Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi
Pamukkale University Journal of Engineering Sciences

FPGA prototipleme: RISC-V tabanlı SoC için çip üstü hata ayıklama modülü ve Raspberry Pi ile openOCD JTAG üzerinden test süreçleri

FPGA prototyping: On-chip debug module for RISC-V based SoC and testing processes using raspberry Pi with OpenOCD over JTAG

Sezen BAL[1], Hayriye KORKMAZ[2], Doğukan BİÇER[3], Celal Alperen BAYAR[2], Eren KALE[2], Eray KAYAİLLİ[2], Armağan Bİ[2]*

[1]Bilgisayar Teknolojileri Bölümü, Teknik Bilimler Meslek Yüksekokulu, Marmara Üniversitesi, İstanbul, Türkiye.
sezen.bal@marmara.edu.tr
[2]Elektrik-Elektronik Mühendisliği Bölümü, Teknoloji Fakültesi, Marmara Üniversitesi, İstanbul, Türkiye.
hkorkmaz@marmara.edu.tr, alperenbayar22@marun.edu.tr, erenkale@marun.edu.tr, eraykayailli88@gmail.com, armaganbi1233@gmail.com
[3]Elektrik-Elektronik Mühendisliği Bölümü, Fen Bilimleri Enstitüsü, Marmara Üniversitesi, İstanbul, Türkiye.
dogukanbicer52@gmail.com

Geliş Tarihi/Received: 06.12.2024 Düzeltme Tarihi/Revision: 21.03.2025 doi: 10.5505/pajes.2025.95694
Kabul Tarihi/Accepted: 02.05.2025 Araştırma Makalesi/Research Article

Öz

Bu çalışma, RISC-V tabanlı bir System-on-Chip (SoC) tasarımının FPGA üzerinde prototipleme sürecini ve bu süreçte entegre edilen çip üstü hata ayıklama modülünü ele almaktadır. FPGA prototipleme, sayısal tasarımların donanım seviyesinde değerlendirilmesi amacıyla FPGA üzerinde uygulanması ve test edilmesi sürecidir. Çalışmada, Raspberry Pi ve OpenOCD kullanılarak JTAG aracılığıyla gerçekleştirilen hata ayıklama işlemleri, SoC'nin çeşitli fonksiyonlarını ve performansını etkin bir şekilde test etmek için kritik bir adım olarak vurgulanmaktadır. Bu yaklaşım, geliştirme sürecinde maliyetleri düşürmeye ve pazara sunma süresini kısaltmaya katkıda bulunurken, tasarımın son silikon versiyonuna geçişte karşılaşılacak riskleri azaltmaktadır. SoC tasarımı, FPGA üzerinde çalıştırılarak fonksiyonel doğrulama kapsamında çeşitli testlere tabi tutulmuş ve başarılı sonuçlar elde edilmiştir. Fonksiyonel doğrulama, bir sayısal tasarımın beklenen davranışları sergileyip sergilemediğini kontrol etmek için uygulanan kritik bir test aşamasıdır. Kayan nokta birimi testi kapsamında sinüs fonksiyonu uygulanarak sinüs değerleri hesaplanmış ve birimin doğru çalıştığı doğrulanmıştır. I2C çevre birimi testinde, bir sensör bağlanarak sensörden elde edilen verilerin doğru şekilde alındığı ve işlendiği tespit edilmiştir. Bit manipulation testi, RISC-V bit manipülasyon komutlarının doğru çalıştığını göstermiştir. FreeRTOS uygulamasında, görev zamanlaması, kesmeler ve kaynak yönetimi başarıyla gerçekleştirilmiş ve sistemin çoklu görev yürütme sırasında hatasız çalıştığı doğrulanmıştır. Atomik işlemler ile CSR (Control and Status Registers) birimlerinin doğru işleyişi değerlendirilmiş ve beklendiği gibi çalıştığı doğrulanmıştır. *Bunun yanı sıra, işlemcinin performansı CoreMark ve Dhrystone kıyaslamaları kullanılarak değerlendirilmiştir. CoreMark testinde işlemci, 41,97 iterasyon/saniye (600 iterasyon) skoruna ulaşmış ve CoreMark/MHz değeri 2,51 olarak hesaplanmıştır. Dhrystone kıyaslamasında ise işlemci, 70,582 Dhrystone/saniye performans sergilemiştir. Her bir bileşenin implementasyonu ve ilgili test tezgahları Verilog HDL ile yazılmış olup, tasarım Terasic De10-Lite FPGA üzerinde uygulanmıştır.*

Anahtar kelimeler: RISC-V İşlemci, FPGA, Hata Ayıklama, Çip üstü Sistem

Abstract

This paper presents the FPGA prototyping process of a RISC-V based System-on-Chip (SoC) design and the on-chip debug module integrated in this process. FPGA prototyping is the process of implementing and testing digital designs on an FPGA for hardware-level evaluation. In the study, debugging operations performed via JTAG using Raspberry Pi and OpenOCD are highlighted as a critical step to effectively test the various functions and performance of the SoC. This approach contributes to reduce costs in the development process and shorten time-to-market, while reducing the risks involved in the transition to the final silicon version of the design. The SoC design was executed on FPGA and subjected to various tests within the scope of functional verification and successful results were obtained. Functional verification is a critical testing phase to check whether a digital design exhibits the expected behavior. Within the scope of the floating point unit test, sine values were calculated by applying the sine function and the correct operation of the unit was verified. In the I2C peripheral test, a sensor was connected and it was determined that the data obtained from the sensor was received and processed correctly. The bit manipulation test showed that the RISC-V bit manipulation instructions worked correctly. In the FreeRTOS implementation, task scheduling, interrupts, and resource management were successfully implemented, and the system was verified to be error-free during multitasking. The correct operation of atomic operations and CSR (Control and Status Registers) was evaluated and verified to work as expected. In addition, the performance of the processor was evaluated using CoreMark and Dhrystone benchmarks. In the CoreMark test, the processor achieved a score of 41.97 iterations/second (600 iterations) and the CoreMark/MHz value was calculated as 2.51. In the Dhrystone benchmark, the processor performed 70.582 Dhrystones/second. The implementation of each component and the corresponding testbenches were wri tten in Verilog HDL and the design was implemented on a Terasic De10-Lite FPGA.

Keywords: RISC-V Processor, FPGA, Debug, System on Chip

1 Giriş

Gömülü sistem tasarım sürecinde hataların erken aşamalarda tespit edilmesi, geliştirme sürecini kısaltarak ürünün piyasaya sürülme süresini önemli ölçüde azaltmaktadır. Geleneksel hata ayıklama yöntemlerinden biri, tasarımın simülasyon aracılığıyla doğrulandığı ön-silikon doğrulama sürecidir. Bu süreçte giriş-çıkış portlarına ve hatta iç sinyallere dahi erişim sağlanabildiğinden, hatalar kolaylıkla belirlenip düzeltilebilmektedir. Ancak simülasyonlar, her zaman gerçek donanım koşullarını yeterince yansıtmayabilmektedir. Özellikle karmaşık sistemlerde simülasyona dayalı

*Yazışılan yazar/Corresponding author

doğrulamanın yetersiz kalabileceği durumlarda, ön-silikon doğrulama sürecinin bir parçası olarak FPGA prototipleme kullanılmaktadır [1]. FPGA prototipleme, ASIC (Application-Specific Integrated Circuit), ASSP (Application-Specific Standard Product) ve SoC (System on Chip) gibi özelleştirilmiş çip tasarımlarının geliştirilmesinde kritik bir role sahiptir. Bu yöntem, tasarımların fiziksel bir silikon çip üretilmesinden önce FPGA üzerinde uygulanarak doğrulanmasına ve test edilmesine olanak sağlamaktadır [2]. Bu doğrultuda, çip üstü hata ayıklama modülleri FPGA prototipleme sırasında işlemcinin iç durumunu izleme ve kontrol etme imkanı sunmaktadır [3].

RISC-V, açık kaynaklı bir işlemci mimarisi olarak, donanım tasarımında modülerlik ve özelleştirme imkanları sunarak teknoloji dünyasında önemli bir yer edinmiştir. Bu mimari, Berkeley'deki California Üniversitesi'nde başlatılan bir proje olup, işlemci tasarımı ve geliştirilmesi süreçlerini daha erişilebilir hale getirmeyi amaçlamaktadır [4]. RISC-V'nin temel avantajlarından biri, lisans ücreti gerektirmemesi ve kaynak kodlarının herkese açık olmasıdır. Bu özellikler, akademik ve ticari kullanımlar için ideal bir platform sağlamaktadır [5].

RISC-V Instruction Set Architecture (ISA), bu mimari tarafından tanımlanan komut setidir [6]. ISA, yazılımın donanım üzerinde nasıl çalıştırılacağını belirleyen temel komutlar ve protokoller bütünüdür. RISC-V ISA, temel tamsayı işlemleri (RV32I, RV64I gibi) sağlamanın yanı sıra, çeşitli işlem türleri için ek genişletmeler sunmaktadır. Bu genişletmeler arasında matematik işlemler, güvenlik işlevleri, sanallaştırma desteği ve çok daha fazlası bulunmaktadır. Özellikle bu çalışmada odaklanılan RISC-V 32 (RV32IMAFBC_Zicsr_Zifencei) konfigürasyonu, geniş bir işlevsellik yelpazesi sunmaktadır. Bu konfigürasyon; tamsayı matematik işlemleri (I), çarpma ve bölme (M), tek ve çift duyarlıklı kayan nokta işlemleri (F), bit maipüşasyon işlemleri (B), atomik işlemler (A), sıkıştırılmış komutlar (C), işlemci durum kayıtlarını yönetme (Zicsr) ve sıralama engelleri (Zifencei) gibi özellikleri içermektedir.

Bu geniş yetenek seti, RISC-V ISA'nın çeşitli teknoloji alanlarında yenilikçi çözümler sunmasını sağlamaktadır [7]. Özellikle yüksek performanslı bilgi işlem (HPC) için, Wang et al. tarafından önerilen Extended Base Global Address Space (xBGAS) genişlemesi gibi özel genişletmeler, düğümler arası bağlantılarla ilgili yazılım maliyetlerini azaltmayı amaçlamaktadır [8]. Sanallaştırma teknolojisinde, Chen et al. tarafından geliştirilen DuVisor, kullanıcı seviyesinde sanallaştırma işlemlerini çekirdeğe sıkışmadan yönetebilmenin yollarını araştırmaktadır [9]. Elsabbagh ve arkadaşları tarafından geliştirilen Vortex genel amaçlı GPU'su, RISC-V tabanlı SIMT (Single Instruction, Multiple Threads) yürütme modelini destekleyecek yeni komutlar sunarak, RISC-V'nin grafik hesaplama yeteneklerini artırarak multimedya uygulamaları için daha etkili çözümler sağlamaktadır [10].

RISC-V, açık hata ayıklama standardı olan RISC-V Debug Specification ile donatılmıştır [11]. Bu spesifikasyon, işlemcilerin hata ayıklama arayüzlerini ve işlevlerini tanımlayarak hata ayıklama süreçlerinin standartlaştırılmasına olanak tanımaktadır. Ayrıca, geliştiricilere sistemin iç durumlarını izleme, hata ayıklama komutlarını iletebilme ve işlemci üzerinde kontrol sağlama gibi imkanlar sunmaktadır [12].

Bu çalışmada incelenen RISC-V tabanlı SoC'ye entegre edilen çip üstü hata ayıklama modülü, işlemcinin iç sinyallerini ve durumlarını dış bir cihazdan izlemeyi mümkün kılmaktadır.

Donanım ve yazılım arasında bağlantı kurarak hata ayıklama verilerine erişim sağlamak için JTAG standardı kullanılmaktadır [13]. IEEE 1149.1 standardına dayalı bu arayüz, elektronik sistemlerin güvenilirliğini ve işlevini sağlamak için gerekli olan cihaz programlama, teşhis ve baskılı devre kartlarının test edilmesine olanak tanımaktadır [14].

Bu entegrasyon, işletim sistemi desteği gerektirmeyen bare-metal sistemlerde hata ayıklama işlemlerini daha verimli hale getirmektedir [15]. Bare-metal sistemler, bir bilgisayar veya gömülü sistemde işletim sistemi katmanı olmaksızın, donanım üzerinde doğrudan çalışan yazılımları ifade etmektedir. Çoğu zaman, gerçek zamanlı performans gereksinimleri nedeniyle işlemcinin çalışmasının durdurulması mümkün olmadığı için, işlemcinin iç durumunun gerçek zamanlı olarak izlenmesi büyük önem taşımaktadır. Gerçekleştirilen gerçek zamanlı izleme yöntemi, işlemcinin belirli sinyallerini ve işlem akışını kesintisiz olarak gözlemlemeyi sağlamaktadır.

OpenOCD (Open On-Chip Debugger), gömülü hedef cihazlar için hata ayıklama, sistem içi programlama ve sınır tarama (boundary scan) testi sağlayan açık kaynaklı bir araçtır [16]. Bir ana bilgisayar ile hedef cihazdaki JTAG arayüzü arasında bir köprü görevi görmektedir ve geliştiricilerin ürün yazılımlarını doğrudan geliştirdikleri donanım üzerinde dağıtmalarına ve hata ayıklamalarına olanak tanımaktadır. OpenOCD çok çeşitli geliştirme kartlarını ve hata ayıklama protokollerini desteklemektedir, bu da onu çok çeşitli donanım geliştirme görevleri için çok yönlü bir araç haline getirmektedir.

Raspberry Pi bu çalışmada hata ayıklama ana bilgisayarı olarak işlev görerek OpenOCD aracılığıyla JTAG üzerinden SoC ile iletişim kurmaktadır.

Shan Gao ve arkadaşları tarafından sunulan çalışmada, RISC-V işlemcisi üzerine kurulu bir hızlı yerinde hata ayıklama tasarımı geliştirilmiştir; bu tasarım JTAG arayüzü kullanılarak yapılmıştır [17]. Onur Toker tarafından sunulan çalışmada ise, Vivado yüksek seviye sentez (HLS) aracı kullanılarak RISC-V RV32I tabanlı bir sistem-üzeri-çip (SoC) tasarım yaklaşımı geliştirilmiştir ve düşük maliyetli bir FPGA kartı üzerinde test edilmiştir [18].

Bu çalışma, özellikle karmaşık sistemlerde geleneksel simülasyona dayalı doğrulama yöntemlerinin yetersiz kaldığı durumları ele almaktadır. Gelişmiş SoC tasarımlarında hata ayıklama ve doğrulama süreçlerinde yeterince etkili olamayan bu yöntemlerin yerine, ön-silikon doğrulama sürecinin bir parçası olarak FPGA prototipleme önerilmektedir. Bu çözüm, gerçek donanım koşullarında kapsamlı testler sağlayarak daha etkin bir doğrulama ve hata ayıklama imkanı sunmaktadır. RISC-V tabanlı SoC'nin çeşitli fonksiyonlarının ve performansının Raspberry Pi ve OpenOCD aracılığıyla JTAG üzerinden gerçekleştirilen hata ayıklama işlemleri sayesinde etkin bir şekilde test edilmesine olanak tanımaktadır.

Çalışmanın geri kalan bölümü, FPGA tabanlı prototipleme ve hata ayıklama süreçlerinin detaylarını ve sonuçlarını sunmaktadır. İkinci bölüm, SoC'nin ana bileşenlerini tanıtırken,

üçüncü bölüm hata ayıklama süreçlerini detaylandırmaktadır. Dördüncü bölümde, bu süreçler çerçevesinde yapılan testler ve sonuçlar tartışılmakta, beşinci ve son bölümde ise elde edilen bulgular değerlendirilerek gelecekteki uygulamalar için öneriler sunulmaktadır.

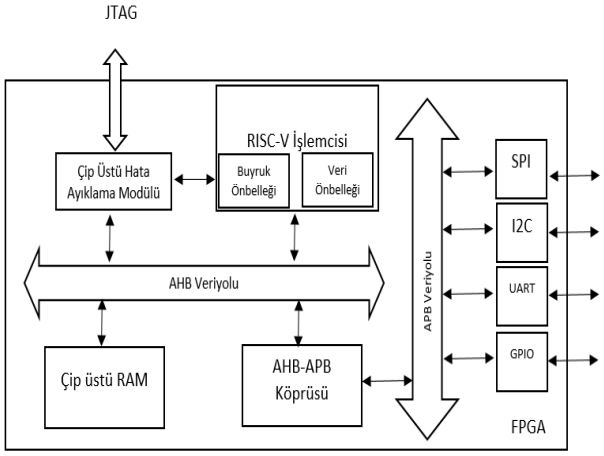
2 SoC Bileşenleri

Şekil 1'deki blok diyagram, FPGA üzerinde bir RISC-V System-on-Chip (SoC) mimarisinin temel bileşenlerini, çip üstü hata ayıklama modülünü ve bu bileşenler arasındaki iletişim yollarını göstermektedir.

Tasarlanan RISC-V işlemcisi, buyruk ve veri önbelleği olarak iki ayrı önbellek bloğuna sahiptir. Bu önbellekler, işlemcinin verimliliğini artırmak ve gerekli verileri hızlı bir şekilde erişilebilir kılmak için kullanılmaktadır. Çip üstü hata ayıklama modülü, JTAG arayüzü üzerinden dışarıdan gelen hata ayıklama komutlarını almaktadır ve işlemci üzerinde uygun hata ayıklama işlemlerini gerçekleştirmektedir. Çip üstü RAM bloğu, işlemci tarafından hızlı veri depolama ve erişim için kullanılmaktadır. AHB (Advanced High-performance Bus) – APB (Advanced Peripheral Bus) köprüsü, farklı hızlarda çalışan iki veriyolu olan AHB ve APB arasında veri aktarımını sağlamaktadır. Köprü, sistem içi iletişimde verimliliği artırarak farklı bileşenler arası veri akışını düzenlemektedir.

AHB veriyolu, yüksek performanslı veri transferleri için tasarlanmıştır ve işlemci ile sistemdeki diğer yüksek hızlı bileşenler arasında veri taşıma işlevini üstlenmektedir.

SoC; SPI, I2C, UART ve GPIO gibi çeşitli standart çevresel ara birimlerle donatılmıştır. Bu ara birimler, FPGA'nın dış dünya ile etkileşimini sağlamaktadır ve çeşitli giriş/çıkış işlemleri için kullanılmaktadır.



Şekil 1. SoC Blok diyagram
Figure 1. SoC Block diagram

Bu çalışmada gerçekleştirilen RISC-V tabanlı SoC'e ait teknik özellikleri Tablo 1'de sıralanmıştır.

Tablo 1. RISC-V Tabanlı SoC Teknik Özellikleri

Table 1. RISC-V Based SoC Technical Specifications

Özellikler	Özellik Değeri
Buyruk Kümesi	RISC-V 32 (RV32IMAFBC_Zicsr_Zifence)
FPGA Kartı	Terasic De10-Lite
Donanım Tanımlama Dili	Verilog HDL
Çevre Birimleri	UART, GPIO, I2C, SPI
Veriyolu	AHB ve APB
Boru Hattı Aşama Sayısı	5 aşamalı CPU, 4 aşamalı FPU
Veri Önbelleği	48KB
Buyruk Önbelleği	128KB

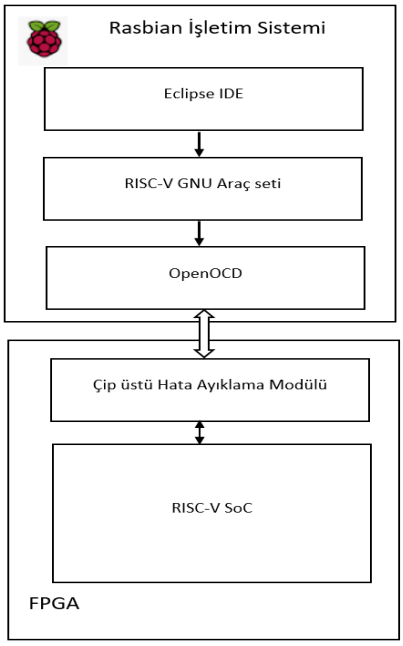
Bu doğrultuda, Şekil 2'de görüldüğü gibi tasarlanan SoC, FPGA üzerinde bulunan mantık elemanlarının %98.46'sı (48.552/49.760) aktif olarak kullanılmaktadır.

Family	MAX 10
Device	10M50DAF484C7G
Timing Models	Final
Total logic elements	48,522 / 49,760 (98 %)
Total registers	9846
Total pins	162 / 360 (45 %)
Total virtual pins	0
Total memory bits	918,528 / 1,677,312 (55 %)
Embedded Multiplier 9-bit elements	40 / 288 (14 %)
Total PLLs	1 / 4 (25 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 2 (0 %)

Şekil 2. Kaynak Kullanımı
Figure 2. Resource Utilization

3 RISC-V SoC için Hata Ayıklama Süreci

Şekil 3, FPGA tabanlı bir RISC-V System-on-Chip (SoC) üzerinde hata ayıklama sürecini tanımlamaktadır. Süreç, Raspbian işletim sistemi yüklü bir Raspberry Pi kullanılarak, Eclipse Integrated Development Environment (IDE) aracılığıyla yazılım kodunun geliştirilmesi ile başlamaktadır. Geliştirilen kod, RISC-V GNU Araç Seti kullanılarak derlenmektedir. OpenOCD, JTAG protokolünü kullanarak, çip üstü hata ayıklama modülüne erişim sağlamaktadır ve bu modül aracılığıyla derlenen programın yüklenmesini ve çalıştırılmasını koordine etmektedir. Çip üstü hata ayıklama modülü, yazılımın çalışma zamanı sırasında veri akışını izleme ve sistemin durumunu kontrol etme yeteneği sağlamaktadır. Böylece, hata ayıklama süreci sırasında programın davranışı üzerinde detaylı analizler gerçekleştirilebilmektedir.



Şekil 3. Hata ayıklama blok şeması
Figure 3. Debugging block diagram

Bu çalışma kapsamında, Tablo 2'de bulunan Raspberry Pi 4 Model B'nin GPIO pinleri, RISC-V tabanlı FPGA tasarımı üzerinde kapsamlı hata ayıklama ve programlama işlemleri gerçekleştirmek amacıyla JTAG arayüzü olarak yapılandırılmıştır. Geleneksel JTAG bağlantısının dört temel pini olan TCK (Test Clock), TMS (Test Mode Select), TDI (Test Data Input) ve TDO (Test Data Output), JTAG protokolünün ana işlevselliğini sağlamak üzere kullanılmıştır.

Tablo 2. Raspberry Pi 4 Model B'nin GPIO Pinlerinin OpenOCD JTAG Hata Ayıklayıcı Olarak Atanması

GPIO Pinleri	JTAG Pin
GPIO4	TDI
GPIO5	TDO
GPIO12	TMS
GPIO13	TCK

4 TESTLER

Bu bölümde, hata ayıklama süreci çerçevesinde işlemcinin çeşitli birimlerinin fonksiyonel doğruluğunu saptamak amacıyla yürütülen deneysel testler ele alınmaktadır.

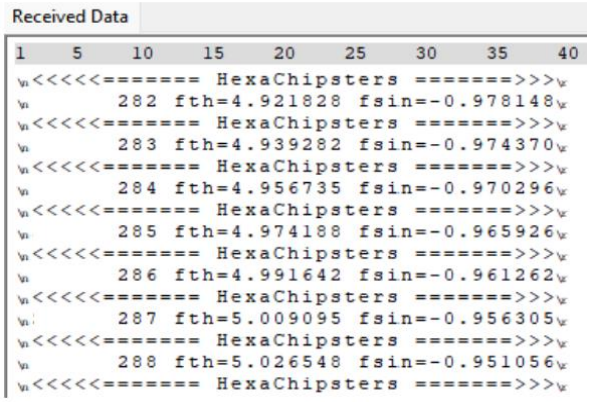
4.1 Kayan Nokta Birimi Testi

Bu test, işlemcinin Kayan Nokta Birimi (Floating Point Unit - FPU)'ni test etmektedir. Test kapsamında, FPU'nun matematiksel hesaplamaları ne derece doğru gerçekleştirebildiği değerlendirilmekte olup, özellikle sinüs fonksiyonu üzerinden örneklemeler yapılmaktadır. Program, Tablo 3'te belirlenen açılara karşılık gelen sinüs değerlerini hesaplayarak bu sonuçları seri terminale aktarmaktadır. Geliştirilen FPU biriminin performansı ve doğruluğu, seri terminal üzerinden alınan çıktılarla değerlendirilmekte ve elde edilen veriler Şekil 4'te görselleştirilmektedir.

Tablo 3. FPU birimi ile hesaplanan Sinüs fonksiyonun Değerleri

Table 3. Sine Function Values Calculated by the FPU Unit

Derece	Radyan (teta)	Sinüs değeri (sin(teta))
282	4.921828	-0.978148
283	4.939282	-0.974370
284	4.956735	-0.970296
285	4.974188	-0.965926
286	4.991642	-0.961262
287	5.009095	-0.956305
288	5.026548	-0.951056

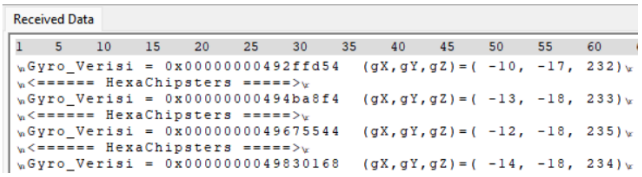


Şekil 4. Seri terminal üzerinden alınan sinüs fonksiyonun değerleri
Figure 4: Sine Function Values Received from the Serial Terminal

Hata payları, işlemcinin sinüs fonksiyonunu hesaplama kabiliyetinin doğruluğunu yansıtan önemli bir göstergedir. Testler sonucunda, FPU biriminin genel olarak 10^{-7} ve 10^{-8} mertebesinde oldukça düşük hata payları ile sinüs değerlerini hesapladığı gözlemlenmiştir. Bu hata oranları, birimin yüksek düzeyde doğruluk ve performansla çalıştığını göstermektedir.

4.2 I2C Çevre Birimi Testi

Bu test, I2C birimi aracılığıyla ADXL345 ivmeölçer sensöründen veri okuma işlemine odaklanmaktadır. ADXL345, üç eksenli bir hareket algılama sensörü olup, I2C arayüzü yoluyla veri alışverişinde bulunmaktadır. Bu deneme, I2C biriminin fonksiyonel doğruluğunu ve sensörden elde edilen verilerin uygun şekilde işlenip işlenmediğini değerlendirmeyi amaçlamaktadır. Test sonuçları ve elde edilen verilerin doğruluğu, seri terminal üzerinden alınan çıktılar Şekil 5'te görselleştirilmiştir.



Şekil 5. I2C birimi üzerinden toplanan ivmeölçer verileri
Figure 5. Accelerometer data collected via I2C unit

Test süreci, I2C haberleşme protokolünün fonksiyonel doğruluğunu değerlendirmek amacıyla gerçekleştirilmiştir. Test kapsamında, ADXL345 ivmeölçer sensörü ile işlemci arasındaki veri alışverişi analiz edilerek, protokolün güvenilirliği ve stabilitesi incelenmiştir. Öncelikle, I2C iletişiminin başarıyla kurulduğu doğrulanmış ve sensörün kimlik doğrulama (WHO_AM_I) register'ından okunan veri, üretici tarafından belirlenen referans değerleriyle karşılaştırılmıştır. I2C protokolünün güvenilirliği, ACK/NACK sinyallerinin analizi ile incelenmiş ve her veri iletimi sonrasında beklenen ACK sinyallerinin alındığı gözlemlenmiştir.

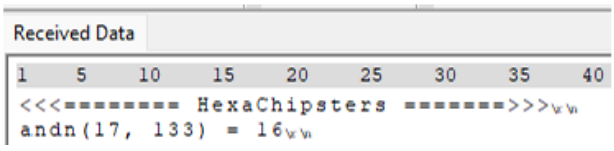
Ardından ADXL345 ivmeölçer sensörü, durağan bir yüzeye yerleştirilerek herhangi bir hareket uygulanmadan ölçümler alınmış ve yerçekimi ivmesinin etkileri göz önünde bulundurularak X, Y ve Z eksenlerinden elde edilen değerlerin beklenen sınırlar içinde olduğu doğrulanmıştır. Ayrıca, sensörden alınan bilgilerin doğruluğunu test etmek amacıyla dinamik hareket testleri de gerçekleştirilmiştir. Sensör farklı yönlerde belirli açılarla döndürülerek ve belirli hızlarda hareket ettirilerek, sensörden gelen verilerin fiziksel hareketle uyumlu olup olmadığı değerlendirilmiştir. Hareketin yönüne ve hızına bağlı olarak X, Y ve Z eksenlerindeki ivme değerlerinin değişimi gözlemlenmiş ve ölçümlerin beklendiği gibi olduğu doğrulanmıştır. Bu, I2C veri iletiminin yalnızca kesintisiz olmadığını, aynı zamanda sensörden alınan verilerin de fiziksel gerçeklikle uyumlu olduğunu kanıtlamaktadır.

4.3 Bit Manipülasyon İşlemlerinin Doğrulanması

Şekil 6'dan da görüldüğü gibi bu test, işlemci üzerinde çalıştırılan ve bit manipülasyon komutlarını içeren örnek bir Inline Assembly koduyla gerçekleştirilmiştir. Inline Assembly, düşük seviyeli assembly dilinde yazılmış kod parçalarının yüksek seviye programlama dilleri içerisine entegre edilmesi sürecidir. Kodun işlevselliği, UART çıkışı üzerinden elde edilen verilerle detaylı bir şekilde kontrol edilmiş ve elde edilen sonuçlar Şekil 7'de görselleştirilmiştir.

```
uint32_t a = 0x11;   // 'a' değişkeni 0x11 (17)
uint32_t b = 0x85;   // 'b' değişkeni 0x85 (133)
uint32_t result;     // Sonucun atanacağı değişken
asm volatile ("andn %0, %1, %2" : "=r" (result) : "r" (a), "r" (b));
printf("andn(%d, %d)= %d\n", a, b, result;
```

Şekil 6. Örnek İnline Assembly kodu
Figure 6. Sample Inline Assembly code



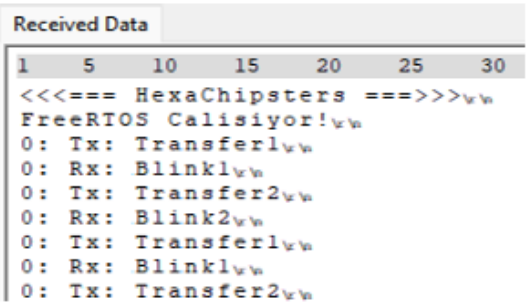
Şekil 7. Bit Manipülasyon işleminin UART birimine aktarılan çıktısı

Figure 7. Output of the Bit Manipulation process transferred to the UART unit

Bit seviyesinde işlem yapmayı sağlayan ANDN (Bitwise AND NOT) komutu, belirlenen değişkenler (a ve b) üzerinde uygulanmış ve sonuç, UART çıkışı üzerinden elde edilen verilerle doğrulanmıştır. Burada, a değişkeninin tüm bitleri ters çevrilmekte (NOT işlemi uygulanmakta) ve ardından b değişkeni ile bitwise AND işlemi yapılmaktadır. Matematiksel olarak hesaplanan değer 16 (0x10)'dur. Şekil 7'de UART üzerinden alınan çıktı da andn(17, 133) = 16 olarak gözlemlenmiştir. Kodun farklı değişken değerleri ile test edilmesi sonucu, ANDN komutunun işlemci üzerinde tutarlı bir şekilde çalıştığı görülmüştür.

4.4 FreeRTOS Uygulaması ve Atomik İşlemler Testi

Bu test, işlemci üzerinde FreeRTOS'un çalışmasını ve Atomik işlemler ile CSR (Control and Status Registers) birimlerinin doğru işleyişini değerlendirmektedir. FreeRTOS, gömülü sistemler için tasarlanmış gerçek zamanlı bir işletim sistemidir. Test kapsamında, tipik bir FreeRTOS uygulaması çalıştırılmış. Çalıştırılan uygulama, atomik işlemler ile CSR kullanımlarını içermektedir. FreeRTOS uygulamasında 3 ayrı görev çalıştırılmıştır: Birinci görev UART birimindeki alma (receive) işlemidir. İkinci görev UART birimindeki gönderme (transmit) işlemidir. Üçüncü görev donanım kesme (IRQ) kontrol işlemidir.



Şekil 8. FreeRTOS'un görevlerinin UART biriminde gösterilmesi
Figure 8. Representation of FreeRTOS tasks in the UART unit

Şekil 8'deki test sonuçları, FreeRTOS'un çalışma durumunun başarılı bir şekilde doğrulandığını göstermektedir. Terminal çıktılarında görülen "FreeRTOS Çalışıyor!" mesajı, sistemin başlatıldığını ve işletim sisteminin görev yönetim mekanizmasının aktif hale geldiğini kanıtlamaktadır. UART üzerinden elde edilen verilere göre, FreeRTOS üzerinde çalışan belirli görevlerin düzenli bir şekilde yürütüldüğü gözlemlenmiştir. Transmit (Tx) görevleri "Transfer1" ve "Transfer2" mesajlarını belirli bir sırayla gönderirken, Receive (Rx) görevleri "Blink1" ve "Blink2" mesajlarını almaktadır. Bu düzenli veri akışı, FreeRTOS'un görev planlayıcısının (scheduler) periyodik olarak görevleri çalıştırdığını ve sistemde tanımlanan görevlerin planlanan önceliklere göre yürütüldüğünü göstermektedir. Bunun yanı sıra, terminal çıktılarında gözlemlenen "Rx: Blink1" ve "Rx: Blink2" mesajlarının yalnızca butona basıldığında ortaya çıkması, sistemde bir donanım kesme (IRQ) mekanizmasının aktif olarak çalıştığını kanıtlamaktadır.

4.5 Coremark ve Dhrystone Testi

Bu aşamada, işlemcinin performansını değerlendirmek amacıyla özel test programları (benchmarklar) kullanılmıştır [19, 20]. Bu benchmarklar, işlem süresi, bellek kullanımı ve diğer kritik parametreleri ölçmek için tasarlanmıştır. Dhrystone, özellikle tam sayı işlemleri üzerine odaklanmış bir benchmark olup, kayan nokta işlemleri içermez ve tam sayı işlemleri yapan işlemcilerin performansını ölçmek için kullanılmaktadır. Ancak, Dhrystone testi bazı sınırlamalara sahiptir ve derleyici optimizasyonlarının etkisine oldukça duyarlıdır. Bu nedenle, işlemci performansını daha kapsamlı değerlendirmek için CoreMark benchmarkı tercih edilmiştir. CoreMark, matris işlemleri, veri bütünlüğü hesaplamaları ve tam sayı çarpma ve bölme işlemleri gibi çeşitli işlemleri içeren bir dizi iterasyonu kapsamaktadır. CoreMark, Dhrystone'a kıyasla daha objektif bir değerlendirme yapabilmekte ve işlemcinin öngörülen kullanım alanlarındaki işlem türlerine daha uygun bir test yöntemi sunmaktadır. Ayrıca CoreMark, derleyici optimizasyonlarının test sonuçlarına etkisine daha az duyarlıdır. Bu özellikleri ile CoreMark, daha tutarlı ve karşılaştırılabilir sonuçlar elde edilmesine olanak tanımaktadır.

Tablo 4, çeşitli mikroişlemci ve mikrodenetleyici birimlerinin saat hızı başına düşen CoreMark performanslarını karşılaştırmaktadır. CoreMark, saniyede gerçekleştirilen iterasyon sayısını ölçen bir kıyaslama testi olarak kullanılmıştır. Ancak bu parametre işlemci frekansına bağlı olduğundan, farklı işlemcilerin doğrudan karşılaştırılması için ideal olmayabilir. Bu nedenle, performans değerlendirmeleri genellikle CoreMark/MHz olarak ifade edilmekte, bu hesaplama yöntemi farklı işlem hızlarına sahip cihazlar arasında daha adil bir karşılaştırma imkanı sunmaktadır.

Bu çalışmada performans değerlendirmesi, CoreMark ve Dhrystone kıyaslama testleri kullanılarak gerçekleştirilmiştir. CoreMark testi sonucunda, işlemci 16,66 MHz saat frekansında 600 iterasyon üzerinden 41,97 iterasyon/saniye değeri elde etmiştir. Elde edilen CoreMark/MHz oranı 2,51 olarak hesaplanmıştır. Bu sonuç, riscv64-unknown-elf-gcc derleyicisinin 8.6.0 versiyonu ile elde edilmiştir.

Dhrystone testi sonucunda ise işlemci, 70,582 Dhrystone/saniye işlem kapasitesine ulaşmıştır. Her bir Dhrystone işleminin tamamlanması için gereken süre 14,167 mikrosaniye olarak ölçülmüştür. İşlemcinin saat hızı 16,66 MHz olarak belirlenmiştir.

Bu çalışmada, işlemcinin performansını değerlendirmek amacıyla elde edilen sonuçlar, farklı mimarilere sahip gömülü işlemcilerle karşılaştırılmıştır. Karşılaştırma için seçilen işlemciler, gömülü sistemlerde yaygın olarak kullanılan ve farklı performans seviyelerini temsil eden işlemciler arasından belirlenmiştir. Güncel mikroişlemcilerin CoreMark/MHz değerlerine erişmek için Embedded Microprocessor Benchmark Consortium (EEMBC) tarafından sunulan resmi veritabanı, bağımsız araştırmacılar tarafından yürütülen testler ve açık kaynak toplulukları tarafından paylaşılan benchmark sonuçları da güncel işlemcilerin performans verilerini sağlamaktadır. [21,22].

Tablo 4'te belirtildiği üzere, çeşitli kartlar arasında yapılan karşılaştırmada; Atmega 2560 işlemcisine sahip Arduino MEGA 2560; 0,44 CoreMark/MHz değeri ile en düşük performansı sergilerken, ARM Cortex-M3 işlemcili STM32F103C8T6 ve ARM Cortex-M4 işlemcili STM32F411CEU6 sırasıyla 1,13 ve 1,79 CoreMark/MHz değerleri ile orta düzey performans göstermiştir [21,22]. Bu çalışmada geliştirilen RISC-V tabanlı işlemci, ESP8266 ve ARM11 işlemcili Raspberry Pi Model B v2 gibi yüksek performans gösteren diğer gömülü sistemlerle rekabet edebilecek bir performans sergilemiştir [21,22]. Bu performans düzeyi, işlemciyi hem çok yönlü gömülü sistem uygulamaları hem de yüksek veri işleme gereksinimi duyan gelişmiş projeler için uygun bir seçenek haline getirmektedir.

Tablo 4. Performans Karşılaştırması (CoreMark/MHz)

Table 4. *Performance Comparison (CoreMark/MHz)*

Kart Modeli	İşlemci	Frekans	CoreMark/ MHz
Arduino MEGA 2560	Atmega 2560	16	0,44
STM32F103C8T6	ARM Cortex-M3	72	1,13
STM32F401CCU6	ARM Cortex-M4	84	1,79
STM32F411CEU6	ARM Cortex-M4	172	1,72
Bu çalışma	**RISC-V**	**16,66**	**2,51**
Raspberry Pi Model B v2	ARM11	700	2,25
	ESP8266	80	2,375

Dhrystone benchmark performans hesaplamalarında yaygın olarak kullanılan VAX-11/780 işlemcisi baz alınarak 1 DMIPS = 1757 Dhrystone/s referans değeri kullanılarak işlemcinin Dhrystone MIPS (DMIPS) değeri hesaplanmıştır [23]. Bu çalışmada, işlemcinin 40,17 DMIPS performansı sunduğu hesaplanmıştır. Elde edilen sonuçlar, bu çalışöadaki işlemcinin DMIPS ve DMIPS/MHz değerleri açısından gömülü sistemlerde yaygın olarak kullanılan bazı işlemcilerle Tablo 5'teki gibi karşılaştırılmıştır [24-28].

Tablo 5. Performans Karşılaştırması (DMIPS & DMIPS/MHz)

Table 5. *Performance Comparison (*DMIPS & DMIPS/MHz*)*

İşlemci	Saat Hızı(MHz)	DMIPS/MHz	Toplam DMIPS
ARM Cortex-M3	50 MHz	1,50	75,0
ARM Cortex-M4	80 MHz	1,52	121,6
PicoRV32	100 MHz	0,355	35,5
ORCA	100 MHz	1,589	158,9
Bu çalışma	16.66	2,41	40,17

Ancak toplam işlem kapasitesi değerlendirilirken yalnızca DMIPS/MHz değil, işlemcinin saat frekansının da dikkate alınması gerekmektedir. Daha bütüncül bir değerlendirme için,

çeşitli kriterlerin birlikte ele alınması gerekmektedir. Örneğin, DMIPS/MHz, saat hızından bağımsız olarak işlemcinin işlem verimliliğini gösterirken; toplam DMIPS, işlemcinin belirli bir frekansta sunabildiği mutlak işlem gücünü ifade eder. DMIPS/Watt, özellikle IoT ve mobil sistemler gibi düşük güç tüketimi gerektiren uygulamalarda, güç verimliliğini değerlendirmek için önemlidir [24-26].

5 Sonuçlar

Bu çalışmada gerçekleştirilen FPGA prototipleme, RISC-V tabanlı SoC tasarımlarının ön-silikon doğrulama sürecinde, geleneksel yöntemlere kıyasla gerçek donanım koşullarında kapsamlı testler sağlaması açısından önemli avantajlar sunmaktadır. Elde edilen test sonuçları, daha önce sunulan CoreMark test sonuçlarıyla uyumlu olarak, tasarlanan sistemin yüksek performansını ve kararlılığını kanıtlamaktadır. Çalışmada incelenen RISC-V tabanlı SoC, diğer yüksek performanslı gömülü sistemlerle rekabet edebilecek düzeyde performans sergilemiş, bu da gerçeklenen SoC'nin çok yönlü gömülü sistem uygulamaları için uygun bir seçenek olduğunu göstermektedir.

Bu sonuçlar, FPGA prototipleme kullanılarak, nihai silikon tasarıma geçiş öncesinde karşılaşılabilecek risklerin minimize edilmesine ve geliştirme maliyetlerinin azaltılmasına olanak tanıdığını vurgulamaktadır. Ancak bu yöntem, geleneksel simülasyon tekniklerine göre daha yüksek başlangıç yatırımı gerektirir. Başlangıçta daha büyük bir maliyet gibi görünse de, prototipleme sürecinde erken hata tespiti yapılabilmesi ve gerçek donanım üzerinde doğrulama yapılabilmesi, uzun vadede tasarımın daha ekonomik ve güvenilir olmasını sağlamaktadır. Bu nedenle, başlangıçtaki yüksek yatırım maliyeti, nihai ürünün kalitesini ve piyasaya sürülme süresini iyileştirerek telafi edilebilir. Bu da özellikle karmaşık sistemlerde ve yüksek güvenilirlik gerektiren uygulamalarda kritik bir avantaj olarak değerlendirilmektedir.

Bu çalışmalarda, FPGA prototipi sabit bir saat frekansı altında test edilmiş olup, daha ileri değerlendirmelerde farklı frekans ve termal koşullarda sistemin davranışı incelenerek kararlılık ve ölçeklenebilirlik analizleri yapılması planlanmaktadır.

Gelecekteki çalışmalar için, RISC-V mimarisinin modüler yapısından yararlanarak yapay zeka ve derin öğrenme uygulamalarına yönelik özelleştirilmiş işlemci çekirdeklerinin tasarımı ve geliştirilmesi önerilmektedir. Ayrıca, özellikle IoT cihazları için düşük güç tüketimine sahip, enerji verimliliği yüksek işlemci çekirdeklerinin geliştirilmesi, bu mimarinin kullanım alanını genişletecek ve endüstriyel ile tüketici elektroniği pazarlarında önemli etkiler yaratabilecektir.

6 Conclusions

This study highlights significant advantages of FPGA prototyping in the pre-silicon validation process of RISC-V based SoC designs, offering comprehensive testing in real hardware conditions compared to traditional methods. The test results obtained are consistent with previously presented CoreMark test outcomes, demonstrating the high performance and stability of the designed system. The RISC-V based SoC examined in this study has shown performance levels competitive with other high-performance embedded systems, indicating that the realized SoC is a suitable option for versatile embedded system applications.

These findings emphasize that FPGA prototyping can minimize risks encountered before transitioning to final silicon design and reduce development costs. However, this method requires a higher initial investment compared to traditional simulation techniques. Although this may seem like a higher cost at the outset, the ability to detect errors early in the prototyping process and validate on actual hardware ensures that the design is more economical and reliable in the long run. Therefore, the initial high investment cost can be justified by improving the quality of the final product and the time to market. This is especially critical in complex systems and applications requiring high reliability.

In this study, the FPGA prototype was tested under a fixed clock frequency, and further evaluations are planned to analyze the stability and scalability by examining the behavior of the system under different frequency and thermal conditions.

For future studies, it is recommended to utilize the modular structure of the RISC-V architecture to design and develop customized processor cores for artificial intelligence and deep learning applications. Additionally, the development of low power consumption, high energy efficiency processor cores, particularly for IoT devices, will expand the usage of this architecture and could have significant impacts in industrial and consumer electronics markets.

7 Yazar katkı beyanı

Gerçekleştirilen çalışmada Yazar 1 literatür taraması ve elde edilen sonuçların değerlendirilmesi; Yazar 2 yazım denetimi ve içerik açısından makalenin kontrol edilmesi; diğer yazarlar tasarımın gerçekleştirilmesi ve test aşamalarında katkı sunmuşlardır.

8 Etik kurul onayı ve çıkar çatışması beyanı

Hazırlanan makalede etik kurul izni alınmasına gerek yoktur. Hazırlanan makalede herhangi bir kişi/kurum ile çıkar çatışması bulunmamaktadır.

9 Kaynaklar

[1] Farooq U, Mehrez H. "Pre-Silicon Verification Using Multi-FPGA Platforms: A Review". *Journal of Electron Testing*, 37(1), 7-24, 2021.

[2] Korkmaz N. "Fitzhugh-Nagumo nöron modelinin rotasyon-geçiş prosedürü ve donanım doğrulaması". *Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi*, 24(3), 316-323, 2024.

[3] Zang Z, Liu Y, Cheung RCC. "Reconfigurable RISC-V Secure Processor And SoC Integration". *2019 IEEE International Conference on Industrial Technology (ICIT)*, Melbourne, Australia, 13 - 15 February 2019.

[4] University of California, Berkeley. "RISC-V Project." [Online]. Available: https://riscv.org/about/history/ (07.01.2024).

[5] Cui, Enfang & Li, Tianzheng & Wei, Qian. RISC-V Instruction Set Architecture Extensions: A Survey. IEEE Access, 2023. PP. 1-1. 10.1109/ACCESS.2023.3246491.

[6] RISC-V ISA Manual. Erişim adresi https://github.com/riscv/riscv-isa-manual (Erişim tarihi: 12.01.2025)

[7] E. Cui, T. Li and Q. Wei, "RISC-V Instruction Set Architecture Extensions: A Survey," in *IEEE Access*, vol. 11, pp. 24696-24711, 2023, doi: 10.1109/ACCESS.2023.3246491.

[8] X. Wang et al., "xBGAS: A Global Address Space Extension on RISC-V for High Performance Computing," 2021 IEEE International Parallel and Distributed Processing Symposium (IPDPS), Portland, OR, USA, 2021, pp. 454-463, doi: 10.1109/IPDPS49936.2021.00054.

[9] J. Chen, D. Li, Z. Mi, Y. Liu, B. Zang, H. Guan, and H. Chen, "Du-visor: a user-level hypervisor through delegated virtualization," 2022,arXiv:2201.09652.

[10] Elsabbagh, Fares & Tine, Blaise & Roshan, Priyadarshini & Lyons, Ethan & Kim, Euna & Shim, Da Eun & Zhu, Lingjun & Lim, Sung & kim, Hyesoon. "Vortex: OpenCL Compatible RISC-V GPGPU". 2020, 10.48550/arXiv.2002.12151.

[11] RISC-V International. "RISC-V Debug Specification".https://riscv.org/wp-content/uploads/2019/03/riscv-debug-release.pdf (07.01.2024)

[12] Poli L., Saha S., Zhai X., Mcdonald-Maier K.D. "Design and Implementation of a RISC V Processor on FPGA". *2021 17th International Conference on Mobility, Sensing and Networking (MSN)*, Exeter, United Kingdom, 13-15 December 2021.

[13] Pilavendiran SA, Keshavarajan JL, Vijayakumar S. "Design and Development of JTAG Using Raspberry Pi". *International Journal of Engineering Applied Sciences and Technology*, 4, 75-79, 2020.

[14] Rico-García P, Yañez-Vargas JI, Parra-Michel R, Conde-Almada E. "Debugging and programming a RISC-V processor, using the IEEE 1149.1 standard". *Journal Applied Computing,* 7(21), 1-9, 2023

[15] Salehi M, Degani L, Roveri M, Hughes D, Crispo B. "Discovery and Identification of Memory Corruption Vulnerabilities on Bare-Metal Embedded Devices". *IEEE Transactions on Dependable and Secure Computing*, pp(99):1-1, 2022.

[16] OpenOCD. "Open On-Chip Debugger". https://openocd.org/doc/pdf/openocd.pdf (20.01.2024)

[17] Gao S, Wu D, Xiao W, Wang Z, Yang Z, Gao W. "A novel method for on-chip debugging based on RISC-V processor". *MATEC Web of Conferences*, Xiamen, China, 29-30 December, 2021.

[18] Toker O. "A High-Level Synthesis Approach for a RISC-V RV32I-Based System on Chip and Its FPGA Implementation". *10th International Electronic Conference on Sensors and Applications,* Online, 15-30 November, 2023.

[19] Heinz C, Lavan Y, Hofmann J, Koch A. "A Catalog and In-Hardware Evaluation of Open-Source Drop-In Compatible RISC-V Softcore Processors". *2019 International Conference on ReConFigurable Computing and FPGAs*, Cancun, Mexico, 9-11 December, 2019.

[20] Elsadek I, Tawfik EY. "RISC-V Resource-Constrained Cores: A Survey and Energy Comparison". *19th IEEE International New Circuits and Systems Conference*, Toulon, France, 13-16 June, 2021.

[21] EEMBC (Embedded Microprocessor Benchmark Consortium). CoreMark Benchmark. Erişim adresi: www.eembc.org (Erişim tarihi: 15.01.2025).

[22] Kreier, R. CoreMark Benchmarking Overview. Erişim adresi: https://kreier.github.io/benchmark/CoreMark/ (Erişim tarihi: 15.03.2025).

[23] R. Weicker, "Dhrystone: A Synthetic Benchmark," *Communications of the ACM*, vol. 27, no. 10, pp. 1013-1030, Oct. 1984.

[24] Mo, W., Liu, J., Wang, Y., Yan, F., Xiong, B., & Guan, J. An SDPF RISC-V Processor With 55.9% Dhrystone Improvement Using Two-Stage Pseudo-Pipelined Architecture for IoT Applications. International Journal of Circuit Theory and Applications 2025. https://doi.org/10.1002/cta.4514

[25] S. Bora and R. Paily, "A High-Performance Core Micro-Architecture Based on RISC-V ISA for Low Power Applications," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 68, no. 6, pp. 2132-2136, June 2021, doi: 10.1109/TCSII.2020.3043204.

[26] ORCA: A RISC-V Processor for FPGA Evaluation. GitHub Repository. Erişim adresi: https://github.com/riscveval/orca-1 (Erişim tarihi: 15.03.2025).

[27] PicoRV32: A Size-Optimized RISC-V CPU Core. GitHub Repository. Erişim adresi: https://github.com/YosysHQ/picorv32 (Erişim tarihi: 15.03.2025).

[28] mmRISC-1: A Minimal RISC-V CPU Implementation. GitHub Repository. Erişim adresi: https://github.com/munetomo-maruyama/mmRISC-1 (Erişim tarihi: 01.01.2024).