEARNING_START [3:0] registeri ile kontrol edilmektedir. Söz konusu registerin her biti bir HYPERCOLUMN ünitesini kontrol etmektedir. HYPERCOLUMN ünitelerinde öğrenme işlemi birbirlerinden bağımsız olarak başlatılabilir.

HYPERCOLUMN_LEARNING_DONE [3:0] registeri ilgili ünitede öğrenme işleminin bittiğini bildirir.

7.10 Güncellenen Nöron Durum Parametrelerinin Donanımdan Çekilmesi

Güncellenen nöron durum parametreleri NMC ünitelerinin Post-Sinaptik arayüzlerinden çekilmektedir. Güncellenen nöron durumlarının veya elde edilmek istenen diğer parametrelerin NMC ünitelerinden çekilmesi için NMC ünitelerinin Post-Sinaptik arayüzleri aktif edilmelidir.

NMC_POST_SYN_INTERFACE_CNTRL [15:0] registeri NMC ünitelerinin Post-Sinaptik arayüzlerini kontrol etmektedir. Register bitlerinin kontrol ettiği NMC üniteleri Tablo 7.9'da verilmektedir.

Tablo 7.9 Nmc_post_syn_interface_cntrl bitlerinin post-sinaptik arayüzünü aktif ettiği nmc üniteleri

Register Bitleri	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**
NMC_POST_SYN_INTERFACE_CNTRL [15:8]	NMC 15	NMC 14	NMC 13	NMC 12	NMC 11	NMC 10	NMC09	NMC 08
Register Bitleri	7	6	5	4	3	2	1	0
NMC_POST_SYN_INTERFACE_CNTRL [7:0]	NMC 07	NMC 06	NMC 05	NMC 04	NMC 03	NMC 02	NMC01	NMC 00

NMC Post-Sinaptik Arayüzlerinin aktif edilmesinden sonraki adım POST_SYN_MUX [3:0] ile genel NMC post sinaptik registerleri olan NMC_R_XX ile gösterilen registerlerin veri çekilecek NMC ünitelerinin post sinaptik arayüzlerine bağlanmasıdır. POST_SYN_MUX [3:0] ile NMC_XX registerlerinin hangi NMC Post-Sinaptik arayüzüne bağlandığı her bir register için Tablo 7.10'da verilmiştir.

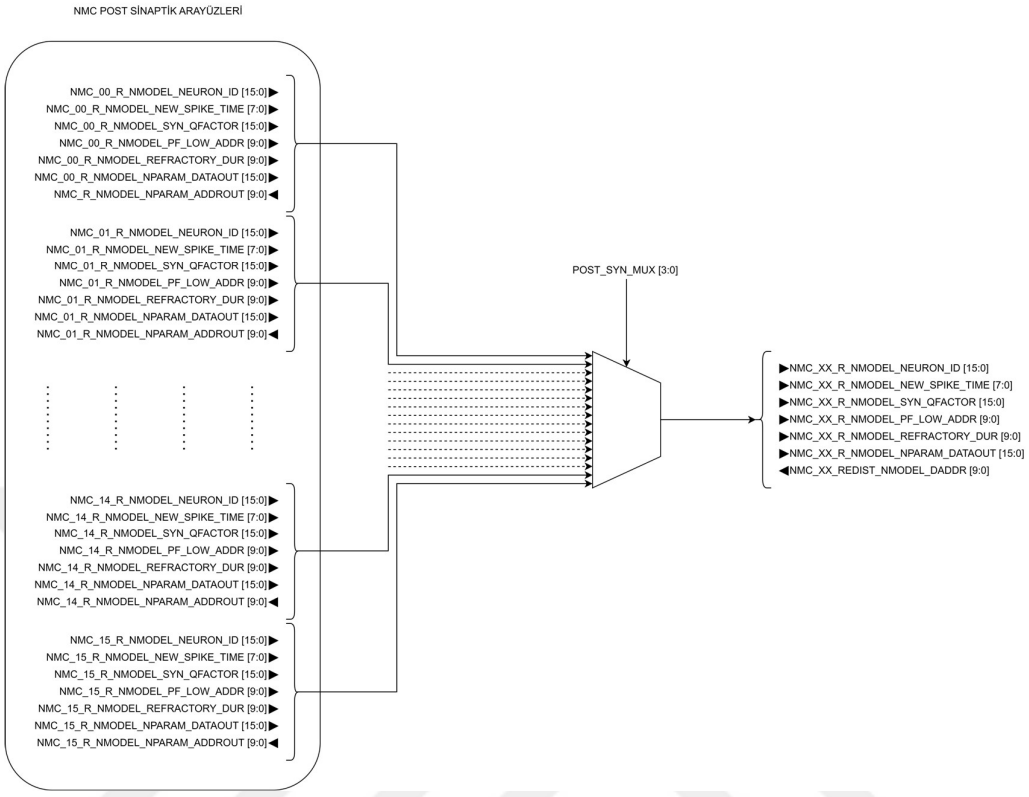
Tablo 7.10 Post_syn_mux registerinin aldığı değere göre nmc_xx registerinin nmc ünitelerinin post-sinaptik arayüzlerine bağlanması

POST_SYN_MUX [3:0]	**X"0"**	**X"1"**	**X"2"**	**X"3"**	**X"4"**	**X"5"**	**X"6"**	**X"7"**
NMC_XX_R_NEURON_ID [23:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_R_NEW_SPIKE_TIME [7:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_R_SYN_QFACTOR [15:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_R_PF_LOW_ADDR [9:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07

Tablo 7.10 Post_syn_mux registerinin aldığı değere göre nmc_xx registerinin nmc ünitelerinin post-sinaptik arayüzlerine bağlanması (devamı)

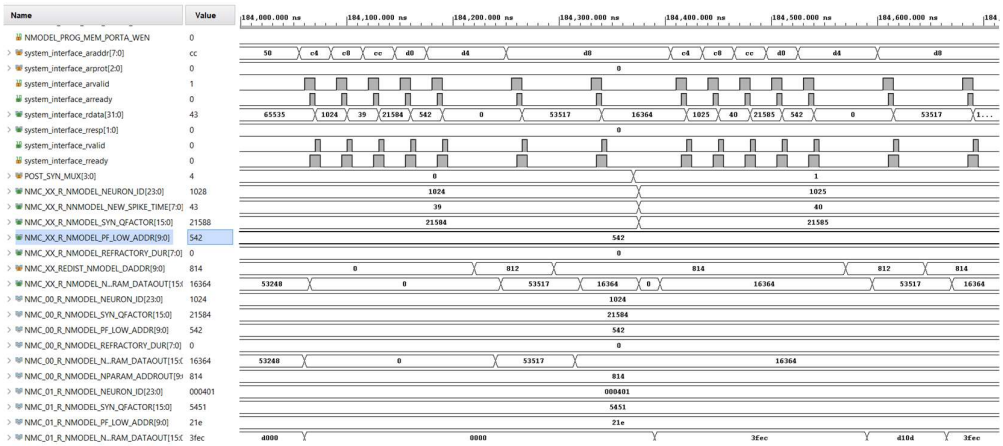
NMC_XX_R_NMODEL_REFRACTORY_DUR [7:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_R_NMODEL_NPARAM_DATAOUT [15:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
NMC_XX_REDIST_NMODEL_DADDR [9:0]	NMC 00	NMC 01	NMC 02	NMC 03	NMC 04	NMC 05	NMC 06	NMC 07
POST_SYN_MUX [3:0]	**X"8"**	**X"9"**	**X"A"**	**X"B"**	**X"C"**	**X"D"**	**X"E"**	**X"F**
NMC_XX_R_NEURON_ID [23:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_R_NEW_SPIKE_TIME [7:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_R_SYN_QFACTOR [15:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_R_PF_LOW_ADDR [9:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_R_NMODEL_REFRACTORY_DUR [7:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_R_NMODEL_NPARAM_DATAOUT [15:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15
NMC_XX_REDIST_NMODEL_DADDR [9:0]	NMC 08	NMC 09	NMC 10	NMC 11	NMC 12	NMC 13	NMC 14	NMC 15

Tablo 7.12'de gösterilen POST_SYN_MUX ile NMC post-sinaptik arayüzleri arasındaki ilişki Şekil 7.12'de görselleştirilmiştir.



Şekil 7.12 Nmc post-sinaptik arayüzlerinin post_syn_mux [3:0] registeri ile genel nmc post-sinaptik arayüzlerine bağlanması

NMC ünitelerinden nöron durumlarının geri okunması NMC_XX_R_NMODEL_NPARAM_DATAOUT [15:0] ve NMC_XX_REDIST_NMODEL_DADDR [9:0] registerleri aracılığı ile yapılmaktadır. Veri çekilmek istenen NMC ünitesinin önce post-sinaptik arayüzü aktif edilir. Arayüz aktif edildikten sonra POST_SYN_MUX veri çekilmek istenen NMC ünitesine bağlanarak NMC_XX_REDIST_NMODEL_DADDR [9:0] registerine çekilecek verinin adresi verilir ve veri NMC_XX_R_NMODEL_NPARAM_DATAOUT [15:0] registerine yazılır. Bu registerden AXI4-Lite arayüzü yardımıyla veri dış dünyaya verilir. Bu süreci gösteren bir test görüntüsü Şekil 7.13'te verilmiştir.



Şekil 7.13 Nmc post-sinaptik arayüzlerinden güncellenmiş durum parametrelerinin elde edilmesi

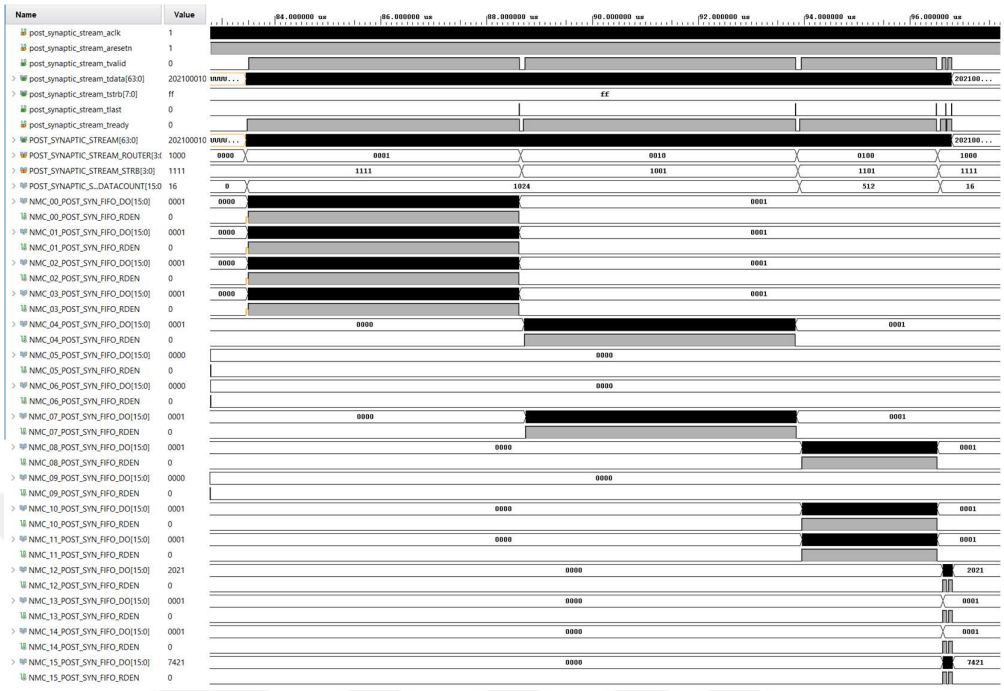
Şekil 7.12'de NMC00 ve NMC01 ünitelerinin post-sinaptik arayüzleri sırayla NMC_XX registerlerine POST_SYN_MUX [3:0] registeri aracılığı ile bağlanmış ve post-sinaptik arayüzündeki veriler AXI4-Lite arayüzünden ilgili registerler yardımıyla dış dünyaya verilmiştir.

7.11 Güncellenen Sinaptik Verilerin Donanımdan Çekilmesi

Sinaptik verilerin donanımdan çekilmesi Post-Sinaptik AXI Stream arayüzü yardımıyla yapılmaktadır. Post-Sinaptik AXI Stream arayüzü eğitim işleminden sonra Post-Sinaptik FIFO'lara yazılan sinaptik ağırlıkları alarak dış dünyaya göndermekte kullanılır.

Güncellenen verilerin donanımdan çekilmesinde 3 register rol oynamaktadır. POST_SYN_MUX [3:0], POST_SYNAPTIC_STREAM_DATACOUNT [15:0] ve POST_SYNAPTIC_STROBE [3:0] rol oynamaktadır.

Veri çekilme işlemi FIFO boşalana kadar değil istenen sayıda verinin çekilmesi şeklinde gerçekleştirilir. Post-Sinaptik FIFO'dan veri çekilmesinde çekilecek veri sayısını belirlemek için POST_SYNAPTIC_STREAM_DATACOUNT registeri kullanılır. Post-Sinaptik FIFO'lardan çekilen 16-Bit uzunluğundaki sinaptik veriler 64-bitlik tek bir veri halinde paketlenerek Post-Sinaptik AXI Stream arayüzünden dışarı verilir. Bu süreci gösteren bir test görüntüsü Şekil 7.14'te görülmektedir.

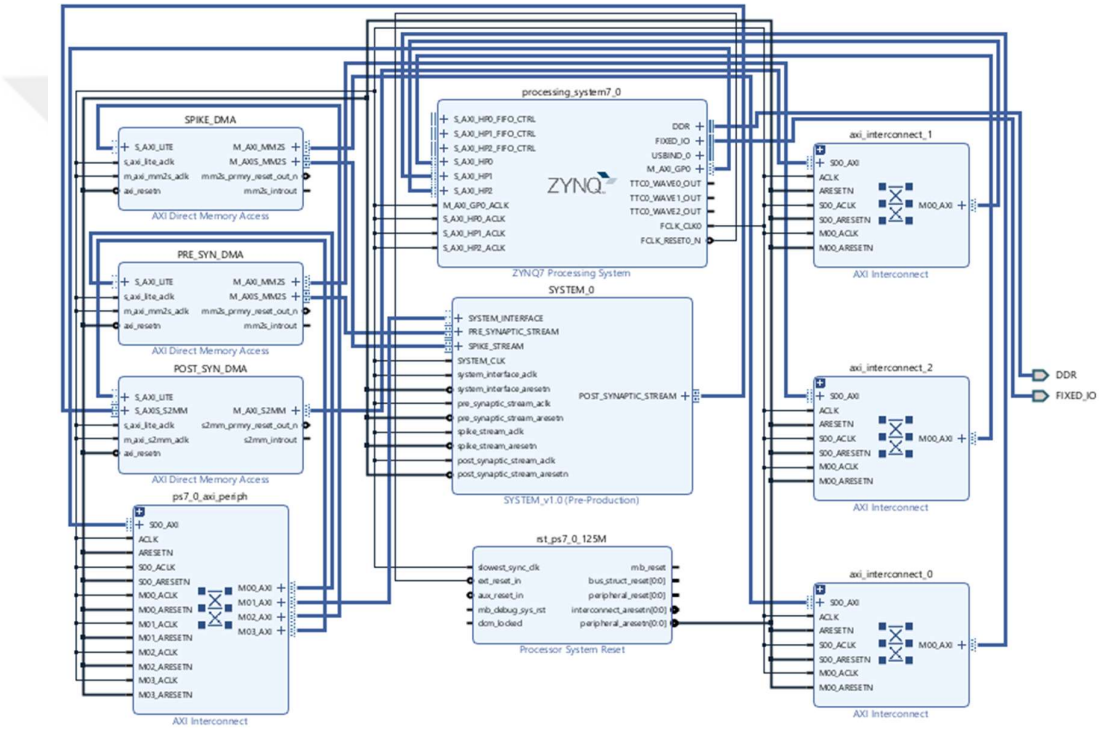


Şekil 7.14 Post-sinaptik fifolardan çekilen verilerin post-sinaptik axi stream arayüzünden dış dünyaya verilmesi

8
DENEYSEL SONUÇLAR

8.1 Kaynak Kullanımı

Donanımın test edilmesi için oluşturulan Vivado projesine ait bir blok tasarım Şekil 8.1'de verilmiştir.



Şekil 8.1 Sistemin testi için oluşturulan blok tasarım

Donanım arayüzleri tek yönlü çalışan AXI DMA blokları ile işletim sistemine bağlanmıştır. SPIKE_DMA atımların donanıma gönderilmesinde, PRE_SYN_DMA nöron ağırlıklarının donanıma gönderilmesinde, POST_SYN_DMA eğitilmiş nöron ağırlıklarının donanımdan işletim sistemine geri gönderilmesinde görev alırken, nöron durumları, ateşlenme bilgileri ve diğer programlama işlemleri önceki bölümde bahsedildiği gibi SYSTEM_INTERFACE arayüzü kullanılarak yapılmaktadır.

İşletim sistemi olarak PYNQ işletim sistemi seçilmiştir. PYNQ işletim sistemi ile Zynq-7000 hedef platformu üzerinde Jupyter Notebook ile Python kütüphaneleri kullanılabilmekte ve donanımla haberleşme yapılabilmektedir. Donanımın DMA blokları hariç toplam kaynak kullanımına ilişkin rapor Tablo 8.1'de sunulmuştur.

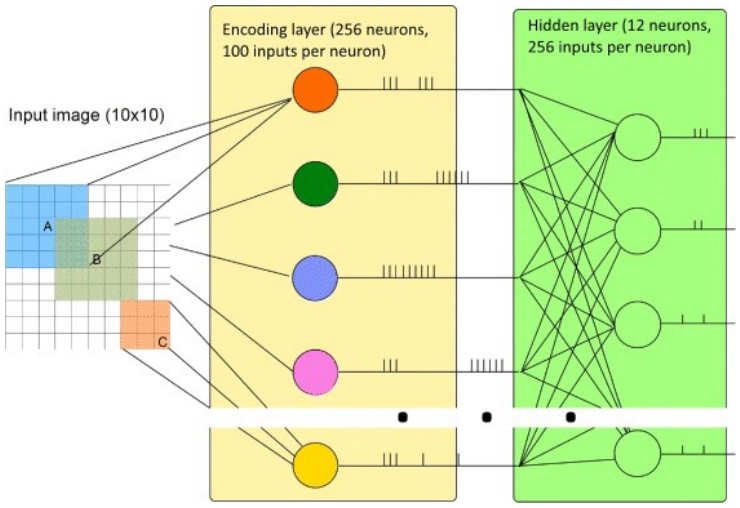
Tablo 8.1 Kaynak kullanım raporu

Kaynak	Toplam/Kullanılan
Slice LUT	53200/32197
Slice Registers	106400/44866
Block RAM	140/108
DSP	220/112

Tablo 8.1'de kaynak kullanım raporu verilen donanım 16 NMC, 32 PROCESSING ELEMENT, 32 ULEARN ünitesi içermektedir. Kullanılan hedef platformun kaynak miktarına göre implementasyon sonuçları ve ünite sayıları istenildiği takdirde değiştirilebilmektedir.

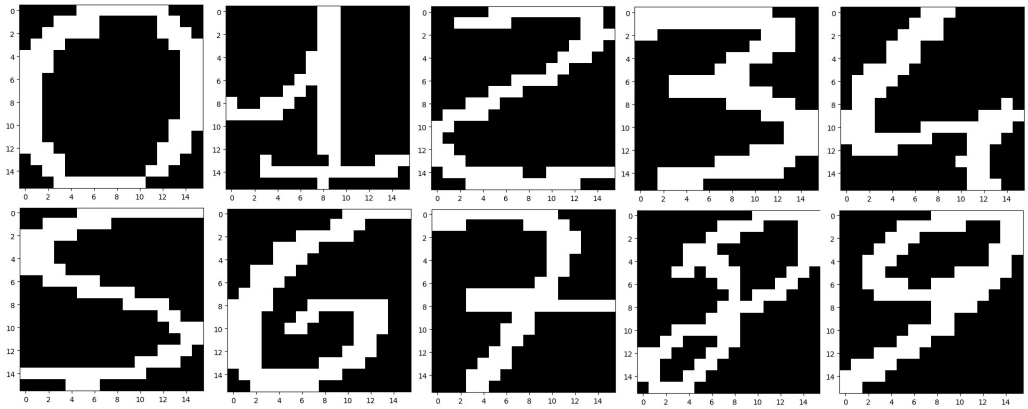
8.2 Örnek Ağ Mimarisi ve Performans Metrikleri

Örnek teşkil etmesi için donanım üzerinde koşturulan ağ oluşturulurken [18], [19], [20] ve [21] çalışmalardan yararlanılmıştır. Ağ mimarisi "Simplified spiking neural network architecture and STDP learning algorithm applied to image classification" [21] adlı çalışmadan doğrudan alınmıştır. Ağ mimarisi Şekil 8.2'de gösterilmiştir.



Şekil 8.2 Donanımda çalıştırılan ağın mimarisi [21]

Çalışmada[21] tanıtılan SNN ağı, 256 nöronluk bir kodlayıcı katmana, 12 nöronluk sınıflandırıcı katmana sahiptir. Kodlayıcı katman SEMEION veri setinden alınan 0-9 sayılara ait elyazısı ile yazılmış 16x16'lık bir siyah-beyaz resmin öznitelik haritasını çıkarmakta kullanılmıştır. Şekil 8.2'de 10x10'luk giriş resmi temsili olarak verilmiştir. SEMEION veri seti 1593 adet 16x16 boyutunda resim içermektedir. SEMEION veri setinden alınan bazı resimler Şekil 8.3'te gösterilmiştir.



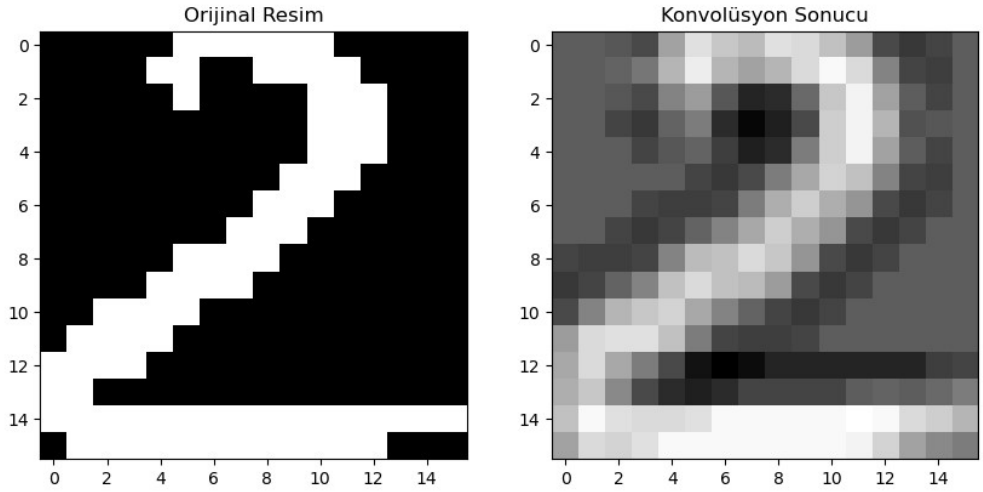
Şekil 8.3 Semeion veri setinin 0-9 arası sayılara ait bir alt sınıfı

SEMEION veri setinin resimleri 0 ve 1 değerli piksellerden oluşmaktadır. Ara değerlerin olmamasından ötürü resimlerde grinin tonları bulunmamaktadır. Resmin atımlara dönüştürülmesi için öncelikle konvolüsyon işlemine tabi tutulması gerekmektedir. Kodlayıcı katman konvolüsyon işlemini gerçekleştirmektedir.

Konvolüsyon işlemi merkez odaklı, [-0.5,1] arası uniform ağırlıklandırılmış, pikseller arasındaki Manhattan mesafesini ölçen bir filtre ile yapılmaktadır. Bu filtre, memeli görsel korteksinin tek noktaya odaklama işlemini modellemektedir. Filtrenin boyutları çalışmada 5x5 olarak belirlenmiştir. Filtrenin matris hali (8.1)'de verilmiştir.

$$RF = \begin{bmatrix} -0.5 & -0.125 & 0.125 & -0.125 & -0.5 \\ -0.125 & 0.125 & 0.625 & 0.125 & -0.125 \\ 0.125 & 0.625 & 1 & 0.625 & 0.125 \\ -0.125 & 0.125 & 0.625 & 0.125 & -0.125 \\ -0.5 & -0.125 & 0.125 & -0.125 & -0.5 \end{bmatrix} \quad (8.3)$$

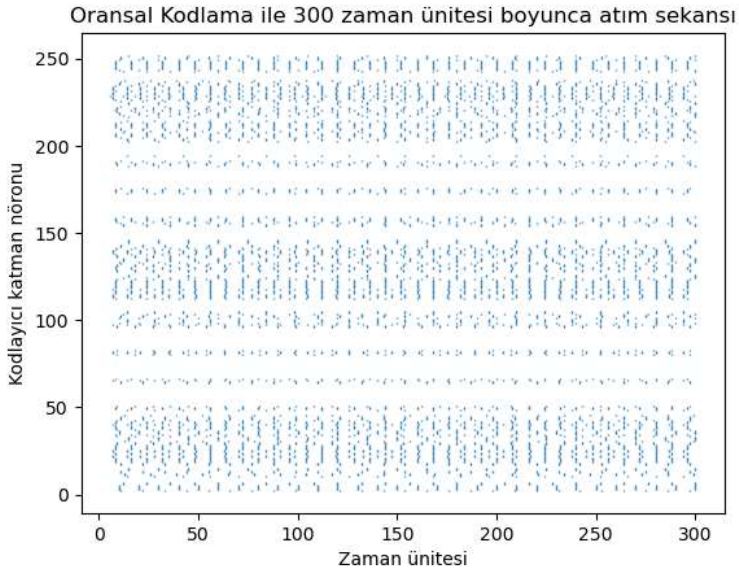
Filtrenin SEMEION veri setinden alınan bir resim ile konvolüsyonunun sonucu Şekil 8.4'te gösterilmiştir.



Şekil 8.4 "2" sayısına ait 16x16'lık görüntünün konvolüsyon sonucu

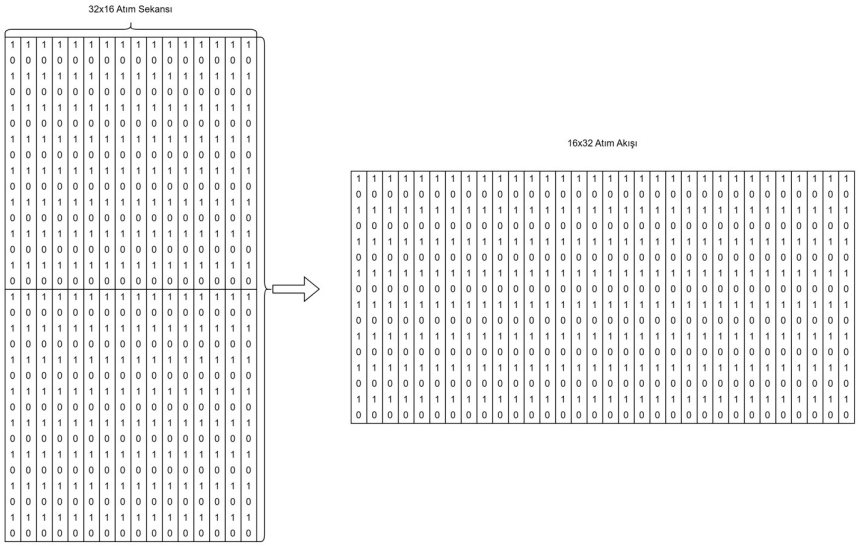
Öznitelik haritası olarak değerlendirilen konvolüsyon sonuçları, resmin içerdiği her piksel için üst üste bindirilerek kodlayıcı katman için membran gerilimi haritasını oluşturmaktadır. Söz konusu harita, ağın birinci katmanında atım sekanslarına dönüştürülmektedir. Dönüştürülen atım sekansları sınıflama görevini üstlenen ikinci katmana verilmektedir. Öğrenme kuralları, birinci ve ikinci katman arasındaki ağırlıklara uygulanmaktadır. Konvolüsyon ve atım sekansı oluşturma, işletim sistemi kısmında Python kodları ile yapılmaktadır.

Çalışmada [21] konvolüsyon sonucunun atım sekansına dönüştürülmesinde kullanılan teknik net bir biçimde ortaya konulmadığı için oransal kodlama tekniği kullanılmıştır. Oransal kodlamanın Şekil 8.4'teki resme uygulanmış halinin 300 zaman ünitesi boyuncaki atım sekansı Şekil 8.5'te görselleştirilmiştir. Atım kodlama işletim sistemi tarafında yapılmıştır.



Şekil 8.5 "2" sayısına ait 300 zaman ünitesi boyuncaki atım sekansı

Atımların SPIKE_BUFFER ünitelerine gönderilip HYPERCOLUMN ünitelerinde işlenebilmesi için 16-Bit uzunluğunda vektörler halinde paketlenmesi gerekmektedir. Paketleme işlemi yine işletim sistemi tarafında gerçekleştirilmektedir. Paketleme işleminin görselleştirilmiş hali Şekil 8.6'da verilmiştir.



Şekil 8.6 32x16'lık bir atım sekansının 16-bit işaretsiz tamsayı şeklinde paketlenmesi

Atım sekansları nöron indeksleri dikkate alınarak paketlendikten sonra donanımda girişim ve eğitim işlemleri yapılmaktadır.

Çalışmada [21] kullanılan nöron modeli, LiF nöron modelinin basitleştirilmiş halidir. Kullanılan nöron modeli (8.4)'te verilmiştir.

$$P_t = \begin{cases} P_{t-1} + \sum_{i=1}^{n} W_i S_{it} - D \ , & P_{min} < P_{t-1} < P_{threshold} \\ P_{refract} \ , & P_{t-1} \geq P_{threshold} \\ R_p \ , & P_{t-1} \leq P_{min} \end{cases} \quad (8.4)$$

Nöronların öğrenme kuralı olarak çalışmada STDP kullanılmıştır. Çalışmada [21] verilen STDP öğrenme penceresi 60 noktada örneklenmiştir. Donanım 128 negatif, 128 pozitif olmak üzere 256 örnekle çalışabilecek kapasitededir. Bu nedenle 60 yerine 256 örnek kullanılmıştır. Çalışmada [21] kullanılan STDP öğrenme penceresinin matematiksel ifadesi (8.5)'te verilmiştir. Öğrenme kuralları ağırlık değerleri olan W_i katsayılarını düzenlemektedir.

$$STDP(\Delta W) = \begin{cases} A^{-} exp\left(\frac{\Delta t}{\tau^{-}}\right) \ , & \Delta t \leq -2 \\ 0 \ , & -2 < \Delta t < 2 \\ A^{+} exp\left(\frac{\Delta t}{t^{+}}\right) \ , & \Delta t \geq -2 \end{cases} \quad (8.5)$$

Nöronların ağırlıklarının alabileceği maksimum değer 1, minimum değer -1 olarak seçilmiştir. Öğrenme oranı bütün nöronlar için 0.0625 olarak belirlenmiştir. Nöronların başlangıç membran gerilimi 0 olarak atanmıştır. Diğer nöron parametrelerinin değerleri Tablo 8.1'de verildiği gibidir. Bütün parametreler birimsizdir ve sadece test amacıyla rastgele atanmıştır.

Tablo 8.2 Nöron parametreleri

Nöron Parametresi	**Değeri**
R_p	0
D	3
$P_{threshold}$	77
$P_{refract}$	2
P_{min}	-37

Donanıma gönderilen membran haritaları kodlanırken 100 zaman adımı kullanılmıştır. Donanım atımları vektörler halinde almaktadır. Tek bir girişim döngüsünde 16 adet atım alınıp HYPERCOLUMN ünitelerine verilmektedir.

Donanım konvolüsyon işlemini desteklememektedir. Donanımda sadece girişim ve eğitim işlemleri yapılmaktadır. Dolayısıyla sinir ağının ilk katmanı olan kodlayıcı katman, işletim sistemi tarafından çalıştırılmaktadır. Gösterilen performans metrikleri girişim ve eğitim süreçlerini ilgilendirmektedir. Bir 16x16 resim için girişim, eğitim ve veri transferi işlemleri için geçen süreçler Tablo 8.3'te gösterilmiştir. Donanımın kendisi ve tüm arayüzleri 100 MHz saat frekansında çalışmaktadır. Tablo 8.3'te verilen zaman değerleri donanım içerisine sayaçlar gömülerek elde edilmiştir.

Tablo 8.3 Performans metrikleri

İşlem	**Geçen Zaman**
Ağırlıkların donanıma transferi	2.5 – 4 µs
256 sinapslı 16 nöronun tek vektör için girişim işlemi	4.01 µs
Nöron durumunun hesaplanması	1-1.1 µs
256 sinapslı 16 nöronun tek vektör için eğitim işlemi	10.26 µs
Eğitilmiş ağırlıkların donanımdan çekilmesi	2.5 – 4 µs
Toplam	15.37 µs

DMA üniteleri aynı anda 4 Pre-Sinaptik FIFO'ya veri göndermektedir. DMA ünitelerinin 256 adet veriyi transfer etmesi 2.5 – 4 µs arası ölçülmüştür. DDR RAM'in meşguliyetine göre DMA transfer hızları değişiklik göstermektedir.

Oluşturulan sinir ağının ağırlıkları [-1,1] aralığında rastgele dağıtılmıştır. Ağırlıkların kuantalama faktörü olarak 0.00784313725 değeri seçilmiştir.

DMA transfer hızının en kötü durumu göz önüne alındığında, örnek gösterilen ağda donanımın 1 adet 16x16'lık resim için girişim, nöron durumları hesaplama ve eğitim işlemini 33 ms sürede tamamladığı gözlemlenmiştir. Bütün ölçümler 100 MHz saat frekansı ile yapılmıştır.

9
SONUÇ

Çalışmada önerilen donanım mimarisi ile programlanabilir nöron modeli ve yazılımla programlanabilir sinaptik plastisiteye sahip bir programlanabilir lojik tabanlı nöromorfik sistem ortaya konulmuştur. Nöromorfik sistemlerde büyük önem arz eden yüksek hızda girişim, eğitim ve nöron modelinin hesaplanması işlemleri için ayrı ayrı çalışabilecek donanım blokları tasarlanarak bir mimari düzen içerisinde çalışmaları gözlemlenmiştir.

Programlanabilir bir nöron modelinin varlığı donanımın çok değişik görevlerde ve değişik deneysel çalışmalarda kullanılabilmesinin önünü açmaktadır. Ancak programlanabilir nöron modelleri, sabit nöron modellerine kıyasla daha fazla kaynak tüketmekte ve daha düşük frekanslarda çalışmayı da beraberinde getirmektedir.

Deneysel sonuçlarda gözlemlendiği üzere girişim, eğitim ve durum hesaplama hızları tek seferde sadece 16 adet nöron çalıştırabilen bir donanım için yeterli seviyededir. Deneysel sonuçlar göstermektedir ki SoC tabanlı sistemleri hedefleyen programlanabilir lojik tabanlı nöromorfik sistemleri kısıtlayan en önemli etmen işletim sistemi/CPU ile programlanabilir lojiğin haberleşmesinde geçen süreçtir. Deneysel sonuçlarda görüldüğü üzere eğitim, girişim ve parametre hesaplama işlemleri için toplam geçen süreç tek yönlü veri transferinde geçen süreçten çok daha azdır. İleriye yönelik çalışmalarda tasarımcıların veri alış verişini en aza indirgeyen, veya veri alış verişini mümkün olan en yüksek hızda gerçekleştirebilen donanımlar tasarlaması daha uygun bir tasarım yaklaşımı teşkil etmektedir.

KAYNAKÇA

[1] S. Li, Z. Zhang, R. Mao, J. Xiao, L. Chang and J. Zhou, "A Fast and Energy-Efficient SNN Processor With Adaptive Clock/Event-Driven Computation Scheme and Online Learning," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 68, no. 4, pp. 1543-1552, April 2021,

[2] N. N. Thao N., B. Veeravalli and X. Fong, "An FPGA-Based Co-Processor for Spiking Neural Networks with On-Chip STDP-Based Learning," 2022 IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 2022, pp. 2157-2161

[3] M. Heidarpur, A. Ahmadi, M. Ahmadi and M. R. Azghadi, "CORDIC-SNN: On-FPGA STDP Learning with Izhikevich Neurons," 2020 IEEE International Symposium on Circuits and Systems (ISCAS), Seville, Spain, 2020, pp. 1-1

[4] E. Z. Farsa, A. Ahmadi, M. A. Maleki, M. Gholami and H. N. Rad, "A Low-Cost High-Speed Neuromorphic Hardware Based on Spiking Neural Network," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 66, no. 9, pp. 1582-1586

[5] Y. Liu, Y. Chen, W. Ye and Y. Gui, "FPGA-NHAP: A General FPGA-Based Neuromorphic Hardware Acceleration Platform With High Speed and Low Power," in IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 69, no. 6, pp. 2553-2566, June 2022, doi: 10.1109/TCSI.2022.3160693.

[6] C. Lammie, T. Hamilton and M. R. Azghadi, "Unsupervised Character Recognition with a Simplified FPGA Neuromorphic System," 2018 IEEE International Symposium on Circuits and Systems (ISCAS), Florence, Italy, 2018, pp. 1-5, doi: 10.1109/ISCAS.2018.8351532.

[7] L. Lapicque, “Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization,” Journal de Physiologie et de Pathologie Generalej, vol. 9, pp. 620-635, 1907.

[8] A.L. Hodgkin, A.F. Huxley, “A quantitative description of membrane current and its application to conduction and excitation in nerve.” J Physiol.116, 449-566, 1952

[9] E. M. Izhikevich, "Simple model of spiking neurons", IEEE Transactions on Neural Networks, vol. 14, no. 6, pp. 1569-1572, Nov. 2003

[10] Gerstner, Wulfram, “Neuronal dynamics: from single neurons to networks and models of cognition.”, Kistler, Werner M., 1969-, Naud, Richard., Paninski, Liam. Cambridge, United Kingdom.

[11] Y. Zexiang, L. Jing, L. Qidong, Z. Hegui, L. Dong, L. Jizhao,"Learning rules in spiking neural networks: A survey", Neurocomputing, Volume 531, Pages 163-179, 2023

[12] M. Davies et al., "Loihi: A Neuromorphic Manycore Processor with On-Chip Learning," in IEEE Micro, vol. 38, no. 1, pp. 82-99, Feb. 2018

[13] S. Furber and P. Bogdan, "SpiNNaker: A Spiking Neural Network Architecture", Directory of Open Access Books, 2020,

[14] Zynq-7000 SoC Data Sheet: Overview (DS190) AMD Corporation

[15] G. Wenzhe, Eltawil. F. Mohammed, M. Ahmed, Nabil S. Khaled," Neural Coding in Spiking Neural Networks: A Comparative Study for Robust Neuromorphic Systems", Frontiers in Neuroscience, Volume 15, 2021

[16] D.E. Shulz, D.E. Feldman, in ''Neural Circuit Development and Function in the Brain'' Edited by: John L.R. Rubenstein and Pasko Rakic, 2013

[17] 7 Series DSP48E1 Slice User Guide (UG479) AMD Corporation

[18] T. Masquelier, S.J. Thorpe, "Unsupervised Learning of Visual Features through Spike Timing Dependent Plasticity", PLoS Comput. Biol. 3, 2007

[19] P.U. Diehl, M. Cook, "Unsupervised learning of digit recognition using spike-timing-dependent plasticity", Frontiers in Computational Neuroscience, vol. 9, 2015

[20] G. Srinivasan, P. Panda, and K. Roy," STDP-based Unsupervised Feature Learning using Convolution-over-time in Spiking Neural Networks for Energy-Efficient Neuromorphic Computing" ACM Journal of Emerging Technolgies on Computing Systems 14, October 2018

[21] T. Iakymchuk, A. Rosado-Muñoz, J.F. Guerrero-Martínez, et al. "Simplified spiking neural network architecture and STDP learning algorithm applied to image classification", EURASIP Journal on Image and Video Processing, 4, 2015

TEZDEN ÜRETİLMİŞ YAYINLAR

Konferans Bildirileri

1. O. Yüksel, B. Erkmen “Hardware- Software Codesign Approach In Customizable Programmable Logic Based Neuromorphic System Design”, 2023 30th IEEE International Conference on Electronics, Circuits and Systems (ICECS)