TEKNOFEST
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ
ÇİP TASARIM YARIŞMASI
DETAY TASARIM RAPORU

TAKIM ADI

MANTA

PROJE ADI

MANTA Z10 XT

BAŞVURU ID

469697

10.07.2022

İÇİNDEKİLER

KISALTMALAR LİSTESİ ii
1. TEMEL TASARIM ÖZETİ 1
2. PROJE MEVCUT DURUM DEĞERLENDİRMESİ 1
3. PROJE DETAY TASARIMI 2
3.1. Sistem Mimarisi 2
3.2. Tasarım Detayı 3
3.2.1. Çekirdek Tasarımı 3
3.2.2. Bellek Tasarımları 10
3.2.3. Çevre Birimleri Tasarımları 10
4. ÇİP TASARIM AKIŞI 11
5. TEST 12
6. TAKIM ORGANİZASYONU 13
6.1. Takım Organizasyonu 13
6.2. Görev Dağılımı 13
7. İŞ PLANI ve RİSK PLANLAMASI 13
KAYNAKÇA 15

TEKNOFEST
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ

KISALTMALAR LİSTESİ

ALU	Arithmetic Logic Unit
ASIC	Application Specific Integrated Circuit
CISC	Complex Instruction Set Computer
DTR	Detay Tasarım Raporu
FPGA	Field Programmable Gate Array
PDK	Process Design Kit
RISC	Reduced Instruction Set Computer
SPI	Serial Peripheral Interface
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver-Transmitter
ÖTR	Ön Tasarım Raporu

TEKNOFEST
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ

1. TEMEL TASARIM ÖZETİ

Bu yıl ilk defa düzenlenen TEKNOFEST çip tasarım yarışması sayısal kategorisi kapsamında temel olarak, özelleştirilmiş 64-bit RISC-V buyruk kümesini destekleyen, iki seviye önbellekli bir işlemcinin merkezinde yer aldığı *system on chip* tasarımı ve açık kaynaklı SKY130 PDK için yonga serimi yapılması beklenmektedir. Isparta Uygulamalı Bilimler Üniversitesi bünyesinde 2022 yılında kurulan Manta takımı, çip tasarım yarışmasına Manta Z10 XT işlemci projesi ile katılmıştır. Projeye, yüksek başarıma sahip ve düşük kaynak tüketen bir işlemci tasarlama hedefiyle yola çıkılmış olmakla birlikte, üniversitemiz özelinde sayısal tasarım, tümdevre üretimi gibi konularda farkındalık oluşturmak ve bu doğrultuda yapılacak çalışmalara önayak olmak projenin iç hedefleri arasındadır.

Yürütülen çalışmalar sonucunda projede gelinen noktada RV64I işlemci çekirdeği, verilog donanım tanımlama dili ile modellenmiştir. Modellenen çekirdek, FPGA ve ASIC için benzetim ve sentez işlemlerine tabi tutulmuştur. RV64M eklentisinin işlemciye getireceği çarpma, bölme buyrukları ve önbellek yapıları internet üzerinde yer alan açık kaynaklı projelerden faydanılarak geliştirilmektedir. RVC, RVX buyrukları ve UART, SPI çevrebirimleri henüz tasarıma eklenmemiştir. İşlemci tasarımının detayları Çekirdek Tasarımı kısmında, çalışma takvimi ile ilgili detaylar İŞ PLANI ve RİSK PLANLAMASI kısmında verilmiştir.

2. PROJE MEVCUT DURUM DEĞERLENDİRMESİ

Gerçekleştirilen ön çalışma neticesinde belirlenen proje nitelikleri ÖTR'de sunulmuştur. ÖTR'de paylaşılan işlemci mikromimarisinde, başarım hedeflerinde ve tasarım felsefesinde önemli bir değişiklik yapılmamıştır. Bununla beraber ÖTR'de kesin olarak belirtilmeyen bazı detaylar DTR'de netleştirilmiştir.

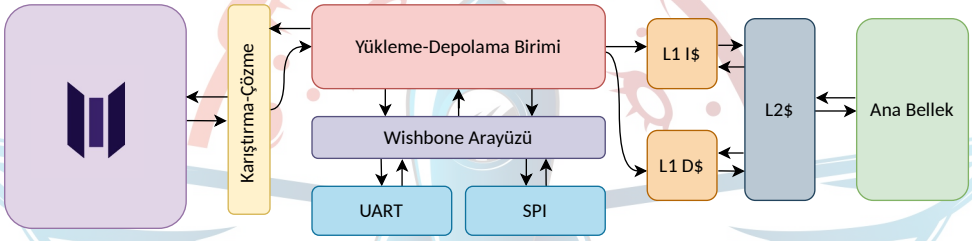
ÖTR'de aritmetik mantık biriminde bulunan toplayıcının *Parallel Prefix* türünde olan *Brent-Kung* veya *Kogge-Stone* yapılarından birinin olması düşünüldüğü belirtilmiştir. Daha sonrasında yapılan çalışmalar sonucunda toplayıcı devresi için yine *Parallel Prefix* türündeki *Sklansky* toplayıcı yapısının kullanılmasına karar verilmiştir. Söz konusu değişikliğin detayları Çekirdek Tasarımı kısımında incelenmiştir. ÖTR'ye göre değiştirilen diğer bir kısım ise çözme-karıştırma biriminin mimarideki konumudur. ÖTR'de bu birim ana bellek ve ikinci seviye önbellek arasında konumlandırılmıştır. Yarışmanın soru cevap etkinliğinden ve e-posta grubundan edinilen bilgiler doğrultusunda bu birimin işlemci ve birinci seviye önbellek arasında yer alması gerektiği anlaşılmıştır ve tasarım bu yönde revize edilmiştir.

3. PROJE DETAY TASARIMI

Bu kısmın alt başlıkları altında; projenin tasarımında kullanılan algoritmalara, devre yapılarına ve projenin gerçeklenmesinde kullanılan yöntemlere değinilmiştir.

3.1. Sistem Mimarisi

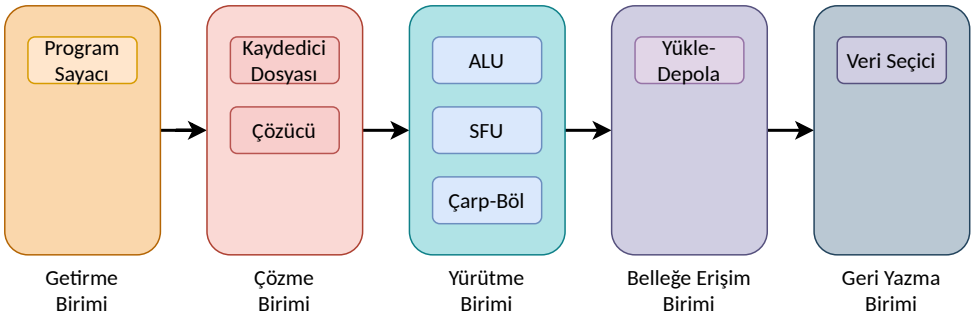
Yarışma kapsamında; 64-bit RISC-V (RV64IMCX) mimarili, 5 aşama boru hattına sahip bir işlemcinin, UART-SPI çevrebirimlerinin ve yarışma şartnamesinde yer alan bazı ek donanımların yer aldığı bir *System on Chip* tasarımı ve SKY130 üretim süreci için yonga serim akışının gerçeklenmesi planlanmıştır. Yarışma kapsamında tasarlanan işlemci projesine Manta Z10 XT ismi verilmiştir. Manta Z10 XT genel blok şeması 3.1'de verilmiştir.



Şekil 3.1. Manta Z10 XT Blok Şeması.

Manta Z10 XT işlemcisi çoğunlukla (Harris & Harris, 2021) kitabındaki 5 aşama boru hatlı işlemci örneği referans alınarak, sıfırdan Verilog donanım tanımlama dili ile modellenmiştir. Mikromimarideki veriyolu elemanları tasarlanırken, internette yer alan Ariane, Ibex, biRISC-V, NEORV32 gibi popüler açık kaynaklı RISC-V işlemci projeleri incelenmiştir. İşlemci veriyolu kabaca 5 parçaya bölünmüş ve her boru hattı aşaması ayrı birer modül olarak tasarlanmıştır. Boru hattı aşamalarını birbirine bağlayan kaydediciler, girişlerinin bağlı olduğu aşamaya dahil edilmiştir. Örneğin getir (fetch) aşaması ile çözme (decode) aşaması arasındaki boru hattı kaydedicileri getir aşaması modülü içinde yer almaktadır. Tasarım ve hata ayıklama süreçlerini kolaylaştırmak için modül sayısı ve üst modül içindeki bağlantı sayısı az tutulmaya çalışılmıştır. Bir boru hattı aşaması ile ilgili tüm birimlerin aynı modül içerisinde tanımlanması, genel hiyerarşiyi basitleştirmekte ve kodun anlaşılabilirliğini artırmaktadır. Veriyolunda bulunan modüller 3.2'de gösterilmiştir.

Manta Z10 XT işlemcisinin tasarımında düşük kaynak tüketimi, düşük güç tüketimi ve yüksek başarım hedefleri benimsenmiştir. Proje hedeflerine ulaşmak için donanımı basitleştirmek gibi literatürde kabul gören bazı prensipler izlenmiştir. Donanımı basitleştirmek sentez sonucunda üretilecek devrelerde karmaşıklığı azaltmakta, maksimum frekansı ve dolayısıyla başarımı artırırken güç tüketimini azaltmaktadır (D. A. Patterson & Hennessy, 2017). Mikromimari tasarımında donanımı basit tutmak yonga boyutunu da küçültmektedir. Donanımı



Şekil 3.2. Manta Z10 XT Veriyolu Modül Şeması.

mümkün olduğunca basit tutma ilkesine RISC-V resmi dökümantasyonlarında ve buyruk kümesi tasarımcılarının kitaplarında da sıklıkla vurgu yapıldığı görülmektedir (D. Patterson & Waterman, 2017). Bu doğrultuda, işlemci çekirdeğinde yer alan birimler mümkün olduğunca basit olarak tasarlanmaya çalışılmıştır. Bununla birlikte işlemci çekirdeğinde daha fazla optimize edilebilir ve sadeleştirilebilir kısımların mevcut olduğu görülmüştür.

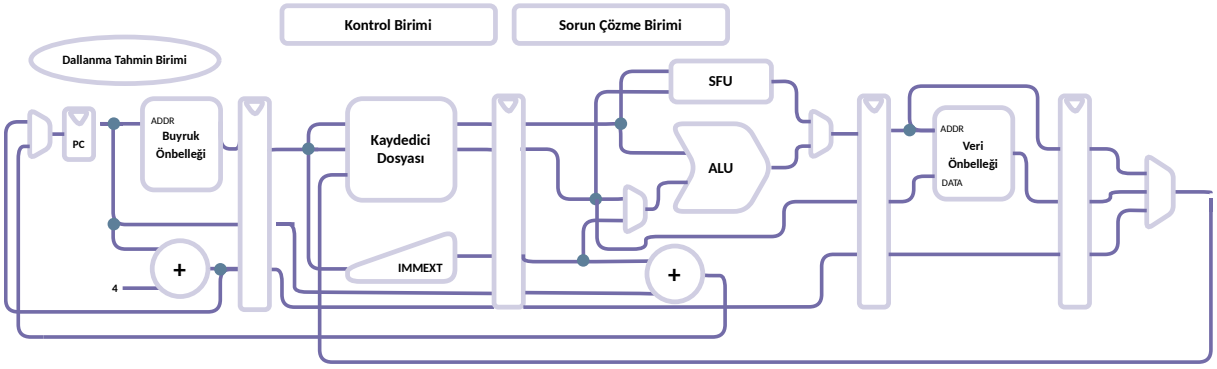
3.2. Tasarım Detayı

3.2.1. Çekirdek Tasarımı

İşlemci çekirdek mikromimarisinin yapısı, başarımı doğrudan etkileyen önemli unsurlardan biridir. İşlemci çekirdek mimarileri tek vuruşlu (single cycle), çok vuruşlu (multi cycle), boru hatlı veya süperskaler yapıda olabilir. Gerçeklenecek işlemcinin yapısı kullanım alanı, başarım ve güç tüketimi gibi kriterler göz önüne alınarak belirlenmelidir. Bir buyruk kümesi işlemci başarımının tek etkeni olmamakla birlikte RISC-V buyruk kümesi, CISC ve bazı eski RISC mimarilerinin aksine boru hattı tasarımını kolaylaştırmaktadır (D. Patterson & Waterman, 2017). RISC-V buyruk kümesi küçük bir mikrodenetleyiciden oda boyutlarında süper bilgisayarlara kadar geniş bir alanda ölçeklendirilebilmektedir (Waterman, 2016). Öte yandan basit bir mikrodenetleyici için birkaç boru hattı aşaması barındıran tek çekirdek bir mimari yeterli olabilirken, bir sunucu için sırasız yürütüm yapan onlarca çekirdeğe ihtiyaç duyulabilmektedir.

Yapısal olarak tek vuruşlu ve çok vuruşlu işlemciler, boru hattı barındıran mimarilere kıyasla daha kolay tasarlanabilmektedir. Fakat tek vuruşlu işlemciler toplam devre gecikmesini artırarak saat vuruş sıklığını azaltmakta ve sonuç olarak başarımı düşürmektedir. Çok vuruşlu işlemcilerde ise buyruklar birden fazla saat çevriminde tamamlanmaktadır. Çok vuruşlu işleme yönteminde işlemci çekirdeğinin içinde bulunan birimler verimli ve efektif olarak kullanılamamaktadır. Boru hattı içeren işlemciler ideal durumda, her saat çevriminde bir buyruk işlemektedir. İşlemci başarımını artıran bir etken olduğu için güncel işlemci mimarilerinde boru hattına sıklıkla rastlanmaktadır. Süperskaler mimarilerde boru hattı yöntemi uygulanmakta fakat ideal durumda, bir saat çevriminde birden fazla buyruk işlenmektedir. Süperskaler

işlemciler genellikle yüksek başarım gerektiren uygulamalarda tercih edilmektedir.



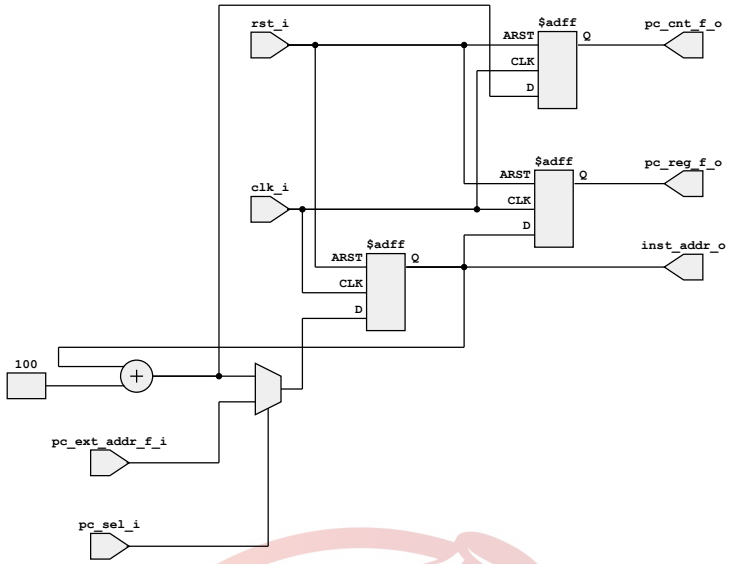
Şekil 3.3. Manta Z10 XT Çekirdek Şeması.

İşlemci mimari tercihinin olumlu tarafları olduğu gibi dezavantajlı durumları da olabilmektedir. Örneğin başarımı artırmak için eklenen ek donanımlar işlemci mimarisinde karmaşıklığı artırabilir. Mimari tasarım bu sebeple başarım beklentileri, alan ve güç isterlerine göre ödünleşme gerektirmektedir. Bu durumlar yarışma şartnamesinde belirtilen gereksinimlere göre değerlendirilmiştir. Yarışma isterlerinde yer verilen sıkıştırılmış buyruklar, donanımsal çarpma bölme desteği gibi özelliklere sahip olan ve kayan noktalı sayılar için donanım hızlandırma desteği bulundurmayan benzer nitelikte mimariler incelenmiş ve sonuç olarak 5 aşama boru hattına ve dallanma tahmin ünitesine sahip bir işlemci tasarlanması uygun bulunmuştur. İşlemci çekirdek mimari şeması Şekil 3.3'de basitleştirilmiş biçimde gösterilmiştir.

Getirme birimi (fetch unit) işlemci boru hattının ilk aşamasıdır. Getirme birimi, program sayacının geçerli değerini buyruk önbelleğine adres olarak iletir. Buyruk önbelleği, getirme biriminin işaret ettiği adresteki 32-bit genişlikteki buyruğu, çözme birimine (decode unit) aktarır. Boru hattı sorunu gibi beklenmeyen bir durum meydana geldiğinde bu birim durdurulabilmekte veya aşama kaydedicileri boşaltılabilmektedir. Tasarlanan getirme biriminin yapısı Netlistsvg yazılımı ile elde edilen RTL şema üzerinden Şekil 3.4'de gösterilmiştir. Getirme biriminin gösterilen sürümünde RV64C eklentisi gerçeklenmemiştir. Bu sebeple nihai tasarımdaki getirme birimi ile farklılıklar gösterecektir.

Çözme birimi, buyruğun ilgili kısımlarını kontrol birimine (control unit) aktarır. Buyruk ivedi (immediate) türde ise anlık değerin işaret biti genişletilir ve yürütme birimine (execute unit) aktarılır. 64-bit genişlikte 32 adet kaydedici barındıran kaydedici dosyası da çözme birimi içinde yer almaktadır.

Yürütme birimi içerisinde aritmetik mantık birimi (arithmetic logic unit-ALU), özel işlem birimi (special function unit-SFU), çarpma-bölme birimi (mul/div unit) ve dallanma hesap birimi gibi devreler bulunmaktadır. İşlemci çekirdeğinin tüm hesaplama işlemleri yürütme biriminde gerçekleşmektedir. Yarışma şartnamesinde bulunan özel kriptografik hızlandırıcı buyruklar özel

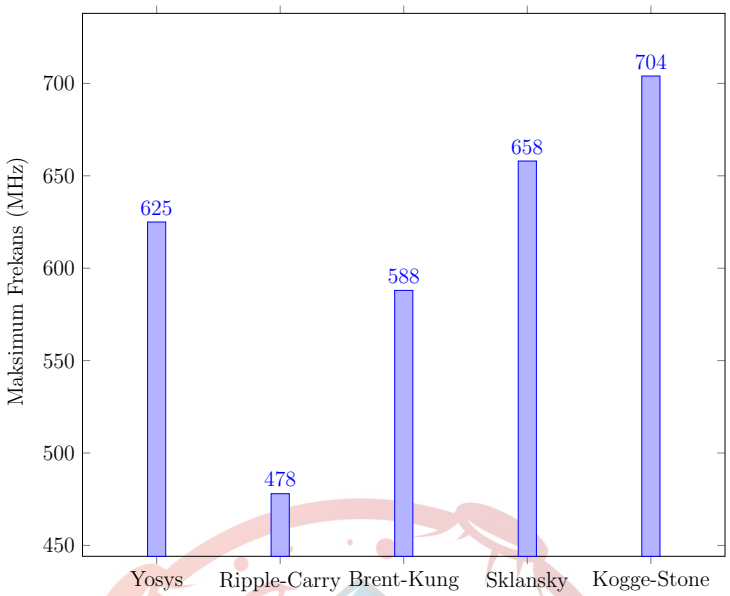


Şekil 3.4. Getirme Biriminin RTL Şeması.

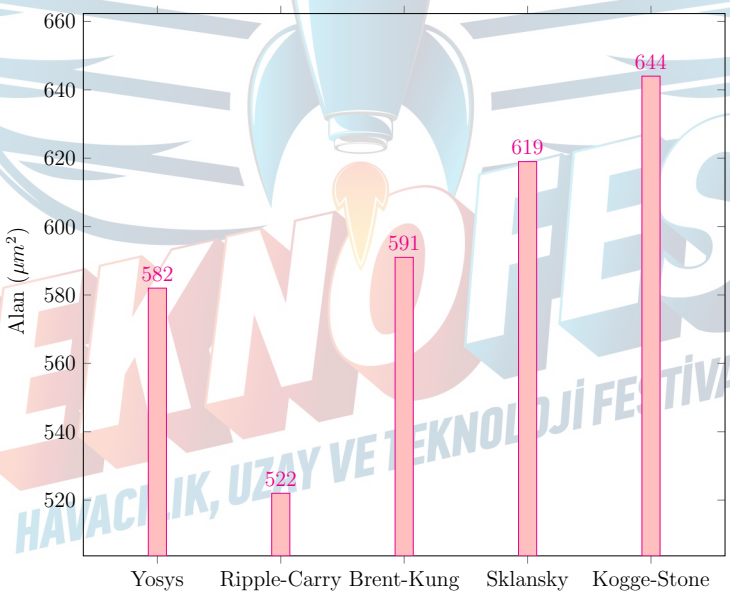
işlem birimi altında gerçeklenecektir. RV64M eklentisi kapsamındaki buyruklar çarpma bölme biriminde yer alacaktır.

RV64I buyruk kümesinde bulunan toplama buyrukları için *Ripple-Carry* ve *Carry-Lookahead* toplayıcılar yerine çok daha hızlı (Harris & Harris, 2021) *Parallel Prefix* toplayıcıların kullanılmasına karar verilmiştir. *Parallel Prefix* formunda birçok toplayıcı devresi literatürde yer almaktadır. (Ene, 2022) açık kaynaklı olarak yayınladığı projesinde, *Prefix* toplayıcı devreleri üretmeyi sağlayan *pptrees* Python kütüphanesini geliştirmiştir. Bu sayede çeşitli türde ve genişlikte toplayıcı yapıları kolaylıkla sentezlenebilmektedir. (Ene & Venn, 2022) başka bir çalışmada *pptrees* kütüphanesi ile oluşturduğu çeşitli toplayıcı devrelerini maksimum frekans ve alan tüketimi gibi hususlarda karşılaştırmıştır. Testleri Skywater PDK kullanarak OpenLane yonga serim akışı üzerinde gerçekleştirmiştir. Manta Z10 XT işlemci mimarisinde tercih edilecek toplayıcı devresi için *pptrees* açık kaynaklı projesinden faydalanılmıştır. *pptrees* ile üretilebilen bazı toplayıcı devrelerinin OpenMPW programı için hesaplanan maksimum frekans ve kaplanan alan bakımından kıyaslamaları Şekil 3.5'de ve Şekil 3.6'da gösterilmiştir.

(Ene & Venn, 2022) çalışması incelendiğinde, en az alan kaplayan toplayıcı devresinin *Ripple-Carry* olduğu, en yüksek frekansta çalışabilen toplayıcı devresinin *Kogge-Stone* olduğu görülmektedir. Manta Z10 XT işlemci mimarisi için Maksimum frekans/kaplanan alan açısından *Sklansky* toplayıcı devresi uygun bulunmuştur. *pptrees* kütüphanesi ile 64-bit genişlikte *Sklansky* toplayıcı devre üretimi için Şekil 3.7'de gösterilen Python kodu çalıştırılmıştır. Bu program toplayıcı devresini sentezlenebilir verilog koduna dönüştürmektedir. Üretilen toplayıcı devresi Şekil 3.8'de verilmiştir.



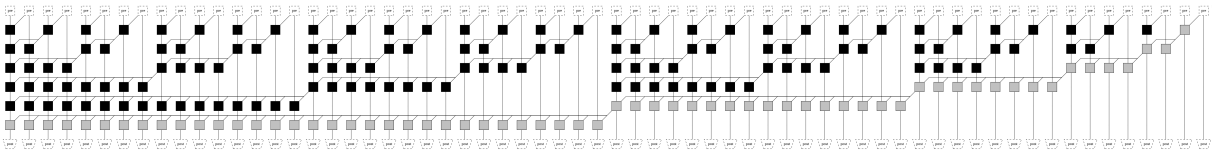
Şekil 3.5. 8-bit Toplayıcı Devrelerinin Maksimum Frekans Bakımından Karşılaştırılması.



Şekil 3.6. 8-bit Toplayıcı Devrelerinin Kaplanan Alan Bakımından Karşılaştırılması.

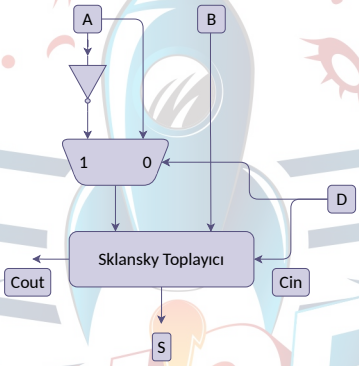
```
from pptrees.AdderForest import AdderForest as forest
width = 64
f = forest(width, alias = "sklansky")
f.hdl('adder.v')
```

Şekil 3.7. pptrees Kütüphanesi ile 64-bit Sklansky Toplayıcı Kodunun Üretimi.



Şekil 3.8. pptrees Kütüphanesi ile Oluşturulan 64-bit Sklansky Toplayıcının Şeması.

RV64I buyruk kümesinde yer alan *Sub* buyruğu için bir çıkarma devresi gerekmektedir. Çıkarma devresi, ikinin tümleyeni aritmetiğinde toplayıcı devresine yapılan küçük modifikasyonlarla gerçeklenebilmektedir. N-bitlik girişlere sahip toplama işlemi $A + B = S$ olarak gösterilirse, çıkarma işlemi $\overline{A} + B + 1 = S$ olmaktadır. Alan tüketimini azaltmak amacıyla, ALU'daki çıkarma işlemi "-" işareti kullanıp devreyi sentezleyiciye bırakmak yerine *Sklansky* toplayıcı iki işlemi de destekleyecek biçimde değiştirilmiştir. Toplama ve çıkarma işlemleri için kullanılan devrenin örnek şeması Şekil 3.9'da gösterilmiştir.



Şekil 3.9. Sklansky Toplayıcı Çıkarıcı Devresi.

Şartnamede yer alan RV64X eklentisi altında değerlendirilen özel buyruklar özel işlem biriminde yürütülecektir. RV64M eklentisinin gerektireceği çarpma ve bölme buyrukları çarpma-bölme biriminde yürütülecektir. Çarpma işlemi *Shift-Add* yöntemi ile bölme işlemi *Shift-Sub* yöntemi ile yapılacaktır. Bu yöntemlerde hesaplama işlemi, işlenen sayıların bit genişliliği ile bağlantılı olarak çok çevrimde tamamlanmaktadır. Örneğin basit bir çok vuruşlu çarpıcı devresinde 64-bit iki sayının çarpımı 64 saat çevrimi harcamaktadır. İşlemci başarımını artırmak amacıyla çarma-bölme biriminde tek çevrimde iki yineleme yapılacaktır. Bu sayede 64-bit sabit genişlikte sayılar ile çarpım ve bölüm işlemleri 32 saat çevriminde tamamlanmış olacaktır. Çözme aşamasına RV64M eklentisinden bir buyruk gelmesi halinde boru hattı ilgili işlemler tamamlanana değin durdurulacaktır.

Bellek erişim birimi (memory access unit), yürütme biriminden gelen adres ve veri gibi bilgileri yükleme-depolama birimine (load-store unit) erişim isteği ile birlikte iletir. Bellek erişim birimi ayrıca yürütme biriminden gelen sonuç verisini bir sonraki aşamanın birimi olan geri yazma birimine (writeback unit) iletir.

Geri yazma birimi, bellek erişim biriminden gelen verilerin kaç bayt içerdiğine, işaretli-işaretsiz olma durumuna bakar ve gerekli durumlarda işaret genişletme işlemi gerçekleştirip, sonucu kaydedici dosyasına yazar. Geri yazma aşaması tasarlanan işlemcinin son boru hattı aşamasıdır.

Kontrol birimi (control unit) diğer tüm çekirdek içi birimlerin doğru şekilde çalışmasından sorumludur. Bu birim kontrol sinyali adı verilen bağlantılarla diğer birimlere bağlıdır. Buyrukların işlenmesi kontrol sinyallerinin doğru bilgileri aktarması ile mümkün olmaktadır.

İşlemci çekirdeğinde yer alan son birim sorun çözme birimidir (hazard unit). Bu birim işlemcinin çalışması esnasında meydana gelebilecek çeşitli boru hattı sorunlarını çözmektedir. Bunlardan ilki veri sorunudur, arkadan gelen buyruk bir önceki buyruğun tamamlanmasına ihtiyaç duyuyorsa bu sorun meydana gelmektedir. Çözüm için boru hattı, ilk buyruk tamamlanana kadar boru hattı bekletilebilir. Fakat bu çözüm işlemci başarımını olumsuz etkileyecektir. Alternatif bir çözüm olarak belleğe erişim ve geri yazma aşamalarından gerekli veri getirilebilir. Bu yönteme veri ilerletme (data forwarding) adı verilmektedir. Tasarlanan işlemcide veri sorununu çözmek için ilerletme yöntemi kullanılmıştır. Boru hattında veri sorununa sebep olan örnek bir assembly kodu Şekil 3.10'da gösterilmiştir.

```
add  x6 ,  x4 ,  x5
xor  x2 ,  x6 ,  x3
and  x9 ,  x7 ,  x6
addi x12 , x6 ,  28
```

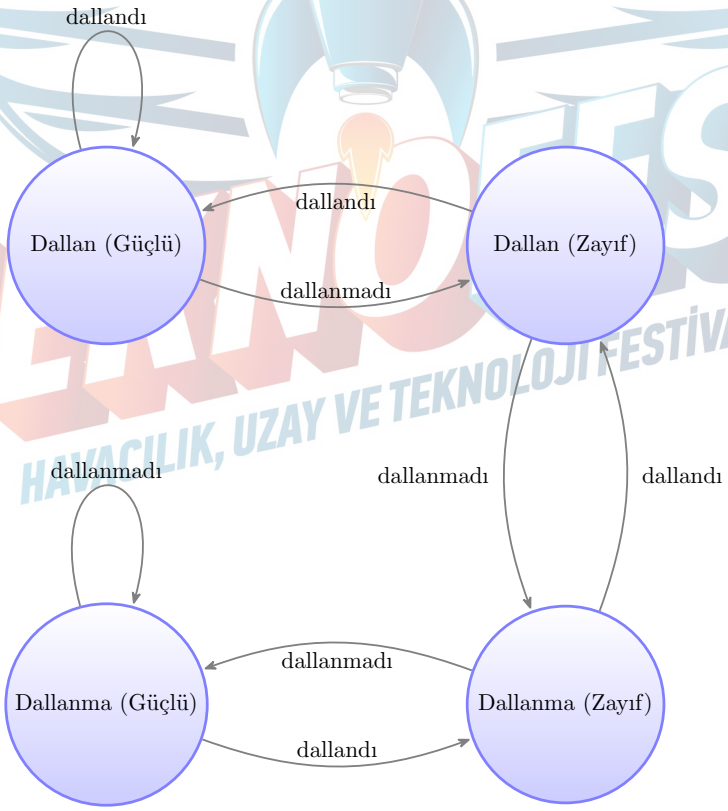
Şekil 3.10. x6 Kaydedicisinin Sebep Olduğu Veri Sorunu.

Boru hattında meydana gelebilecek bir diğer sorun *Load* buyruğu sorunudur. Bu sorun bir *Load* buyruğunun ardından Load buyruğunun sonucuna bağımlı bir buyruk gelmesiyle oluşmaktadır. *Load* buyruğu sorununda bellekten henüz okunmayan verinin yürütme birimine aktarılması gerekmektedir. Bu sebeple *Load* buyruğu sorunu meydana geldiğinde boru hattı bir saat çevrimi durdurulmaktadır. Boru hattında *Load* buyruğu sorununa sebep olan örnek bir assembly kodu Şekil 3.11'de gösterilmiştir.

```
lw   x5 ,   16(x0)
and  x2 ,   x5 ,  x3
xor  x9 ,   x7 ,  x5
sub  x12 ,  x5 ,  x20
```

Şekil 3.11. x5 Kaydedicisinin Sebep Olduğu *Load* Buyruğu Sorunu.

Boru hattında soruna sebep olan son olay, dallanma buyruklarının işlenme durumudur. Bir dallanma buyruğu boru hattına girdiği anda bir sonraki çevrimde işlenecek buyruğun adresi belirsiz durumdadır. Bu noktada dallanmanın gerçekleşip gerçekleşmeyeceği tahmin edilmelidir. Literatürde birçok tahmin yöntemi bulunmakla beraber en temel çözümlerden biri sabit tahminde bulunmaktır. Bu yöntemde dallanma buyruğu tespit edildiğinde işlemci hep dallanma ya da hep dallanmama yönünde bir tercihte bulunur. Dallanma tahmininin doğru olup olmadığı ilgili dallanma buyruğu yürütme aşamasına geçtiğinde anlaşılır. Yanlış bir tahminde bulunulmuşsa hatalı yürütülmüş olan işlemlerin geri alınması gerekmektedir. Bu sebeple yanlış tahmin durumu işlemcinin başarımını düşüren bir etkendir. Tasarladığımız işlemci mikromimarisinde hatalı işlemlerin düzeltilmesi, boru hattının boşaltılması anlamına gelmektedir. Manta Z10 XT işlemcisinin güncel geliştirme sürümünde, dallanma buyrukları geldiğinde dallanmama tercihi yapılmaktadır. İşlemci başarımını artırmak amacıyla 2-bit dinamik dallanma tahmin birimi eklenmesi planlanmıştır. Bu birim 4 tahmin durumu barındıran bir sonlu durum makinesinden oluşmaktadır. Dallanma buyruğu çözüldüğünde, dallanma tahmin biriminin durumuna göre dallanma yapılıp yapılmayacağına karar verilmektedir. 2-bit dallanma tahim biriminin durum makinesi Şekil 3.12'de gösterilmiştir.



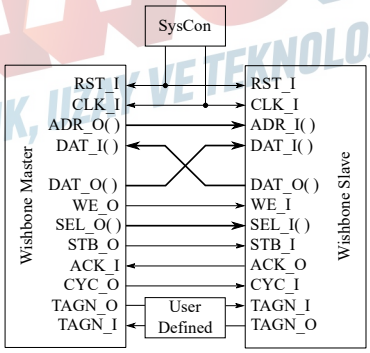
Şekil 3.12. 2-bit Dinamik Dallanma Tahmin Biriminin Durum Makinesi.

3.2.2. Bellek Tasarımları

Belleklerin genel işlemci başarımı üzerinde kritik rolü bulunmaktadır. Belleklerin kapasitelerindeki artış beraberinde yavaşlama getirmektedir. Bu sebeple büyük bellek yapıları yüksek işlemci hızlarına yetişememektedir (D. A. Patterson & Hennessy, 2017). İşlemcinin verilere hızlı erişimini sağlamak için önbellek adı verilen, ana belleğin küçük alt kümeleri kullanılmaktadır. Bir mimaride birden fazla önbellek yer alabilir. Bu durumda en küçük kapasiteli bellek işlemciye en yakın konumda olacak şekilde hiyerarşik yapı meydana getirilir. Önbellek seviyesi yükseldikçe belleğe erişim gecikmesi artar, dolayısıyla işlemci başarımı düşer. İki seviye önbellek içeren bir mimaride, işlemci ilk önce birinci seviye önbelleğe başvurur, aranan veri birinci seviye önbellekte yer almıyorsa bu defa bir sonraki seviye önbelleğe başvurur. Aranan veri önbelleklerde yer almıyorsa, işlemci ana belleğe başvurur. Aranan verinin en düşük seviyeli önbellekte olması ideal durumdur ve pratikte her zaman mümkün olamamaktadır.

Manta Z10 XT işlemcisi için iki seviye önbellek kullanılması planlanmıştır. 2 KB boyutundaki birinci seviye önbellekler doğrudan eşlemeli olarak tasarlanacaktır. Doğrudan eşlemeli önbellekler en basit önbellek yapısıdır. Bu yöntemde veriler, adreslerine göre belirlenen önbellek alanlarına yerleştirilir. 8 KB boyutundaki ikinci seviye önbellek için, 4 yollu küme ilişkili önbellek tasarımı kullanılması planlanmıştır. Önbellek için gereken SRAM hücreleri OpenRAM ile derlenerek oluşturulacaktır. Önbellek tasarım hususunda çalışmalar devam etmektedir. İşlemci çekirdeği ile birinci seviye önbellekler arasına karıştırma-çözme birimi yerleştirilecektir. Bu birim, ana bellekte karıştırılmış olarak tutulan buyrukları ve verileri işlemci için anlamlı hale getirecektir.

3.2.3. Çevre Birimleri Tasarımları



Şekil 3.13. Wishbone Arayüzü.

Manta Z10 XT işlemcisi için bellek haritasına eşlemeli olarak UART ve SPI çevrebirimleri tasarlanacaktır. Çevrebirimleri işlemcinin yükleme-depolama birimine Wishbone arayüzü ile bağlanacaktır. Wishbone protokolü açık kaynaklı bir standarttır ve bu yönüyle projenin tasarım felsefesine uygundur.

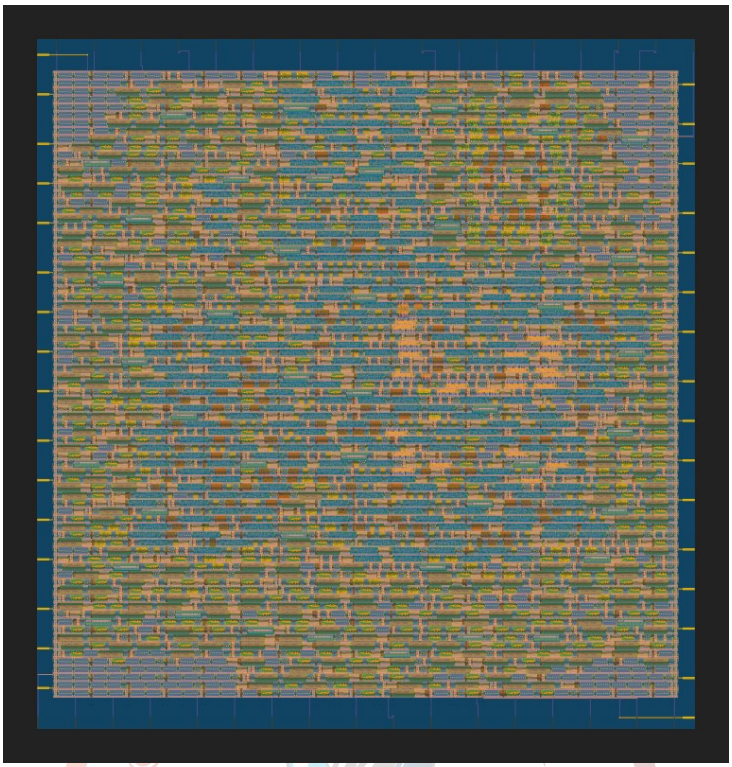
Çevrebirim arayüzü olarak Wishbone'un tercih edilmesinin bir diğer nedeni ise internet üzerinde Wishbone uyumlu birçok açık kaynaklı proje bulunmasıdır. Bu sayede SoC mimarisine farklı çevrebirimlerin entegrasyonu kolaylaşmış olacaktır. Örnek bir Wishbone arayüzü Şekil 3.13'de gösterilmiştir.

4. ÇİP TASARIM AKIŞI

Açık kaynaklı SKY130 PDK için yonga tasarım akışı OpenLane ile yapılmaktadır. OpenLane, Verilog kodundan GDSII dosyasına kadar olan ASIC akışını otomatik olarak yürüten açık kaynaklı bir yazılım paketidir. Akış çeşitli aşamalardan oluşmaktadır ve her aşamada farklı açık kaynak yazılımlar bulunmaktadır. Sentez, ASIC akışının birinci aşamasıdır. Sentez aracı donanım tanımlama dili ile yazılan tasarımı girdi olarak alır ve tasarımı PDK'de tanımlı olan standart hücrelere dönüştürür. OpenLane akışında sentez işlemi Yosys ile yapılmaktadır. Bir sonraki aşama *Static Timing Analysis*'dir. Bu aşamada saat sinyalinin flip-flop gibi ardışık (sequential) elemanlara doğru şekilde ulaşıp ulaşmadığının kontrolü yapılır. OpenLane akışında *Static Timing Analysis* işlemi OpenSTA ile yapılmaktadır. Daha sonra *Floor Planning* denilen hiyerarşik yerleşim planı hazırlanır. Bu aşamanın ardından *Place and Route* işlemleri yapılır. *Placement* aşamasında standart hücreler yerleştirilir. *Route* aşamasında ise standart hücrelerin bağlantıları yapılır. Her iki aşama önce kabaca işlemden geçer ve daha sonra detaylı olarak tekrar işlenir. Saat sinyalleri özel olarak *Clock Tree Synthesis* aşamasında işlenir. Son olarak fiziksel doğrulama testleri gerçekleştirilir. Bu testler *DRC*, *LVS* ve *Anten* kontrolüdür. Akışın hatasız biçimde tamamlanması halinde GDSII dosyası üretilmektedir.

Manta Z10 XT işlemci çekirdeğinin (RV64I) OpenLane akışı ile yonga serim işlemi gerçekleştirilmiştir. Akışta, kararlı çalıştığı tecrübe edilen "2022.02.10-02.15.04" sürümü kullanılmıştır. Daha güncel sürümlerin ilerleyen süreçte denenmesi planlanmıştır.

Yonga seriminin ilk birkaç denemesinde hata alınmıştır. Hata incelendiğinde *Placement* aşamasında bulunan *Resizer* programından kaynaklandığı anlaşılmıştır. Bu program tasarımda opsiyonel olarak optimizasyonlar yapmaktadır. *config.tcl* dosyası üzerinden *Resizer* devre dışı bırakılmış ardından başarılı şekilde akış tamamlanmıştır. Akış raporunda DRC, LVS ve Anten kontrolü temiz olarak sonuçlanmıştır. OpenLane varsayılan saat frekansına göre *setup hold* ihlalleri ortaya çıkmamıştır. İşlemci çekirdeği *sky130_fd_sc_hd* standart hücrelerinden 2101 adet kullanmıştır. Manta Z10 XT işlemci çekirdeğinin *gdstk* Python kütüphanesinden elde edilen GDS görüntüsü Şekil 4.1'de verilmiştir.



Şekil 4.1. Manta Z10 XT Çekirdeğinin Yonga Serimi.

5. TEST

Manta Z10 XT işlemcisinin çekirdek tasarımındaki serim öncesi benzetim işlemlerinde Vivado 2018.2 yazılımı kullanılmıştır. Veriyolunda tasarlanacak elemanlar ve işlemcinin desteklemesi gereken buyruklar küçük parçalar halinde geliştirilmiştir. Her yeni sürümde, işlemci çekirdeğinin fonksiyonelliği yeniden test edilmiştir. Böylelikle oluşabilecek hataların en kısa sürede fark edilip düzeltilmesi sağlanmıştır. Gerçeklenen RV64I buyrukların test edilmesi için assembly programları yazılmıştır. Boru hattının doğrulanması için, sorunlara yol açan kodlar işlemci üzerinde test edilmiştir. Yazılan assembly programları ilk test mahiyetindedir, detaylı doğrulama işlemleri resmi RISC-V uyumluluk testleri (Newsome, 2022) ile yapılacaktır.

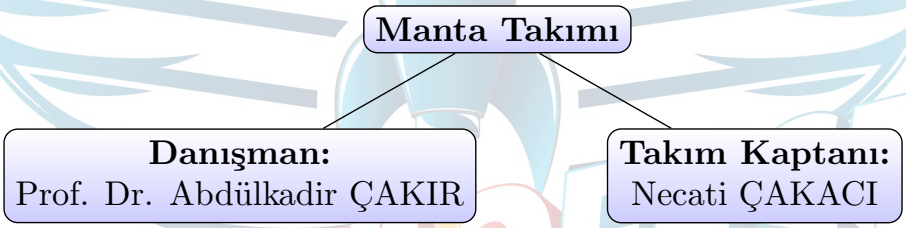
Tasarıma sonradan eklenecek önbellekler, karıştırma-çözme birimi ve çevrebirimleri gibi birimler için de fonksiyonel testler gerçekleştirilecektir. Benzetim aşamasının başarıyla tamamlanması halinde, tasarım bir FPGA üzerinde sentezlenecek ve yonga serim aşamasına geçilecektir. Serim sonrası gerçekleştirilecek doğrulama işlemlerinde açık kaynaklı Iverilog ve GTKWave yazılımları kullanılacaktır.

Başarımı ölçmek amacıyla, FPGA'da sentezlenen işlemci üzerinde CoreMark ve Dhrystone gibi yaygın olarak kullanılan *benchmark* yazılımları koşturulacaktır. Son aşamada, tasarlanan işlemci için çevrebirim kontrol fonksiyonları ve bazı yardımcı fonksiyonları barındıran bir C kütüphanesi (*mantality.h*) geliştirilecektir.

6. TAKIM ORGANİZASYONU

6.1. Takım Organizasyonu

Takımın akademik danışmanı Prof. Dr. Abdülkadir ÇAKIR, Isparta Uygulamalı Bilimler Üniversitesi Elektrik-Elektronik Mühendisliği bölümünde görev yapmaktadır. Proje yürütücüsü Necati ÇAKACI, Isparta Uygulamalı Bilimler Üniversitesi Elektrik-Elektronik Mühendisliği bölümünde yüksek lisans öğrencisidir. Takım organizasyonu Şekil 6.1'de gösterilmiştir.



Şekil 6.1. Takım Organizasyonu.

6.2. Görev Dağılımı

Yarışma kapsamında değerlendirilen tasarım, benzetim, sentez ve yonga serim akışı çalışmaları proje yürütücüsü tarafından gerçekleştirilmiştir.

7. İŞ PLANI ve RİSK PLANLAMASI

Ön tasarım raporu hazırlanırken, tüm çalışmanın detay tasarım raporu teslimine kadar tamamlanması gerektiği gibi anlaşılmış ve iş planı buna göre yapılmıştır. ÖTR'de paylaşılan iş planı Şekil 7.1'de gösterilmiştir. Yeşil renkte verilen iş paketleri tamamlanmış, sarı renkte verilen iş paketleri kısmen tamamlanmış, kırmızı renkte verilen iş paketleri tamamlanmamış olduğu anlamına gelmektedir. İşlemci çekirdek tasarımında yaşanan aksaklıklar nedeniyle proje takviminin gerisinde kalınmıştır. Projenin belirtilen süre içinde tamamlanamama riski olasıdır.

İş Paketi	Mart				Nisan				Mayıs			Haziran			Temmuz			
Ön çalışma ve ÖTR'nin hazırlanması	x	x	x	x														
RV64I mimarinin gerçeklenmesi			x	x	x	x	x											
Aritmetik algoritmaların gerçeklenmesi							x	x										
RV64IM mimarinin gerçeklenmesi								x	x									
Önbellek yapılarının tasarıma dahil edilmesi									x	x	x							
Karıştırma ve çözme mekanizmasının hazırlanması										x	x							
RV64IMCX mimarinin gerçeklenmesi											x	x	x					
Çevrebirimlerin tasarımı												x	x	x				
DTR'nin hazırlanması												x	x	x				
Hata düzeltme ve iyileştirme çalışmaları															x	x	x	x	x	x
Final sunumunun hazırlanması															x	x	x	x	x	x

Şekil 7.1. İş Planı.

Plan takvimine yetişip, projeyi tamamlamak için ders döneminin sona ermesiyle birlikte yoğun çalışma temposuna geçilmiştir.

TEKNOFEST
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ

KAYNAKÇA

Ene, T. D. (2022). *Synth opt adders*. Retrieved from https://github.com/tdene/synth_opt_adders ([Online; accessed 10-July-2022])

Ene, T. D., & Venn, M. (2022). *Instrumenting adders*. Retrieved from https://github.com/mattvenn/instrumented_adder ([Online; accessed 10-July-2022])

Harris, S. L., & Harris, D. (2021). Digital design and risc-v computer architecture textbook. In *2021 acm/ieee workshop on computer architecture education (wcae)* (p. 1-5). doi: 10.1109/WCAE53984.2021.9707615

Newsome, T. (2022). *Risc-v tests*. Retrieved from https://github.com/riscv-software-src/riscv-tests ([Online; accessed 10-July-2022])

Patterson, D., & Waterman, A. (2017). *The risc-v reader: An open architecture atlas* (1st ed.). Strawberry Canyon.

Patterson, D. A., & Hennessy, J. L. (2017). *Computer organization and design risc-v edition: The hardware software interface* (1st ed.). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Waterman, A. (2016). *Design of the risc-v instruction set architecture* (Doctoral dissertation, EECS Department, University of California, Berkeley). Retrieved from http://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-1.html

TEKNOFEST
HAVACILIK, UZAY VE TEKNOLOJİ FESTİVALİ