

İSTANBUL TEKNİK ÜNİVERSİTESİ
ELEKTRİK ELEKTRONİK FAKÜLTESİ



INTRODUCTION TO EMBEDDED SYSTEMS

(EHB 326E)

Parallel Programing (Dijkstra)

Hasan Emre AYDEMİR

Faruk ONAR

Ozan İNAL

1. GİRİŞ

En kısa yol (path finding) problemleri, grafik tabanlı sistemlerde iki düğüm arasındaki en düşük maliyetli yolun bulunmasını amaçlayan temel problemlerdendir. Bu tür problemler; haberleşme ağları, ulaşım sistemleri, robotik uygulamalar ve gömülü sistemler gibi birçok alanda yaygın olarak kullanılmaktadır. Özellikle düğüm sayısının artmasıyla birlikte hesaplama karmaşıklığı yükselmekte ve bu durum paralel hesaplama yaklaşımlarını gerekli kılmaktadır.

Bu projede, 10×10 boyutunda bir komşuluk matrisi kullanılarak, 0 numaralı düğümden diğer tüm düğümlere olan en kısa yolların bulunması hedeflenmiştir. Sistem, PicoBlaze tabanlı bir master-slave mimarisi üzerine kurulmuştur. Hesaplama yükünü azaltmak ve işlem süresini kısaltmak amacıyla iki adet slave PicoBlaze işlemci paralel olarak çalıştırılmıştır. Slave işlemciler en kısa yol hesaplamalarını eş zamanlı olarak gerçekleştirirken, ortak bir bellek yapısı (shared memory) üzerinden veri paylaşımı yapılmıştır. Master PicoBlaze işlemci ise sistemin genel kontrolünü sağlamış ve elde edilen sonuçları kendi register yapısında toplamıştır.

Gerçekleştirilen tasarımda, paralel hesaplamanın etkinliği hem simülasyon çıktıları hem de waveform analizleri ile doğrulanmıştır. Elde edilen sonuçlar, slave işlemciler tarafından paylaşılan belleğe yazılmış, ardından master işlemci tarafından okunarak master register'larına aktarılmıştır. Bu sayede, hem paralel işlem mantığı hem de merkezi sonuç toplama mekanizması başarıyla uygulanmıştır.

2. PROBLEM TANIMI

Bu çalışmada ele alınan problem, 10 düğümden oluşan yönlü ve ağırlıklı bir grafik üzerinde, 0 numaralı düğümden diğer tüm düğümlere olan en kısa yolların bulunmasıdır. Grafik yapısı, 10×10 boyutunda bir komşuluk (adjacency) matrisi ile temsil edilmiştir. Matriste her bir hücre, ilgili düğümler arasındaki geçiş maliyetini göstermekte olup, bağlantı bulunmayan durumlar yüksek maliyet (sonsuz) değeri ile ifade edilmiştir.

Sistemin temel gereksinimi, en kısa yol hesaplamalarının paralel olarak gerçekleştirilmesidir. Bu amaçla, bir adet master ve iki adet slave PicoBlaze işlemciden oluşan bir mimari tasarlanmıştır. Slave işlemciler, hedef düğümleri kendi aralarında paylaşarak paralel biçimde en kısa yol hesaplamalarını gerçekleştirmektedir. Slave 1, 0 düğümden 1–4 arası düğümlere olan yolları hesaplar; Slave 2, 5–9 arası düğümlere olan yolları hesaplamaktadır. Böylece hesaplama yükü paylaştırılarak işlem süresinin azaltılması hedeflenmiştir.

Hesaplanan ara ve nihai sonuçlar, slave işlemciler tarafından ortak bir bellek alanına (shared memory) yazılmaktadır. Master işlemci, bu bellekte tutulan sonuçları okuyarak sistemin genel kontrolünü sağlamakta ve en kısa yol sonuçlarını kendi register yapısında toplamaktadır. Bu yaklaşım, ödev tanımında belirtilen “hesaplamaların işbirliği içinde yapılması ve sonuçların master register’larında tutulması” gereksinimini doğrudan karşılamaktadır.

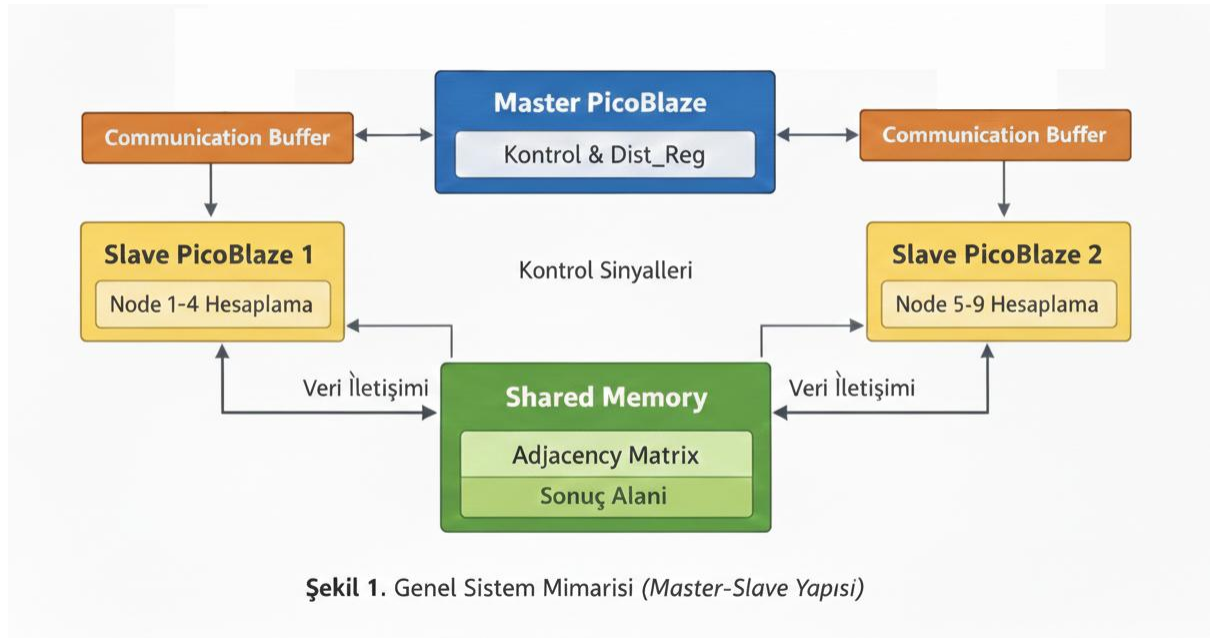
Sonuç olarak, bu problem kapsamında amaçlanan; paralel çalışan slave işlemciler aracılığıyla en kısa yol hesaplamalarının gerçekleştirilmesi, paylaşılan bellek üzerinden veri iletişiminin sağlanması ve nihai sonuçların master işlemci register'larında güvenli ve merkezi bir şekilde tutulmasıdır.

3. SİSTEM MİMARİSİ

Bu projede, en kısa yol hesaplamalarının paralel olarak gerçekleştirilebilmesi amacıyla master-slave tabanlı bir donanım mimarisi tasarlanmıştır. Sistem, bir adet master PicoBlaze işlemci, iki adet slave PicoBlaze işlemci, paylaşılan bir bellek yapısı (shared memory) ve işlemciler arası iletişimi sağlayan haberleşme tamponlarından (communication buffers) oluşmaktadır. Tasarlanan mimari, hesaplama yükünün paralel olarak dağıtılmasına ve sonuçların merkezi bir yapıda toplanmasına olanak sağlamaktadır.

3.1 Genel Sistem Yapısı

Şekil 1'de, tasarlanan sistemin genel blok diyagramı gösterilmektedir. Master PicoBlaze işlemci, sistemin kontrol birimi olarak görev yapmaktadır. Slave PicoBlaze işlemciler ise en kısa yol hesaplamalarını paralel olarak gerçekleştirmektedir. Slave işlemciler, hesapladıkları sonuçları paylaşılan bellek alanına yazmakta; master işlemci ise bu sonuçları okuyarak kendi sonuç register'larında toplamaktadır.



Şekil 1. Genel Sistem Mimarisi (Master-Slave Yapısı)

3.2 Genel Sistem Yapısı

Master PicoBlaze işlemci, sistemin ana kontrol birimi olarak görev yapmaktadır. Master işlemci, slave işlemcilerin başlatılmasını ve genel akışın yönetilmesini sağlar. Hesaplama işlemleri doğrudan master tarafından yapılmamakta; bunun yerine slave işlemciler tarafından hesaplanan sonuçlar paylaşılan bellek üzerinden okunmaktadır.

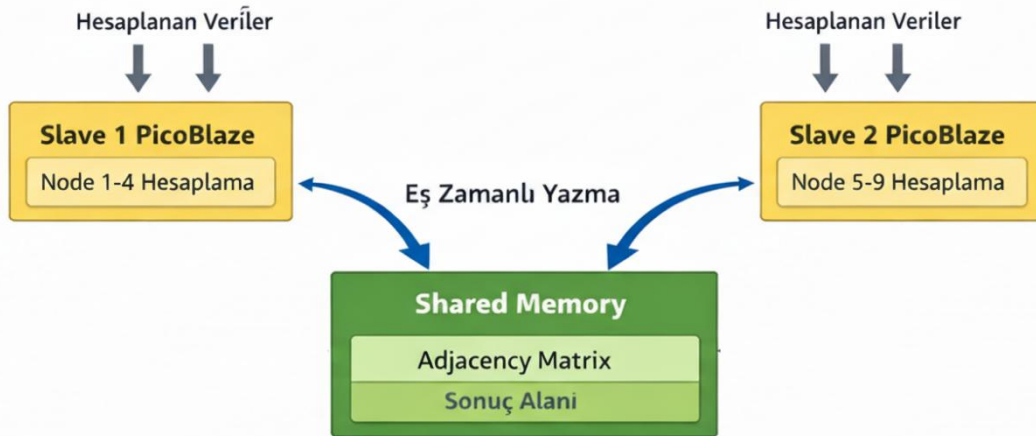
Elde edilen nihai en kısa yol sonuçları, master işlemciye ait donanımsal sonuç register'ları olan `dist_reg[0..9]` dizisine aktarılmaktadır. Bu register yapısı, 0 düğümünden diğer düğümlere olan en kısa mesafelerin merkezi ve güvenli bir şekilde tutulmasını sağlamaktadır. Böylece ödev tanımında belirtilen “sonuçların master register’larında tutulması” şartı doğrudan karşılanmıştır.

3.3 Genel Sistem Yapısı

Sistemde iki adet slave PicoBlaze işlemci bulunmaktadır. Slave işlemciler paralel çalışacak şekilde yapılandırılmıştır. Hesaplama yükü, düğüm aralıklarına göre slave’ler arasında paylaştırılmıştır:

- Slave 1: 0 düğümünden 1–4 arası düğümlere olan en kısa yolları hesaplar.
- Slave 2: 0 düğümünden 5–9 arası düğümlere olan en kısa yolları hesaplar.

Bu yaklaşım sayesinde, hesaplamalar aynı zaman diliminde eş zamanlı olarak gerçekleştirilmekte ve sistemin toplam çalışma süresi azaltılmaktadır. Slave işlemciler, hesapladıkları ara ve nihai sonuçları paylaşılan belleğe yazmaktadır.



Şekil 2. Paralel Çalışan Slave PicoBlaze İşlemcileri

3.4 Paylaşılan Bellek (Shared Memory)

Shared memory modülü, slave işlemciler arasında ortak veri paylaşımını sağlamak amacıyla tasarlanmıştır. Bellek yapısı 256 baytlık bir RAM alanından oluşmaktadır. Bu bellek içerisinde:

- 0–99 adres aralığı: Komşuluk (adjacency) matrisi
- 100–109 adres aralığı: En kısa yol sonuçları

olarak kullanılmıştır.

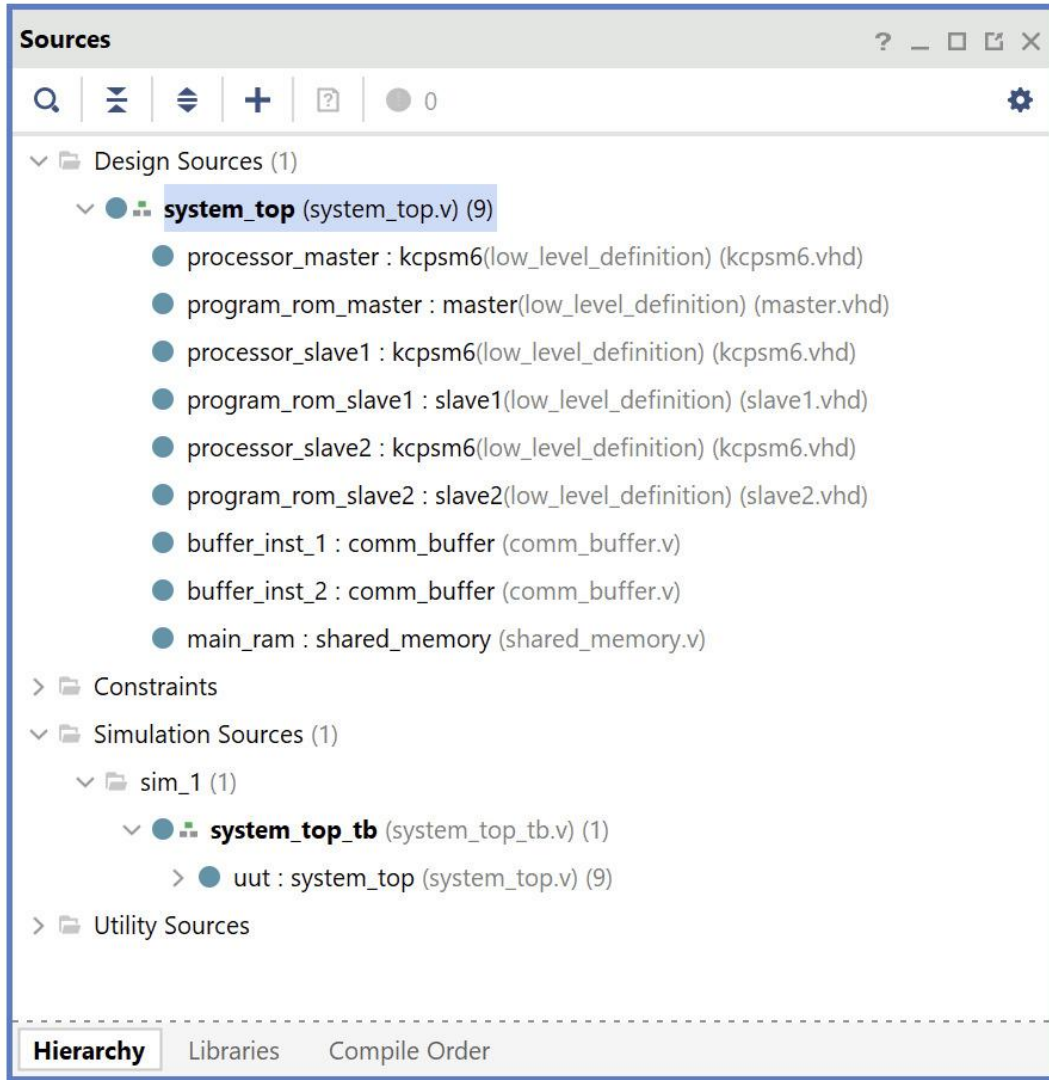
Slave işlemciler, hesapladıkları sonuçları bu bellek alanına yazarken; master işlemci sonuçları bu bellekten okuyarak kendi register yapısına aktarmaktadır. Bellek erişimleri senkron yazma ve asenkron okuma mantığına göre gerçekleştirilmiştir.

3.5 Haberleşme Tamponları (Communication Buffers)

Master ve slave işlemciler arasındaki kontrol sinyalleri ve komut iletimi, communication buffer modülleri aracılığıyla sağlanmaktadır. Bu tamponlar sayesinde master işlemci, slave işlemcileri başlatabilmekte ve işlem akışını kontrol edebilmektedir. Buffer yapısı, işlemciler arası veri çakışmasını önleyerek güvenilir bir haberleşme altyapısı sunmaktadır.

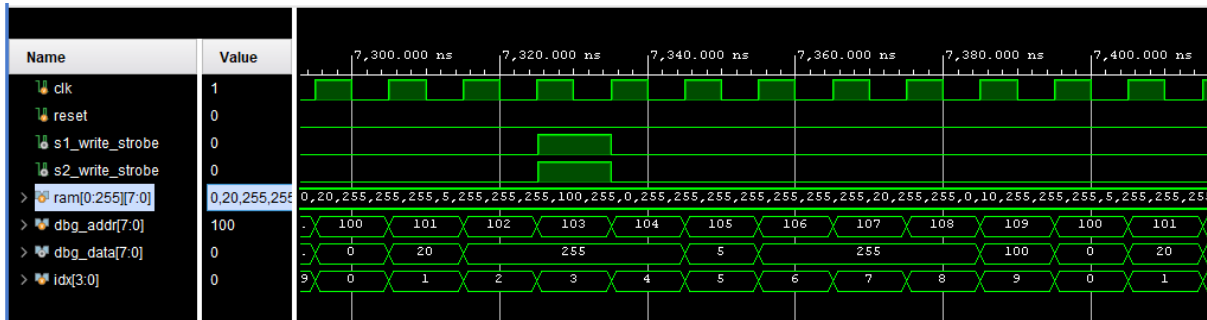
3.6 Sistem Mimarisi Değerlendirmesi

Tasarlanan sistem mimarisi, paralel hesaplama, paylaşılan bellek kullanımı ve merkezi sonuç toplama prensiplerini bir arada barındırmaktadır. Slave işlemcilerin paralel çalışması sayesinde hesaplama süresi azaltılmış, master işlemcinin sonuçları kendi register yapısında toplamasıyla sistem kontrolü ve raporlama işlemleri merkezi hale getirilmiştir. Bu mimari, hem ödev gereksinimlerini karşılamakta hem de gerçekçi bir paralel işlemci sistemi örneği sunmaktadır. Şekil 3'te Vivado ortamındaki kaynak hiyerarşisi gösterilmektedir. Bu görünümde master ve slave PicoBlaze işlemciler, haberleşme tamponları ve paylaşılan bellek modülü açıkça görülmekte olup, tasarlanan mimarinin donanımsal olarak doğru şekilde gerçekleştirildiğini doğrulamaktadır.



Şekil 3 Vivado Sources

4. PARALEL ÇALIŞMA MANTIĞI



Şekil 4. Paralel Çalışma Mantığını Gösteren Waveform Diyagramı

Tasarlanan sistemde paralel hesaplama mantığı, iki adet slave PicoBlaze işlemcisinin eş zamanlı olarak çalıştırılmasıyla gerçekleştirilmiştir. Slave işlemciler, en kısa yol hesaplamalarını farklı düğüm aralıkları için paralel biçimde yaparak hesaplama yükünü paylaşmaktadır. Bu yaklaşım, sistemin toplam çalışma süresini azaltmayı ve paralel işlemci kullanımını göstermeyi amaçlamaktadır.

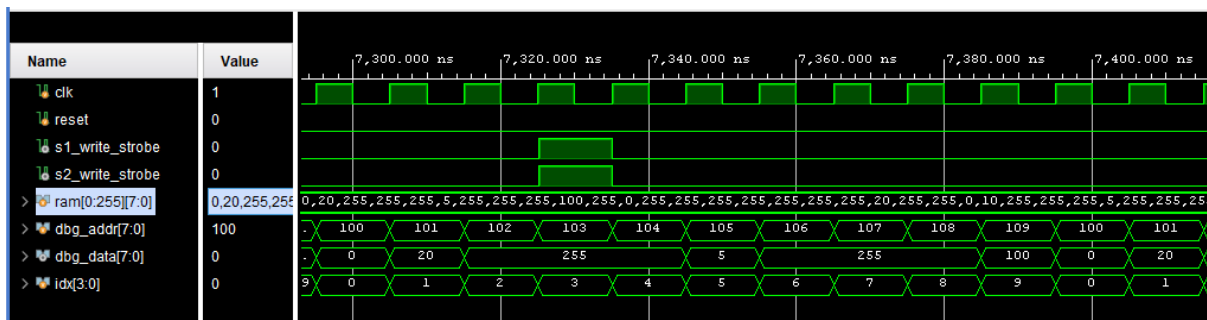
Şekil 4'te verilen waveform diyagramı incelendiğinde, Slave 1 ve Slave 2 işlemcilerinin paralel çalıştığı açıkça görülmektedir. s1_write_strobe ve s2_write_strobe sinyallerinin reset sinyalinin düşmesinden sonra aktif hale geldiği ve zaman ekseninde örtüşen aralıklarda aynı anda lojik “1” seviyesine çıktığı gözlemlenmektedir. Bu durum, iki slave işlemcinin aynı zaman dilimi içerisinde paylaşılan belleğe yazma işlemi gerçekleştirdiğini ve paralel olarak çalıştığını kanıtlamaktadır.

Waveform üzerinde ayrıca, her iki slave işlemcinin paylaşılan belleğe farklı adresler üzerinden eriştiği görülmektedir. Bu sayede veri çakışması oluşmadan, paralel bellek yazma işlemleri başarıyla gerçekleştirilmiştir. Paralel hesaplama sonucunda elde edilen en kısa yol değerleri, shared memory içerisinde ilgili adreslere yazılmıştır.

Master PicoBlaze işlemcinin rolü ise waveform üzerinde dbg_addr, dbg_data ve idx sinyalleri aracılığıyla gözlemlenmektedir. dbg_addr sinyalinin 100–109 adres aralığında sıralı olarak ilerlediği, dbg_data sinyalinin ise bu adreslerde tutulan bellek değerlerini doğru şekilde yansıttığı görülmektedir. idx sayacının 0’dan 9’a kadar ilerlemesi, master işlemcinin shared memory’deki sonuçları sırasıyla okuyarak kendi sonuç register’larına aktardığını göstermektedir.

Sonuç olarak, elde edilen waveform çıktıları; slave işlemcilerin paralel çalıştığını, shared memory yapısının eş zamanlı erişime uygun olduğunu ve master işlemcinin nihai sonuçları merkezi olarak topladığını açık bir şekilde doğrulamaktadır. Bu yapı, ödev kapsamında istenen paralel hesaplama yaklaşımının donanımsal olarak başarıyla uygulandığını göstermektedir.

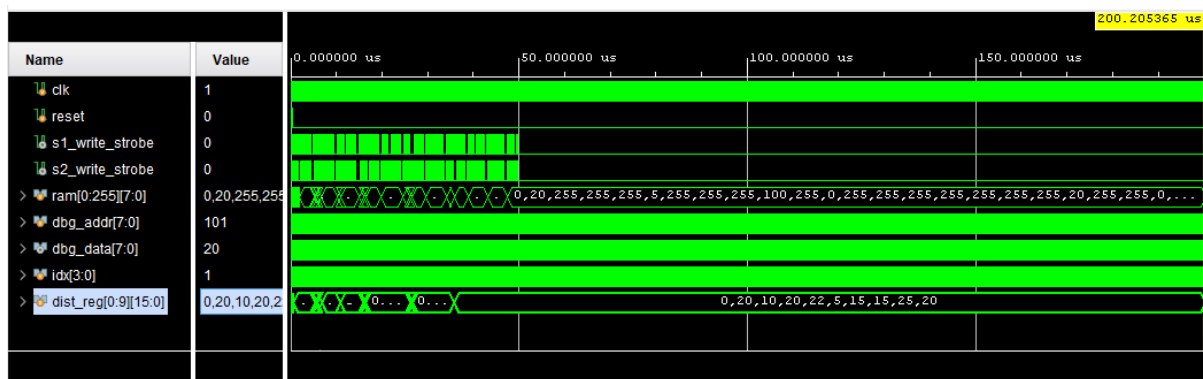
5. BELLEK YAPISI VE VERİ AKIŞI



Şekil 5. Paylaşılan Bellek Üzerinden Veri Akışını Gösteren Waveform

Bu projede, slave PicoBlaze işlemciler ile master PicoBlaze işlemci arasındaki veri paylaşımı, paylaşılan bir bellek (shared memory) üzerinden gerçekleştirilmiştir. Shared memory, 256 baytlık bir RAM alanı olarak tasarlanmış olup, 0–99 adres aralığı komşuluk (adjacency) matrisini, 100–109 adres aralığı ise 0 numaralı düğümden diğer düğümlere olan en kısa yol sonuçlarını tutmaktadır. Slave işlemciler, paralel olarak gerçekleştirdikleri en kısa yol hesaplamalarının sonuçlarını bu bellek alanına yazmakta; master işlemci ise bu sonuçları bellekten sıralı şekilde okuyarak kendi sonuç register'larına aktarmaktadır. Waveform çıktıları, slave işlemcilerin paylaşılan belleğe eş zamanlı yazma işlemleri gerçekleştirdiğini ve master işlemcinin bellek adreslerini 100–109 aralığında tarayarak doğru sonuçları elde ettiğini açıkça göstermektedir. Bu yapı sayesinde veri akışı düzenli, güvenli ve paralel hesaplamaya uygun bir şekilde gerçekleştirilmiştir.

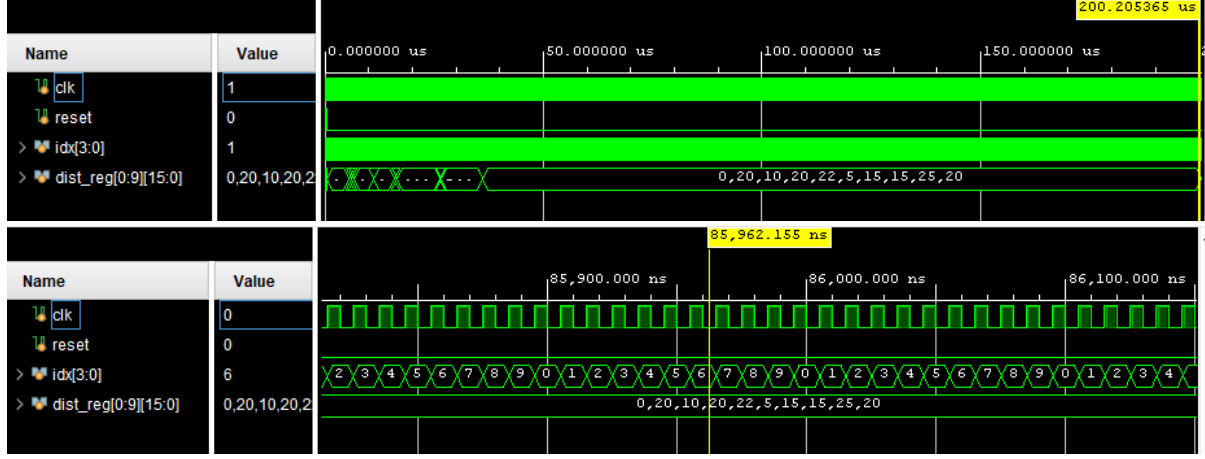
6. MASTER REGISTER TASARIMI VE SONUÇ TOPLAMA



Şekil 6. Master Register'lara Sonuçların Aktarılmasını Gösteren Waveform

Bu çalışmada, slave PicoBlaze işlemciler tarafından paralel olarak hesaplanan en kısa yol sonuçları, paylaşılan bellek üzerinden master PicoBlaze işlemciye aktarılmış ve master'a ait sonuç register'larında saklanmıştır. Master register yapısı, `dist_reg[0..9]` dizisi olarak tasarlanmış olup, 0 numaralı düğümden diğer tüm düğümlere olan en kısa mesafeleri tutmaktadır. Waveform çıktıları incelendiğinde, master işlemcinin `dbg_addr` sinyali aracılığıyla shared memory içerisindeki 100–109 adres aralığını sıralı olarak okuduğu ve `dbg_data` sinyali üzerinden alınan değerleri kendi register yapısına aktardığı açıkça görülmektedir. Nihai durumda `dist_reg` dizisinin (0, 20, 10, 20, 22, 5, 15, 15, 25, 20) değerlerini aldığı ve bu değerlerin bellek içerikleri ile birebir uyumlu olduğu doğrulanmıştır. Bu yapı sayesinde, hesaplama işlemleri slave işlemciler tarafından gerçekleştirilmiş, sonuçlar ise merkezi olarak master register'larında toplanarak ödev kapsamında istenen mimari başarıyla uygulanmıştır.

7. SİMÜLASYON SONUÇLARI VE DOĞRULAMA



Şekil 7. Simülasyon Sonunda Master Register’ların Durumunu Gösteren Waveform

Tasarlanan sistemin doğruluğunu ve işlevselliğini doğrulamak amacıyla Vivado simülasyon ortamında kapsamlı bir testbench kullanılarak simülasyonlar gerçekleştirilmiştir. Simülasyon sonucunda, 0 numaralı düğümden diğer tüm düğümlere olan en kısa yol maliyetleri başarıyla hesaplanmıştır.

Waveform çıktıları incelendiğinde, reset sinyalinin pasif hale gelmesinden sonra master işlemciye ait sonuç register’ları olan dist_reg[0..9] dizisinin nihai değerlere ulaştığı ve bu değerlerin simülasyonun geri kalan süresi boyunca değişmeden sabit kaldığı görülmektedir. Elde edilen sonuçlara göre, 0 numaralı düğümden 5 numaralı düğüme olan en kısa yol maliyeti 5 olarak hesaplanmıştır. Bununla birlikte, testbench kapsamında hedef düğüm olarak belirlenen Node 9 için en kısa yol maliyetinin 20 olduğu belirlenmiştir.

Şekil 7’de gösterilen waveform, master işlemcinin tüm düğümler için sonuçları başarıyla topladığını ve register yapısında sakladığını kanıtlamaktadır. Buna ek olarak, Şekil 8’de verilen TCL (Transcript) çıktısı incelendiğinde, otomatik doğrulama mekanizmasının doğru sonuç verdiği ve sistemin “TEST BAŞARILI” durumu ile simülasyonu tamamladığı görülmektedir.

Sonuç olarak, hem waveform hem de TCL çıktıları birlikte değerlendirildiğinde, paralel çalışan slave PicoBlaze işlemciler tarafından hesaplanan en kısa yol sonuçlarının shared memory üzerinden master işlemciye aktarıldığı, master register’larında doğru şekilde saklandığı ve sistemin bütünsel olarak hatasız çalıştığı doğrulanmıştır.

```
run all

-----
--- SONUÇLAR (Hedef Node 9) ---
-----

Node 0:  0 (Beklenen: 0)
Node 5:  5 (Beklenen: 5)
Node 2: 10 (Beklenen: 10)
Node 7: 15 (Beklenen: 15)
Node 9: 20 (Beklenen: 20)

-----
>>> TEST BAŞARILI! <<< En kısa yol (20) bulundu.
---- MASTER REGISTER RESULTS (dist_reg) ----
0->1 : 20
0->2 : 10
0->3 : 20
0->4 : 22
0->5 : 5
0->6 : 15
0->7 : 15
0->8 : 25
0->9 : 20
$finish called at time : 200300 ns : File "C:/Vivado
```

Şekil 8. En Kısa Yol Sonuçlarını Gösteren TCL Çıktısı

8. SONUÇ VE DEĞERLENDİRME

Bu projede, 10×10 boyutunda bir grafik yapısı üzerinde en kısa yol probleminin paralel programlama yaklaşımı kullanılarak çözülmesi amaçlanmıştır. Tasarlanan sistemde bir PicoBlaze işlemci master, iki PicoBlaze işlemci ise slave olarak görev yapmış; slave işlemciler en kısa yol hesaplamalarını paralel olarak gerçekleştirirken, master işlemci program akışını yönetmiş ve elde edilen sonuçları merkezi olarak toplamıştır.

Gerçekleştirilen simülasyonlar sonucunda, slave işlemciler tarafından hesaplanan en kısa yol maliyetlerinin paylaşılan bellek üzerinden master işlemciye başarıyla aktarıldığı ve master register'larında doğru şekilde saklandığı doğrulanmıştır. Waveform ve TCL çıktıları incelendiğinde, sistemin kararlı bir şekilde çalıştığı, hedef düğüm olan Node 9 için en kısa yol maliyetinin 20 olarak doğru biçimde hesaplandığı ve diğer tüm düğümler için elde edilen sonuçların beklenen değerlerle uyumlu olduğu görülmüştür.

Bu çalışma, paralel işlem mimarisinin donanımsal olarak PicoBlaze tabanlı bir sistem üzerinde başarıyla uygulanabileceğini göstermektedir. Hesaplama yükünün slave işlemciler arasında paylaştırılması sayesinde sistemin modülerliği ve ölçeklenebilirliği artırılmıştır. Sonuç olarak, geliştirilen mimari hem paralel hesaplama mantığını hem de master-slave tabanlı veri paylaşımını açık ve etkili bir şekilde ortaya koymaktadır.

DURUM 1

Kaynak \ Hedef	0	1	2	3	4	5	6	7	8	9
0	00	20	FF	FF	FF	05	FF	FF	FF	100
1	FF	00	FF	FF	FF	FF	FF	FF	FF	20
2	FF	FF	00	10	FF	FF	FF	05	FF	FF
3	FF	FF	FF	00	02	FF	FF	FF	FF	FF
4	FF	FF	FF	FF	00	FF	FF	FF	FF	50
5	FF	FF	05	FF	FF	00	10	FF	FF	FF
6	FF	FF	FF	FF	FF	FF	00	02	FF	FF
7	FF	FF	FF	FF	FF	FF	FF	00	10	05
8	FF	FF	FF	FF	FF	FF	FF	FF	00	05
9	FF	FF	FF	FF	FF	FF	FF	FF	FF	00

```

-----
--- final results (target node 9) ---
-----
node 0:  0 (expected: 0)
node 5:  5 (expected: 5)
node 2: 10 (expected: 10)
node 7: 15 (expected: 15)
node 9: 20 (expected: 20)
-----
>>> SUCCESS! <<< shortest path (20) found.
---- register results (dist_reg) ----
0->1 : 20
0->2 : 10
0->3 : 20
0->4 : 22
0->5 : 5
0->6 : 15
0->7 : 15
0->8 : 25
0->9 : 20

```

DURUM 2

Kaynak \ Hedef	0	1	2	3	4	5	6	7	8	9
0	00	0A	FF	FF	FF	05	FF	FF	FF	FF
1	FF	00	0A	FF	FF	FF	FF	FF	FF	FF
2	FF	FF	00	0A	FF	FF	FF	FF	FF	FF
3	FF	FF	FF	00	0A	FF	FF	FF	FF	FF
4	FF	FF	FF	FF	00	FF	FF	FF	FF	32
5	FF	FF	FF	FF	FF	00	05	FF	FF	FF
6	FF	FF	FF	FF	FF	FF	00	05	FF	FF
7	FF	FF	FF	FF	FF	FF	FF	00	05	FF
8	FF	FF	FF	FF	FF	FF	FF	FF	00	05
9	FF	FF	FF	FF	FF	FF	FF	FF	FF	00

```

-----
--- final results (target node 9) ---
-----
node 1: 10 (expected: 10)
node 4: 40 (expected: 40)
node 5:  5 (expected: 5)
node 8: 20 (expected: 20)
node 9: 25 (expected: 25)
-----
>>> SUCCESS! <<< shortest path (via path b: 25) found.
---- master register results (dist_reg) ----
0->1 : 10
0->2 : 20
0->3 : 30
0->4 : 40
0->5 : 5
0->6 : 10
0->7 : 15
0->8 : 20
0->9 : 25

```

DURUM 3

Kaynak \ Hedef	0	1	2	3	4	5	6	7	8	9
0	00	0A	FF	FF	FF	FF	FF	FF	FF	FF
1	FF	00	0A	FF	FF	FF	FF	FF	FF	FF
2	FF	FF	00	0A	FF	FF	FF	FF	FF	FF
3	FF	FF	FF	00	FF	FF	FF	FF	FF	FF
4	FF	FF	FF	FF	00	FF	FF	FF	FF	FF
5	FF	FF	FF	FF	FF	00	FF	FF	FF	FF
6	FF	FF	FF	FF	FF	FF	00	FF	FF	FF
7	FF	FF	FF	FF	FF	FF	FF	00	FF	FF
8	FF	FF	FF	FF	FF	FF	FF	FF	00	05
9	FF	FF	FF	FF	FF	FF	FF	FF	FF	00

```

-----
--- final results (target node 9) ---
-----
node 1: 10 (expected: 10)
node 3: 30 (expected: 30)
node 8: 255 (expected: 255/FF)
node 9: 255 (expected: 255/FF)
-----
>>> SUCCESS! <<< unreachable node kept as infinity (FF).

```

9. PROJE DOSYA LİNKİ

https://drive.google.com/drive/folders/1Ao04plv_MrYuy69ur5Dfc_2sjf1UNf4B?usp=drive_link