



Emre İŞSEVER
Gömülü Sistem Mühendisi

04.12.25

HDL Coder ile FPGA Tasarımında Model Tabanlı Yaklaşım



Agenda

01 FPGA Teknolojisi

FPGA Nedir? Avantajları Nedir? Nerede Kullanılır?

02 Temel Sayısal Tasarım Altyapısı

Lisans Düzeyi Temel Sayısal Tasarım Uygulamaları ve Vivado Kurulumu

03 FPGA Programlama Metodları

Uygulamalı geleneksel FPGA Programlama Metodları

04 Model Tabanlı Yaklaşım ile FPGA Programlama

Akademik hayatta MBD ile FPGA Programlamanın Yeri

05 Alana Özgü Uygulamalar

Soyutlanma Seviyesi Yüksek Tasarımlar ve Uygulanması

06 FPGA Doğrulama

FPGA Doğrulamada MATLAB Araçlarının Yeri

07 Q & A

FPGA Nedir?

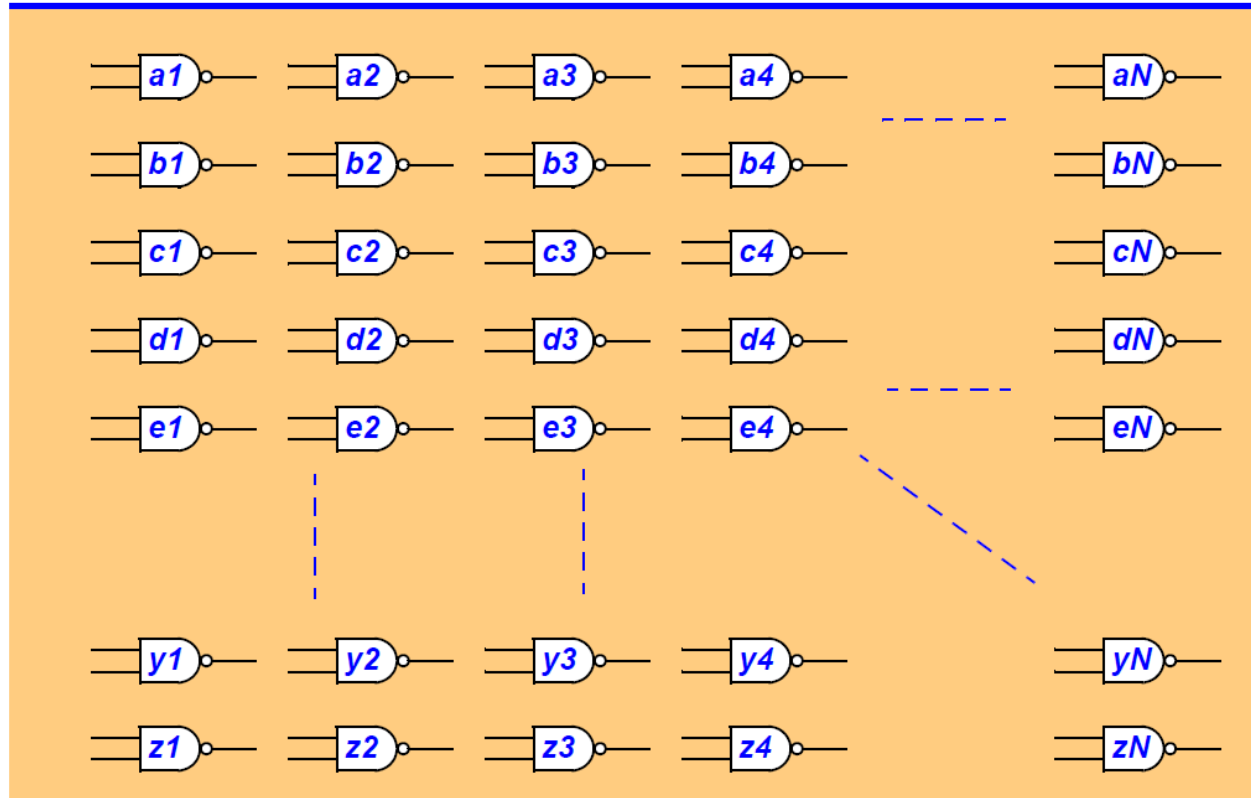
***F**ield **P**rogrammable **G**ate **A**rray (FPGA), davranışını özelleştirmek için programlayabileceğiniz dijital mantık devreleri içeren bir elektronik aygıttır.*

- **"Field Programmable"** – Donanım, fabrika dışında (=sahada 'Field') programlanabilir olduğu için sahada programlanabilir ismini alır.
- **"Gate Array"** – 2 Boyutlu Mantık kapı dizisi.



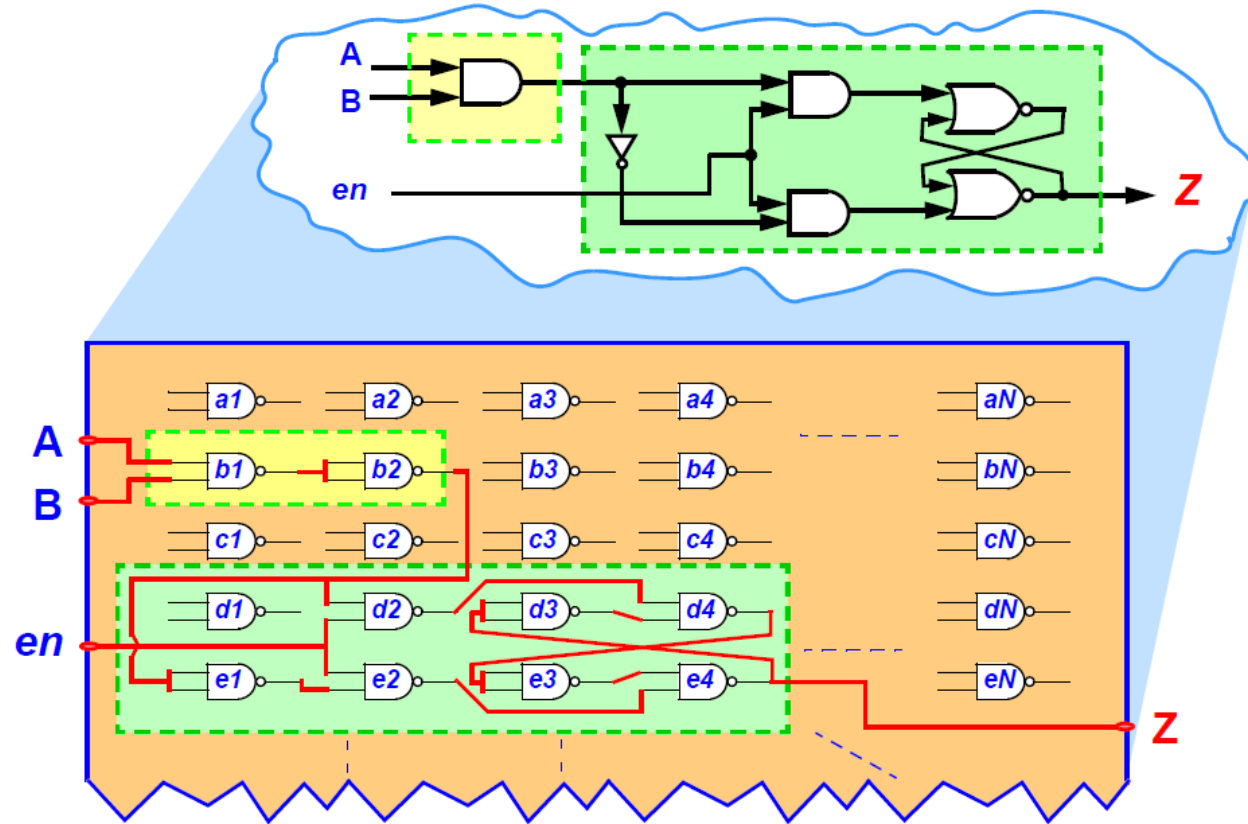
Gate Array – I

Önceleri kapı dizisi (*ing. Gate Arrays*) basitçe bir dizi NAND kapısına deniyordu.

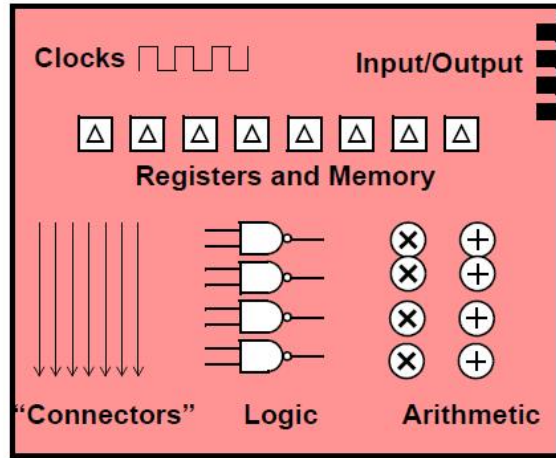


Gate Array – II

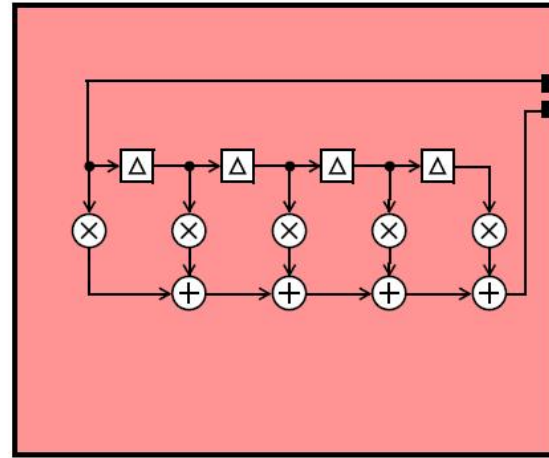
Örneğin burada **AND Kapısı** ve **D-Latch** yapısı NAND kapıları kullanılarak tasarlanmıştır.



FPGA : Bir Kutu Dijital Mantık Devresi



Design
Verify
Place and Route



Nerede Kullanılır?

- **Paralel Hesaplama**
 - Dijital Sinyal İşleme
 - Dijital Haberleşme (ör. SDR)
 - Yüksek hesaplama gücü gerektiren algoritmaların gerçekleştirilmesi.
- **Düşük Gecikme**
 - Motor Kontrol, Güç Dönüştürücüler , Servo Sistemler
- **Deterministik Zamanlama**
 - Radar , Elektronik Harp , Güvenlik Kritik Sistemler
- **Yeniden Programlanabilirlik**
 - Yeni algoritma denemeleri
 - Çip Tasarım Prototipleri
 - Yüksek Lisans/Doktora Projeleri



FPGA Programlama / Tasarlama

Hardware Description Language (HDL)



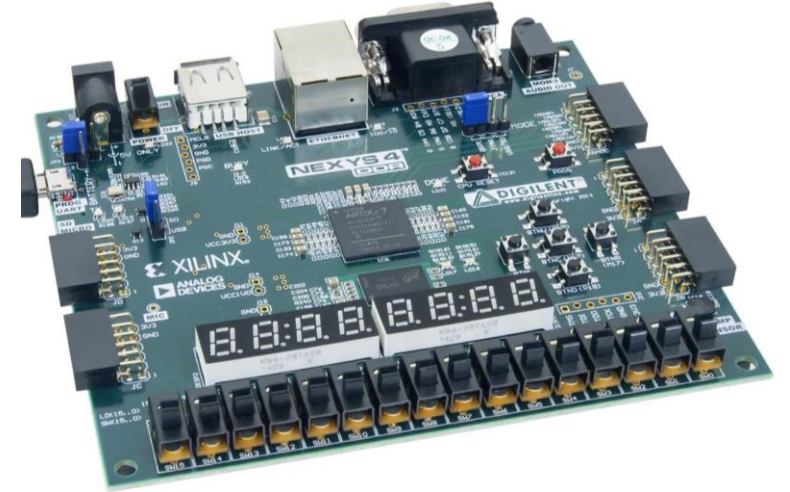
VHDL (.vhd)
Verilog (.v)
SystemVerilog (.sv)



AMD Vivado

Intel Quartus

Microchip Libero



AMD (Xilinx) Vivado Kurulumu

AMD
https://www.xilinx.com › support › download

Downloads

Vivado Software · Vitis Software · Vitis Model Composer · Vitis HLS · Vitis AI · Embedded Software · Power Design Manager. Intellectual Property & Apps. Pre- ...

[Xilinx Vivado Design Suite...](#) [Xilinx ISE 14.7](#) [Installer Information](#)

 [AMD Unified Installer for FPGAs & Adaptive SoCs 2024.2: Windows Self Extracting Web Installer \(EXE - 222.91 MB\)](#)

MD5 SUM Value : 84ee3816d8cbd0e0e38a1f9ac7eaa780

Download Verification 

Digests

Signature

Public Key

Version

2024.2

[2024.1](#)

[2023.2](#)

[Vivado Archive](#)

[ISE Archive](#)

[CAE Vendor Libraries
Archive](#)

Vivado™ Edition - 2024.2 Full Product Installation

Important Information

Vivado™ 2024.2 is now available for download:

Advanced Flow for Place-and-Route of All Versal™ Devices

- Automatic partition-based placement and parallel P&R
- Reduces congestion and improves routability for fast design closure
- Default flow for all Versal devices

Enabling Top-Level RTL Flows for Versal Devices

- Configure key components like NoC and transceivers from top-level RTL
- Enables programmable logic developers to stay in a RTL-centric design environment

Download Includes

Vivado Design Suite (All Editions)

Download Type

Full Product Installation

Last Updated

Nov 18, 2024

Answers

[2024.x - Vivado Known Issues](#)

Documentation

[Release Notes](#)
[OS Support Update](#)
[What's New in Vivado](#)

Support Forums

[Installation and Licensing](#)



AMD (Xilinx) Vivado Kurulumu

Select Product to Install



Select a product to continue installation. You will be able to customize the content in the next page.

☐ Vitis

Installs Vitis Core Development Kit for embedded software and application acceleration development on AMD platforms. Vitis installation includes Vivado Design Suite. Users can also install Vitis Model Composer to design for AI Engines and Programmable Logic in MATLAB and Simulink. There is an option to install Power Design Manager for power estimation of Versal, UltraScale+, and Kria products.

☒ **Vivado**

Includes the full complement of Vivado Design Suite tools for design, including C-based design with Vitis High-Level Synthesis, implementation, verification and device programming. Complete device support, cable driver, and Document Navigator included. Users can also install Vitis Model Composer to design for AI Engines and Programmable Logic in MATLAB and Simulink. Users can select to install the Vitis Embedded Development which is an embedded software development package. There is an option to install Power Design Manager for power estimation of Versal, UltraScale+, and Kria products.

☐ Vitis Embedded Development

The Vitis Embedded Development is a standalone embedded software development package for creating, building, debugging, optimizing, and downloading software applications for AMD FPGA processors. It includes a new Vitis IDE with its new backend Vitis Server, as well as the classic command line utilities such as `hw_server`, `bootgen` and `program_flash`.

☐ BootGen

Installs Bootgen for creating bootable images targeting AMD SoCs and FPGAs.

☐ Lab Edition

Installs only the Vivado Lab Edition. This standalone product includes Vivado Design Programmer, Vivado Logic Analyzer and UpdateMEM tools.

☐ Hardware Server

Installs hardware server and JTAG cable drivers for remote debugging.

☐ Power Design Manager (PDM)

Copyright © 1986-2022 Xilinx, Inc. All rights reserved.
Copyright © 2022-2025 Advanced Micro Devices, Inc. All rights reserved.

< Back

Next >

Cancel

Select Edition to Install



Select an edition to continue installation. You will be able to customize the content in the next page.

☒ **Vivado ML Standard**

Vivado ML Standard Edition is the no-cost, device limited version of the Vivado ML Enterprise edition. Users can add Vitis Model Composer which is an AMD toolbox for MATLAB and Simulink to design for AI Engines and Programmable Logic. Users can select to install the Vitis Embedded Development which is an embedded software development package. If you have been using AMD System Generator for DSP, you can continue development using Vitis Model Composer. There is an option to install Power Design Manager for power estimation of Versal, UltraScale+, and Kria products.

☐ Vivado ML Enterprise

Vivado ML Enterprise Edition includes the full complement of Vivado Design Suite tools for design, including C-based design with Vitis HLS, implementation, verification, and device programming. Complete device support, cable drivers, and documentation Navigator are included. Users can add Vitis Model Composer which is an AMD toolbox for MATLAB and Simulink to design for AI Engines and Programmable Logic. Users can select to install the Vitis Embedded Development which is an embedded software development package. If you have been using AMD System Generator for DSP, you can continue development using Vitis Model Composer. There is an option to install Power Design Manager for power estimation of Versal, UltraScale+, and Kria products.

Copyright © 1986-2022 Xilinx, Inc. All rights reserved.
Copyright © 2022-2025 Advanced Micro Devices, Inc. All rights reserved.

< Back

Next >

Cancel



AMD (Xilinx) Vivado Kurulumu

Vivado ML Standard



Customize your installation by (de)selecting items in the tree below. Moving cursor over selections below provide additional information.

Vivado ML Standard Edition is the no-cost, device limited version of the Vivado ML Enterprise edition. Users can add Vitis Model Composer which is an AMD toolbox for MATLAB and Simulink to design for AI Engines and Programmable Logic. Users can select to install the Vitis Embedded Development which is an embedded software development package. If you have been using AMD System Generator for DSP, you can continue development using Vitis Model Composer. There is an option to install Power Design Manager for power estimation of Versal, UltraScale+, and Kria products.

- ☒ Design Tools
 - ☒ Vivado Design Suite
 - ☒ Vivado
 - ☒ Vitis HLS
 - ☐ Vitis Model Composer (Toolbox for MATLAB and Simulink. Includes the functionality of System Generator for DSP)
 - ☐ Vitis Embedded Development
 - ☐ Power Design Manager (PDM)
 - ☒ DocNav
- ☒ Devices
 - ☐ Install Devices for Kria SOMs and Starter Kits
 - ☒ Production Devices
 - ☐ SoCs
 - ☒ 7 Series (limited support)
 - ☒ UltraScale (limited support)
 - ☒ UltraScale+ (limited support)
 - ☐ Versal ACAP
 - ☐ Engineering Sample Devices
- ☒ Installation Options
 - ☒ Install Cable Drivers (You MUST disconnect all Xilinx Platform Cable USB II cables before proceeding)

Download Size: 15.5 GB
Disk Space Required: 58.0 GB

Reset to Defaults

Copyright © 1986-2022 Xilinx, Inc. All rights reserved.
Copyright © 2022-2025 Advanced Micro Devices, Inc. All rights reserved.

< Back

Next >

Cancel

Download Size: 15.5 GB
Disk Space Required: 58.0 GB



Geleneksel Tasarım Metodolojileri

Yapısal (Karı Seviyesi)
Modelleme

Veri Akışı (Fonksiyonel)
Modelleme

Davranışsal Modelleme



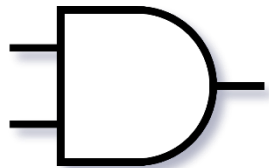
Soyutlama
(Abstraction)



Mantık Kapıları

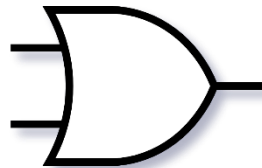
VE Kapısı
(AND Gate)

A	B	Çıkış (Q)
0	0	0
0	1	0
1	0	0
1	1	1



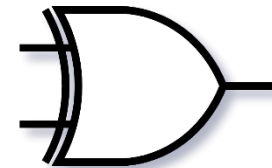
VEYA Kapısı
(OR Gate)

A	B	Çıkış (Q)
0	0	0
0	1	1
1	0	1
1	1	1



ÖZEL-VEYA Kapısı
(XOR Gate)

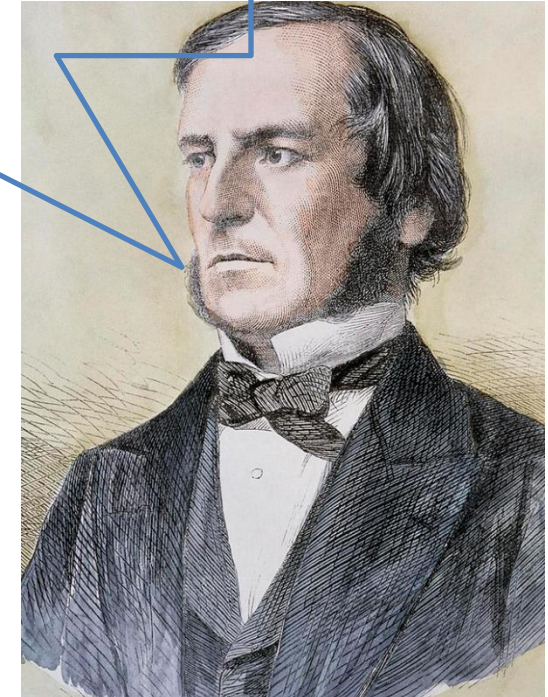
A	B	Çıkış (Q)
0	0	0
0	1	1
1	0	1
1	1	0



ÖRNEK : Geleneksel Tasarım Metodolojileri

A	B	C	Çıkış (Q)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

$$\text{Çıkış (Q)} = A'B'C + A'BC + AB'C' + ABC'$$



George Boole



ÖRNEK : Mantıksal ifadelerin sadeleştirilmesi

$$A'B'C + A'BC + AB'C' + ABC'$$

$$= A'C(B'+B) + AB'C' + ABC'$$

$$= A'C1 + AB'C' + ABC'$$

$$= A'C + AB'C' + ABC'$$

$$= A'C + AC'(B'+B)$$

$$= A'C + AC'1$$

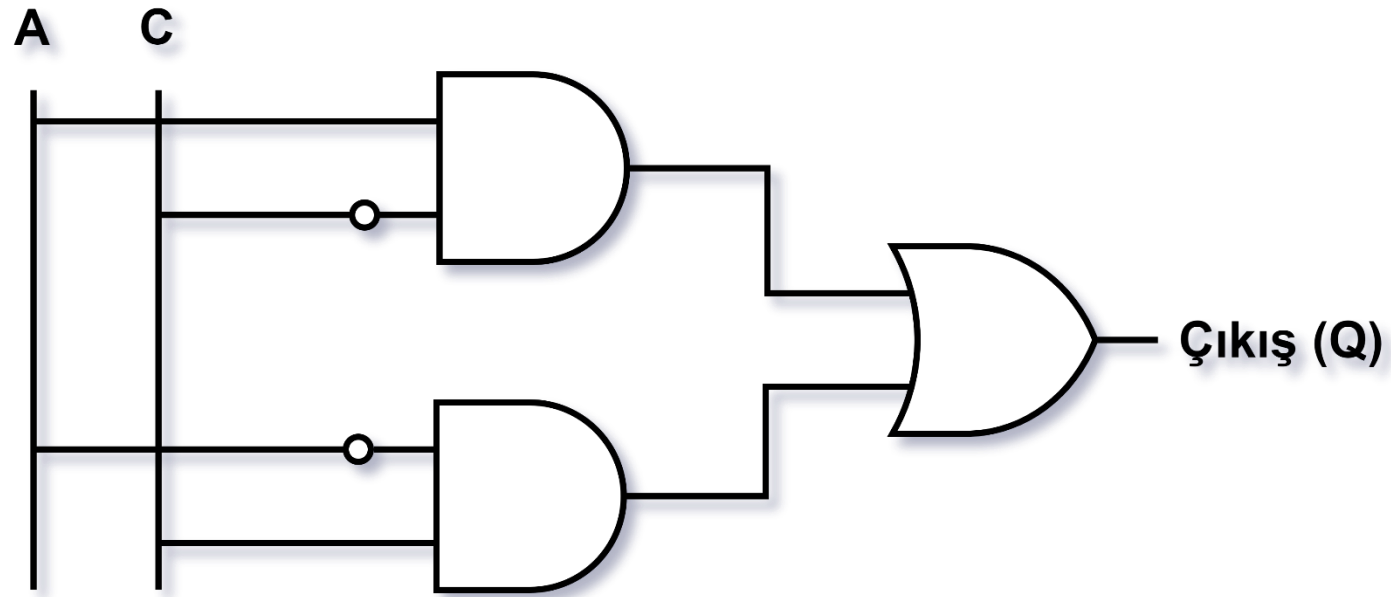
$$= A'C + AC'$$

$\begin{array}{c} C \\ \backslash AB \end{array}$	0	1
00		1
01		1
11	1	
10	1	

$$= A'C + AC'$$

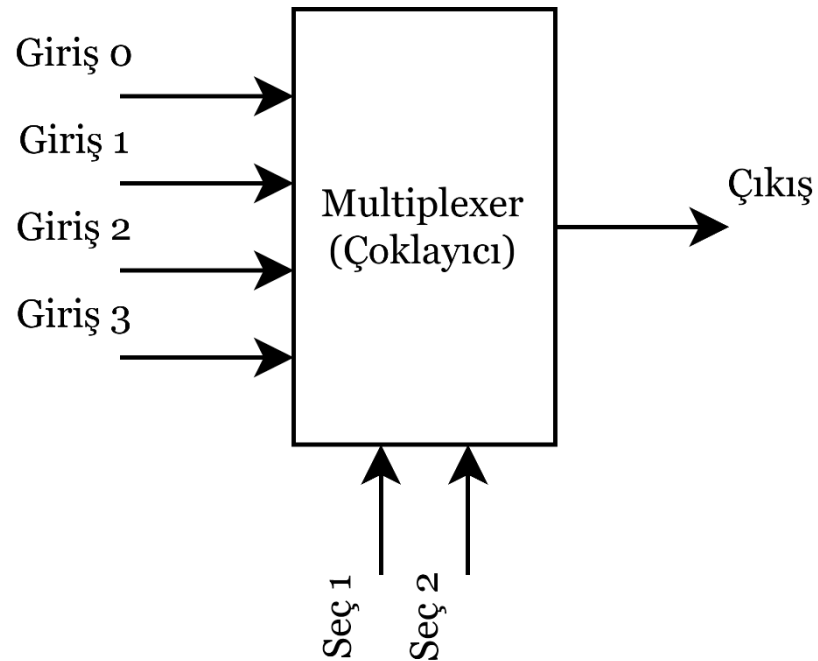


ÖRNEK : Mantıksal ifadelerin sadeleştirilmesi



ÖRNEK : Geleneksel Tasarım Metodolojileri

- Özel bir giriş hattının seçilmesi bir grup seçme hattıyla kontrol edilir. 2^n giriş hattı ve hangi girişin seçileceğini belirleyen bit kombinasyonlarını oluşturan n adet seçme hattı vardır.



Seç1	Seç2	Çıkış
0	0	Giriş0
0	1	Giriş1
1	0	Giriş2
1	1	Giriş3



ÖRNEK : Geleneksel Tasarım Metodolojileri

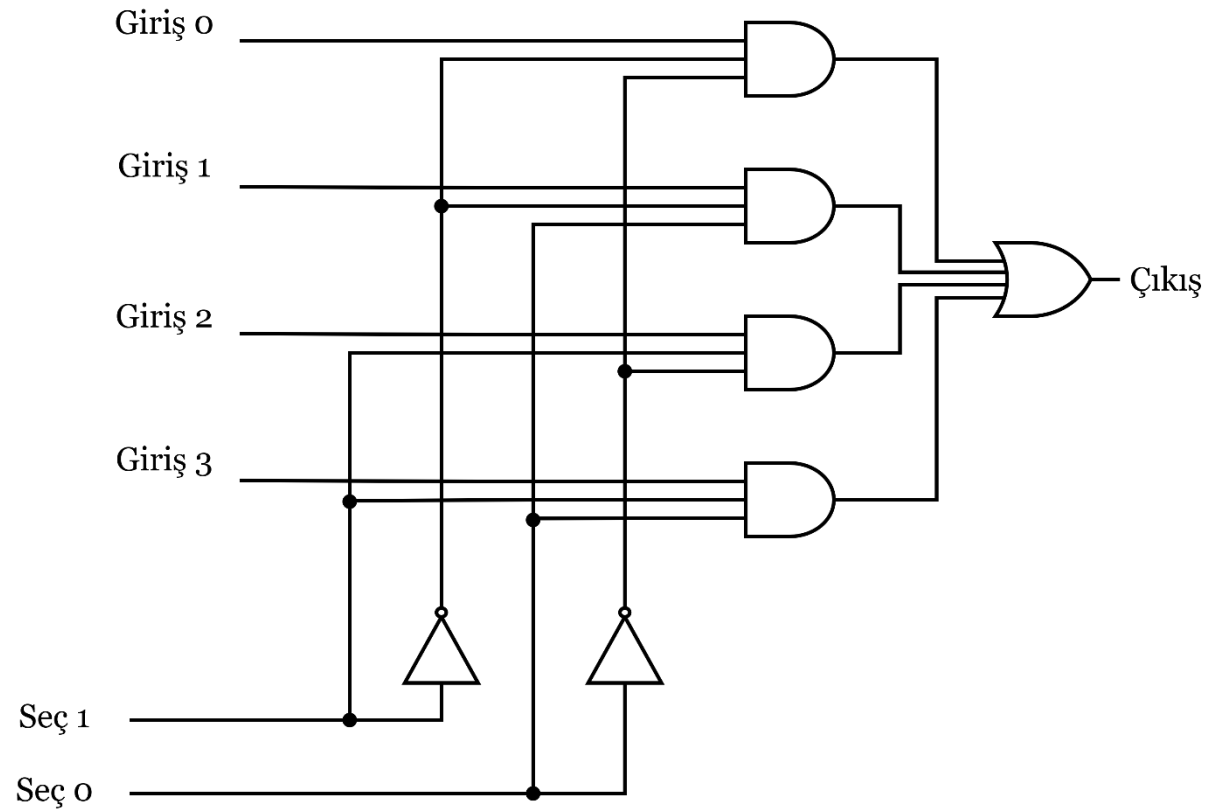
S1	S0	Ç
0	0	Giriş0
0	1	Giriş1
1	0	Giriş2
1	1	Giriş3

Seç1	Seç0	Giriş0	Giriş1	Giriş2	Giriş3	Çıkış
0	0	0	x	X	X	0
0	0	1	x	X	X	1
0	1	X	0	X	X	0
0	1	X	1	X	X	1
1	0	X	X	0	X	0
1	0	X	X	1	X	1
1	1	X	X	X	0	0
1	1	X	X	X	1	1

$$\text{Çıkış} = S0'.S1'.G0 + S0.S1'.G1 + S0'.S1.G2 + S0.S1.G3$$



ÖRNEK : Geleneksel Tasarım Metodolojileri



Yapısal Modelleme (Structural Modeling)

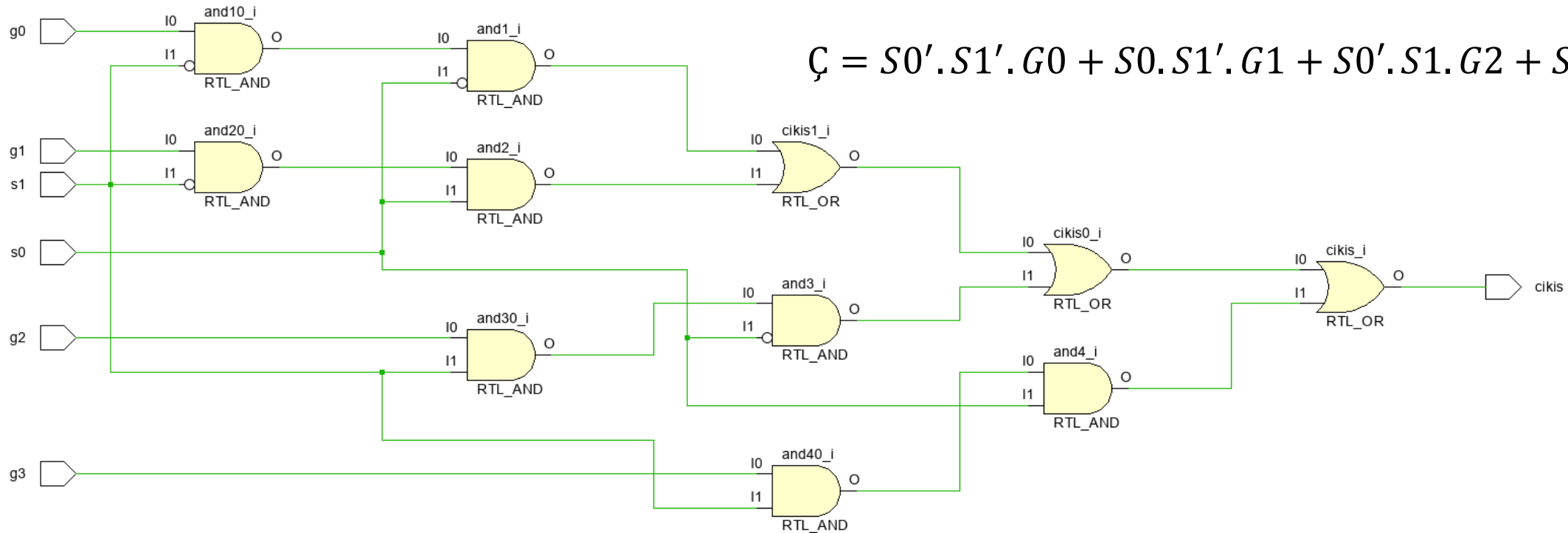
- Bu yöntemde kullanılacak her eleman bir yapı olarak tanımlanmış olmalıdır. Bu sebeple bu tanımlama yöntemine kapı seviyesinde modelleme (Gate-Level Modelling) de denmektedir.

```
1 `timescale 1ns / 1ps
2
3 module Structural(g0, g1, g2, g3, s0, s1, cikis);
4
5     input s0, s1;
6     input g0, g1, g2, g3;
7     output cikis;
8
9     wire not_s0, not_s1;
10    wire and1, and2, and3, and4;
11
12    not (not_s0, s0);
13    not (not_s1, s1);
14    and (and1, g0, not_s1, not_s0);
15    and (and2, g1, not_s1, s0);
16    and (and3, g2, s1, not_s0);
17    or  (cikis, and1, and2, and3, and4);
18    and (and4, g3, s1, s0);
19 endmodule
20
```



RTL Tasarım Çıktıları (Yapısal Modelleme)

- **Register Transistor Level** ifadesinin kısaltılmasıdır.
- Girişler ile Çıkışlar arasındaki ilişkiyi dijital sinyallerin akışı ve bu sinyaller üzerinde gerçekleştirilen mantıksal işlemleri modelleyen bir tasarım soyutlamasıdır.



Veri Akışı Modelleme (Dataflow Modeling)

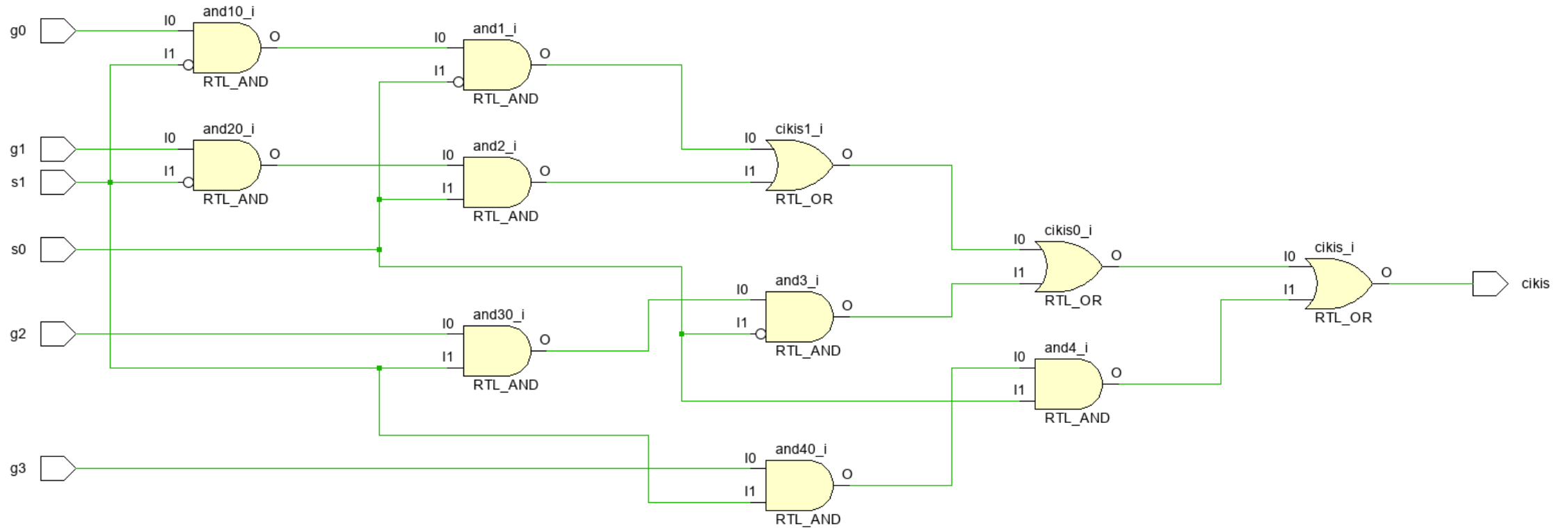
- Bu metotta giriş ve çıkış portları arasındaki ilişki bir fonksiyon olarak ifade edilir. Bu sebepten dolayı bu yönetime fonksiyonel modelleme (Functional Modelling) de denir.

```
1 `timescale 1ns / 1ps
2
3 module Dataflow(g0, g1, g2, g3, s0, s1, cikis);
4
5     input s0, s1;
6     input g0, g1, g2, g3;
7     output cikis;
8
9     assign
10         cikis = (~s0 & ~s1 & g0) | (~s0 & s1 & g1) | (s0 & ~s1 & g2) | (s0 & s1 & g
11     endmodule
```



RTL Tasarım Çıktıları (Veri Akışı Modelleme)

$$\zeta = S0'.S1'.G0 + S0.S1'.G1 + S0'.S1.G2 + S0.S1.G3$$



Davranışsal Modelleme (Behavioral Modelling)

- Tasarlanması istenen sayısal sistem davranışı ile ifade edilir. Yani, koşullu ve özyinemeli ifadelerle karşılık gelen HDL anahtar kelimeleri kullanılabilir.

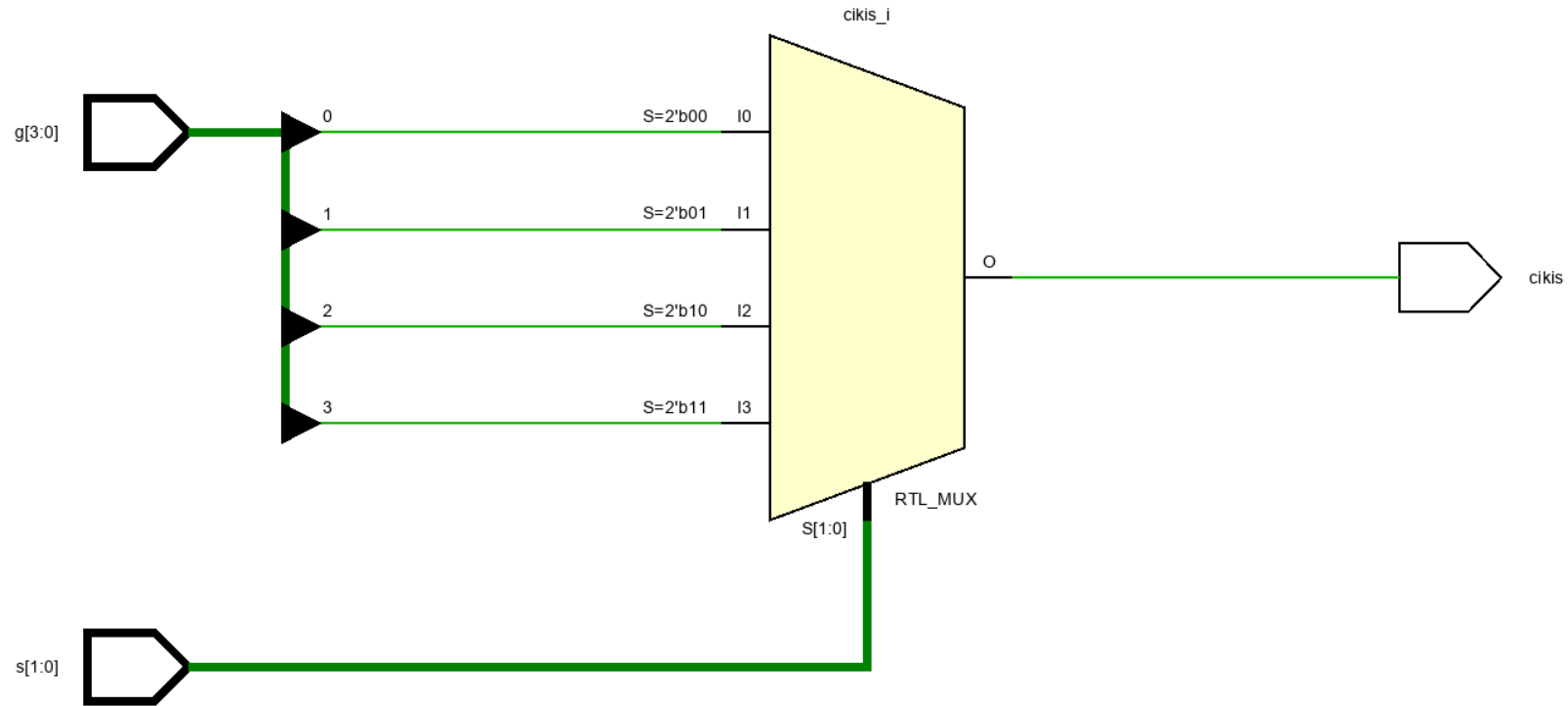
```
1 `timescale 1ns / 1ps
2
3 module Behavioral(g,s,cikis);
4
5     input [1:0] s;
6     input [3:0] g;
7     output reg cikis;
8
9     always @(s,g,cikis) begin
10         case (s)
11             2'b00 : cikis <= g[0];
12             2'b01 : cikis <= g[1];
13             2'b10 : cikis <= g[2];
14             2'b11 : cikis <= g[3];
15         endcase
16     end
17
18 endmodule
```

Klasik programlama dillerindeki döngü yapısına benzer.

Klasik programlama dillerindeki şart yapıları gibi.

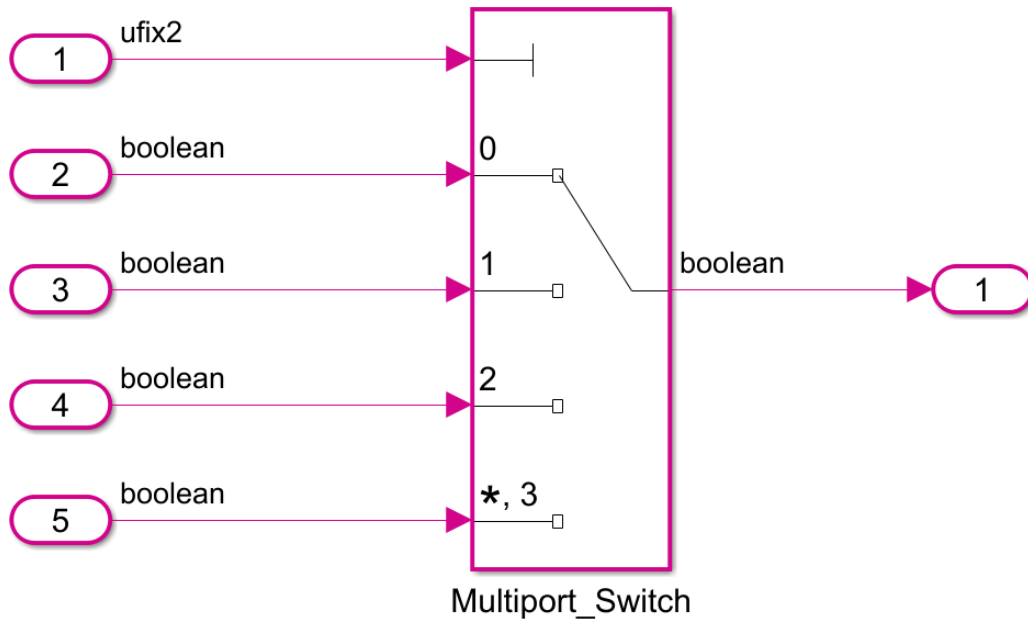


RTL Tasarım Çıktıları (Behavioural Modelling)



Model Tabanlı Tasarım ile MUX Tasarımı

■ 4 Girişli MUX Tasarımı



```
38  input  [1:0] In1;  // ufix2
39  input  In2;
40  input  In3;
41  input  In4;
42  input  In5;
43  output Out1;
44
45
46  wire Multiport_Switch_out1;
47
48
49  assign Multiport_Switch_out1 = (In1 == 2'b00 ? In2 :
50                                (In1 == 2'b01 ? In3 :
51                                (In1 == 2'b10 ? In4 :
52                                In5)));
53
54  assign Out1 = Multiport_Switch_out1;
```

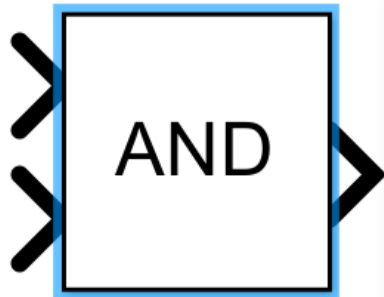


Lisans Seviyesi Sayısal Tasarım Kazanımları

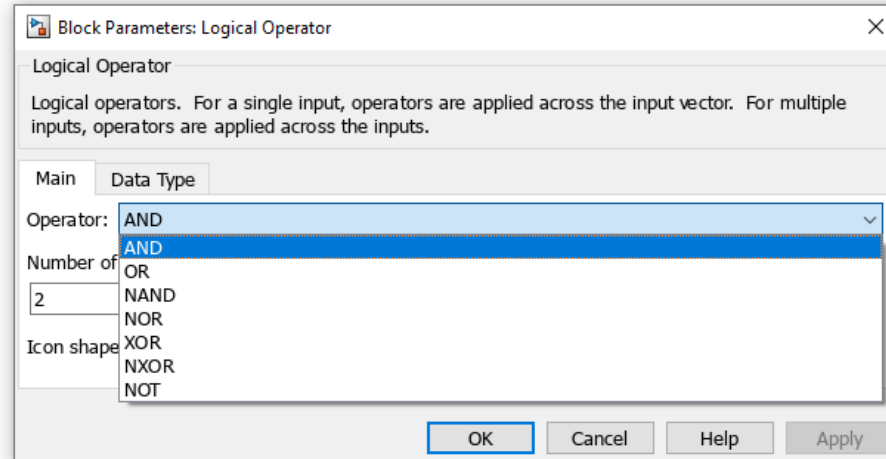
- **Sayı Sistemleri**
- **Mantık Kapıları**
- **Boolean Cebri**
- **Combinational Devreler**
 - Mux
 - Encoder
 - Decoder
 - Comparators
 - Adder
- **Sequential Devreler**
 - Counter
 - Shift Register
 - FSM
 - Mealy Moore
 - Sequence Detector
 - UART
- **Hafıza Birimi Tasarımı**
 - Register
 - RAM
 - FIFO



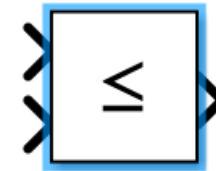
Model Tabanlı Mantık Tasarımı



Logical
Operator



Logical
Operator

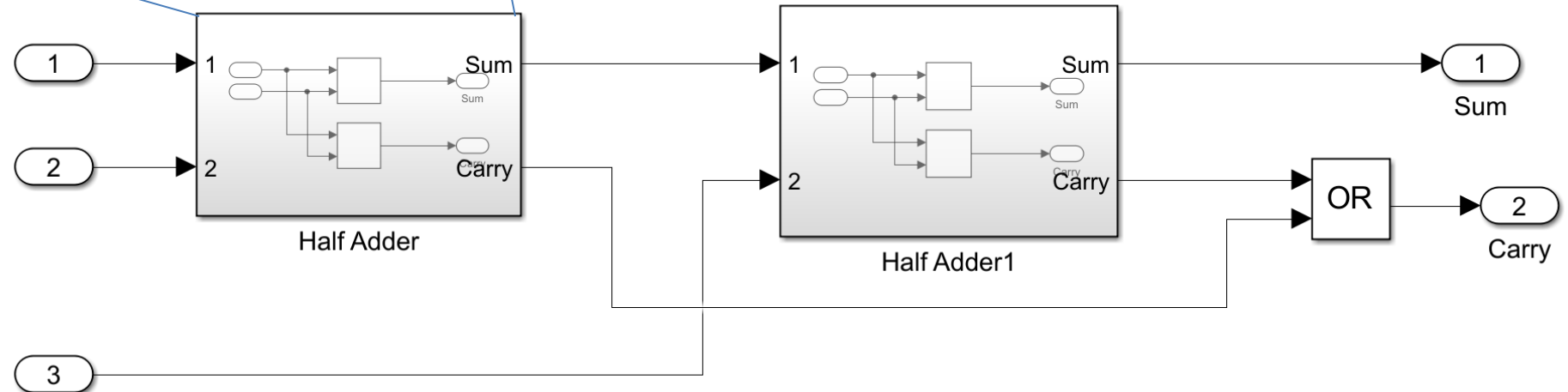
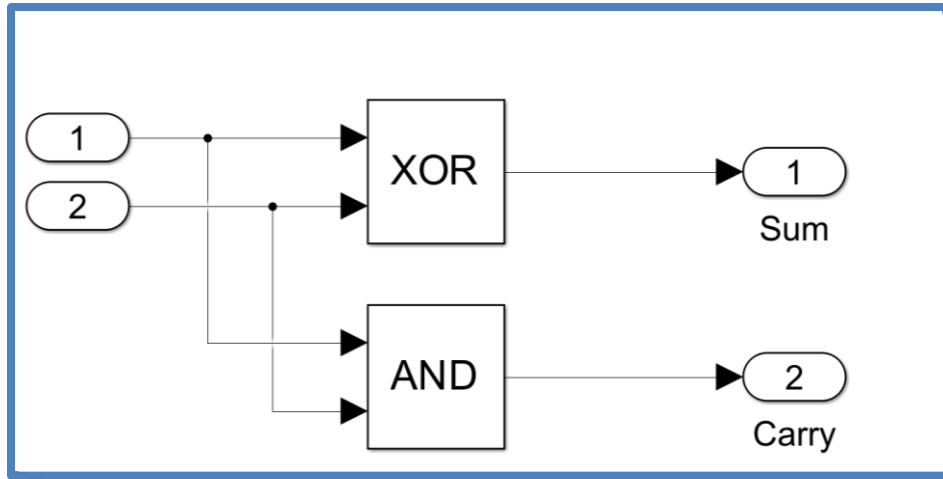


Relational
Operator



Bitwise
Operator

Model Tabanlı Full Adder Tasarımı

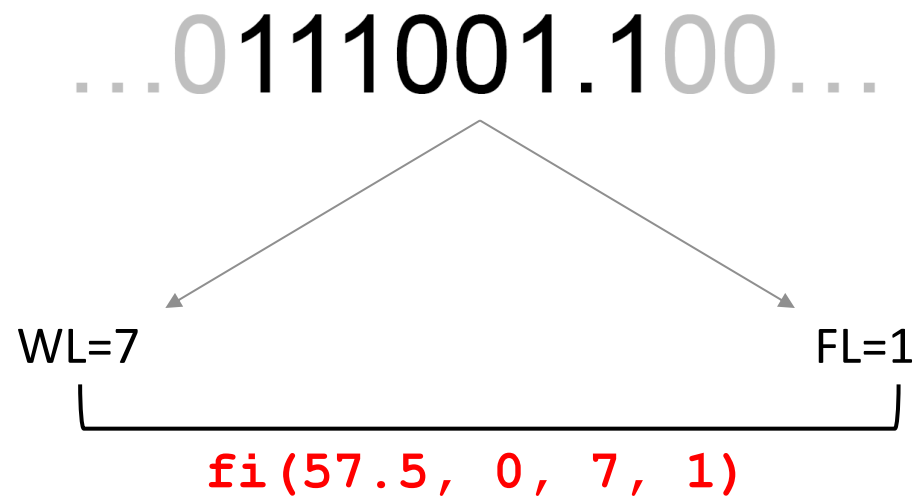


İkili Sayı Sistemi

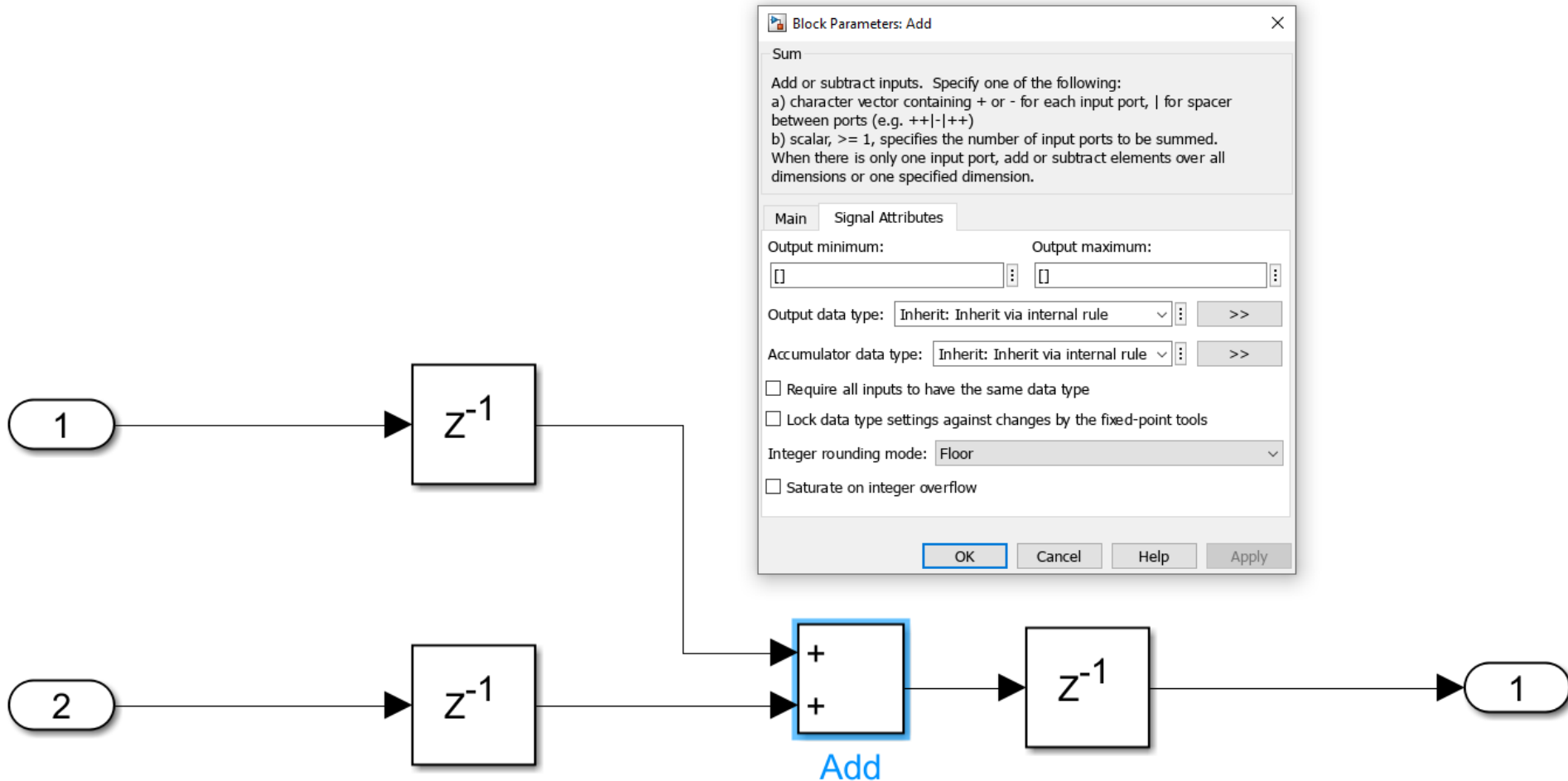
- Create fixed-point objects using the `fi` function:

`fi(value, signedness, wordLength, fractionLength)`

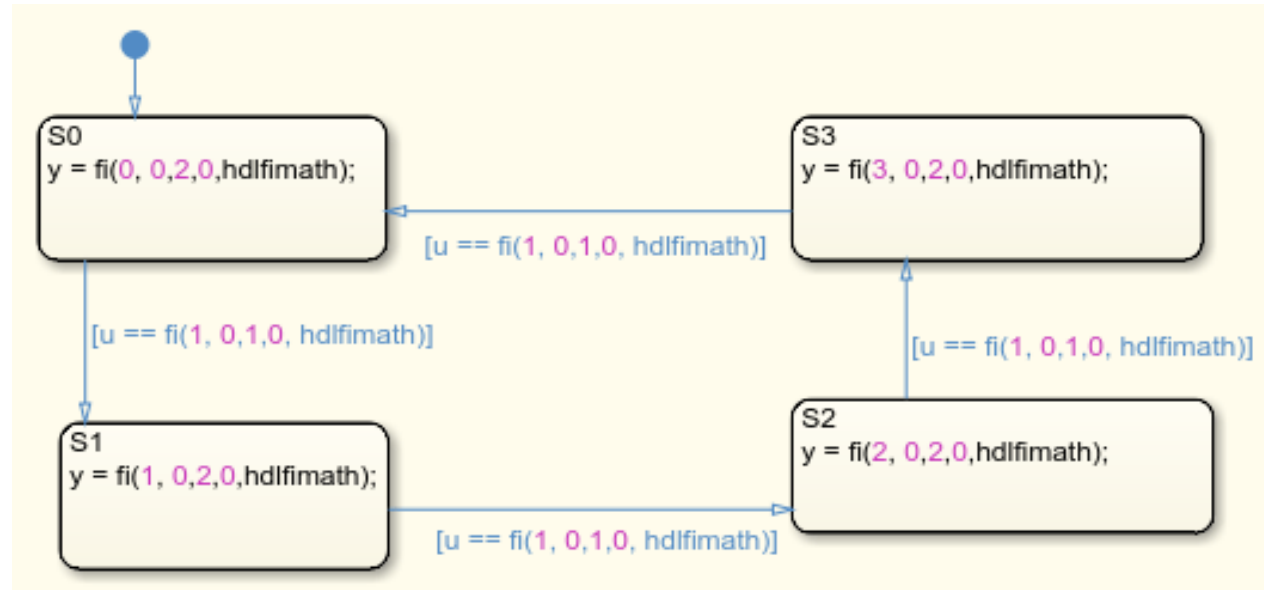
- Example



Model Tabanlı Full Adder Tasarımı



Ardışıl (Sequential) Devreler için Stateflow



```
always @(posedge clk or posedge reset)
begin : Mealy_Chart_1_process
    if (reset == 1'b1) begin
        is_Mealy_Chart <= is_Mealy_Chart_IN_S0;
    end
    else begin
        if (enb) begin
            is_Mealy_Chart <= is_Mealy_Chart_next;
        end
    end
end

always @(is_Mealy_Chart, u) begin
    is_Mealy_Chart_next = is_Mealy_Chart;
    y_1 = 2'b00;
    case ( is_Mealy_Chart)
        is_Mealy_Chart_IN_S0 :
            begin
                if (u == 8'sb00000001) begin
                    y_1 = 2'b00;
                    is_Mealy_Chart_next = is_Mealy_Chart_IN_S1;
                end
            end
        is_Mealy_Chart_IN_S1 :
            begin
                if (u == 8'sb00000001) begin
                    y_1 = 2'b01;
                    is_Mealy_Chart_next = is_Mealy_Chart_IN_S2;
                end
            end
        is_Mealy_Chart_IN_S2 :
            begin
                if (u == 8'sb00000001) begin
                    y_1 = 2'b10;
                    is_Mealy_Chart_next = is_Mealy_Chart_IN_S3;
                end
            end
        default :
            begin
                if (u == 8'sb00000001) begin
                    y_1 = 2'b11;
                    is_Mealy_Chart_next = is_Mealy_Chart_IN_S0;
                end
            end
    endcase
end

assign y = y_1;
```

Donanımda Veri Tipleri

- Arrays can be represented as:

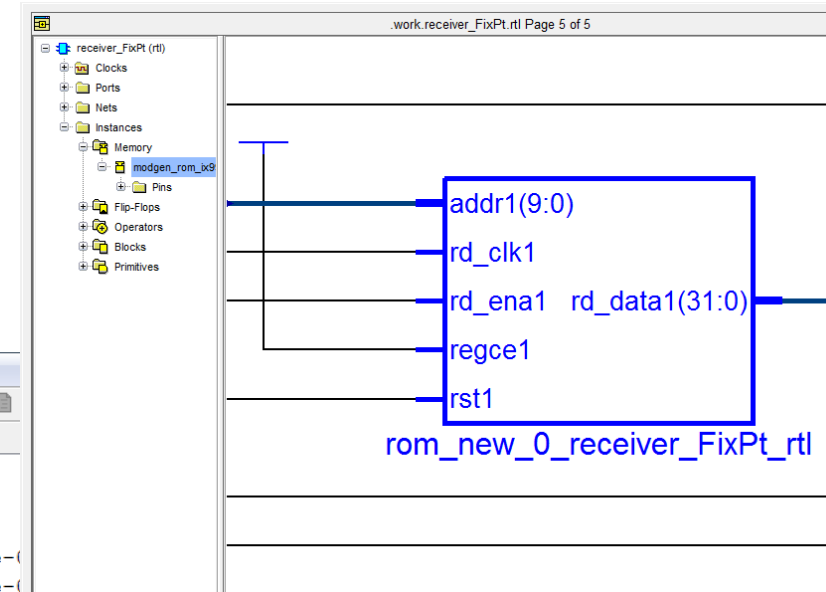
Wires

Flip-Flops

RAM

ROM

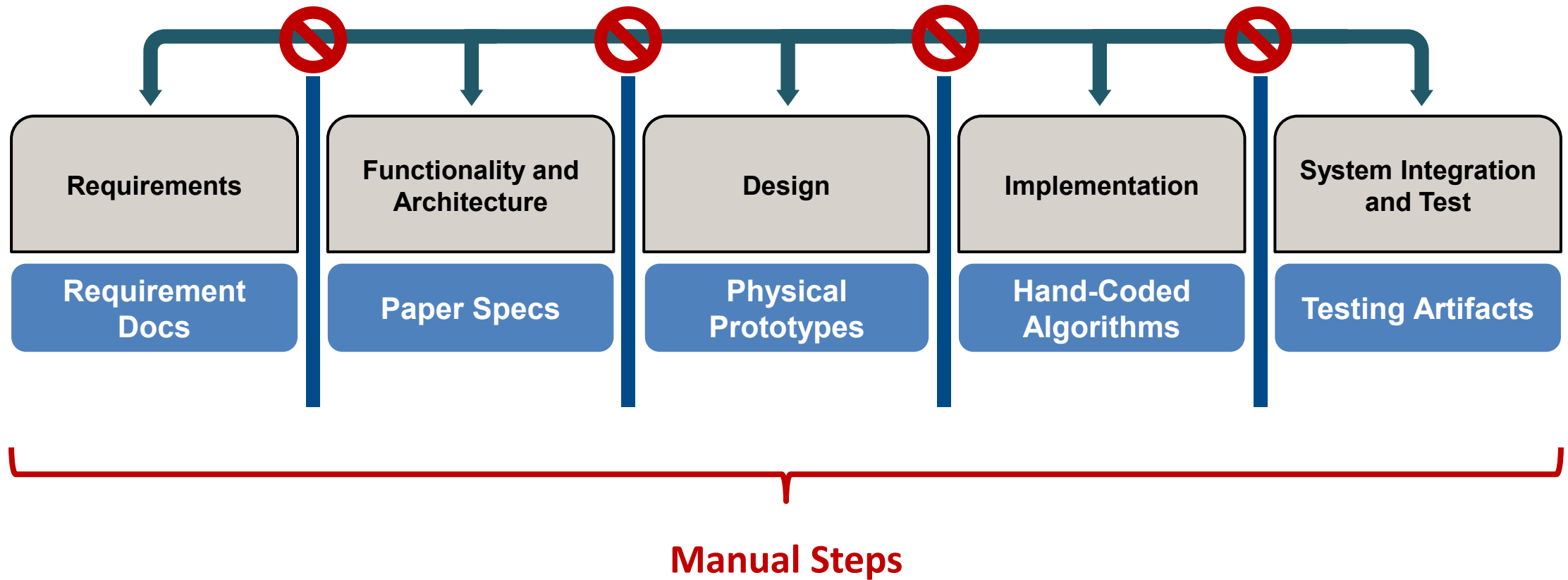
```
File Edit Text Go Cell Tools Debug Desktop Window Help
- 1.0 + ÷ 1.1 x
1 function y = sinusoid_lut
2 %#codegen
3 y = [
4     9.9998117528260111e-01+ 6.1358846491544753e-01
5     9.992470183914450e-01+ 1.2271538285719925e-01
6     9.9983058179582340e-01+ 1.8406729905804820e-02i
7     9.9969881869620425e-01+ 2.4541228522912288e-02i
8     9.9952941750109314e-01+ 3.0674803176636626e-02i
9     9.9932238458834954e-01+ 3.6807222941358832e-02i
10    9.9907772775264536e-01+ 4.2938256934940820e-02i
11    9.9879545620517241e-01+ 4.9067674327418015e-02i
12    9.9847558057329477e-01+ 5.5195244349689941e-02i
13    9.9811811290014918e-01+ 6.1320736302208578e-02i
14    9.9772306664419164e-01+ 6.7443919563664051e-02i
15    9.9729045667869021e-01+ 7.3564563599667426e-02i
16    9.9682029929116567e-01+ 7.9682437971430126e-02i
17    9.9631261218277800e-01+ 8.5797312344439908e-02i
18    9.9576741446765982e-01+ 9.1908956497132752e-02i
```



Hardware
knowledge

Think about how arrays are used in the design to get the best results

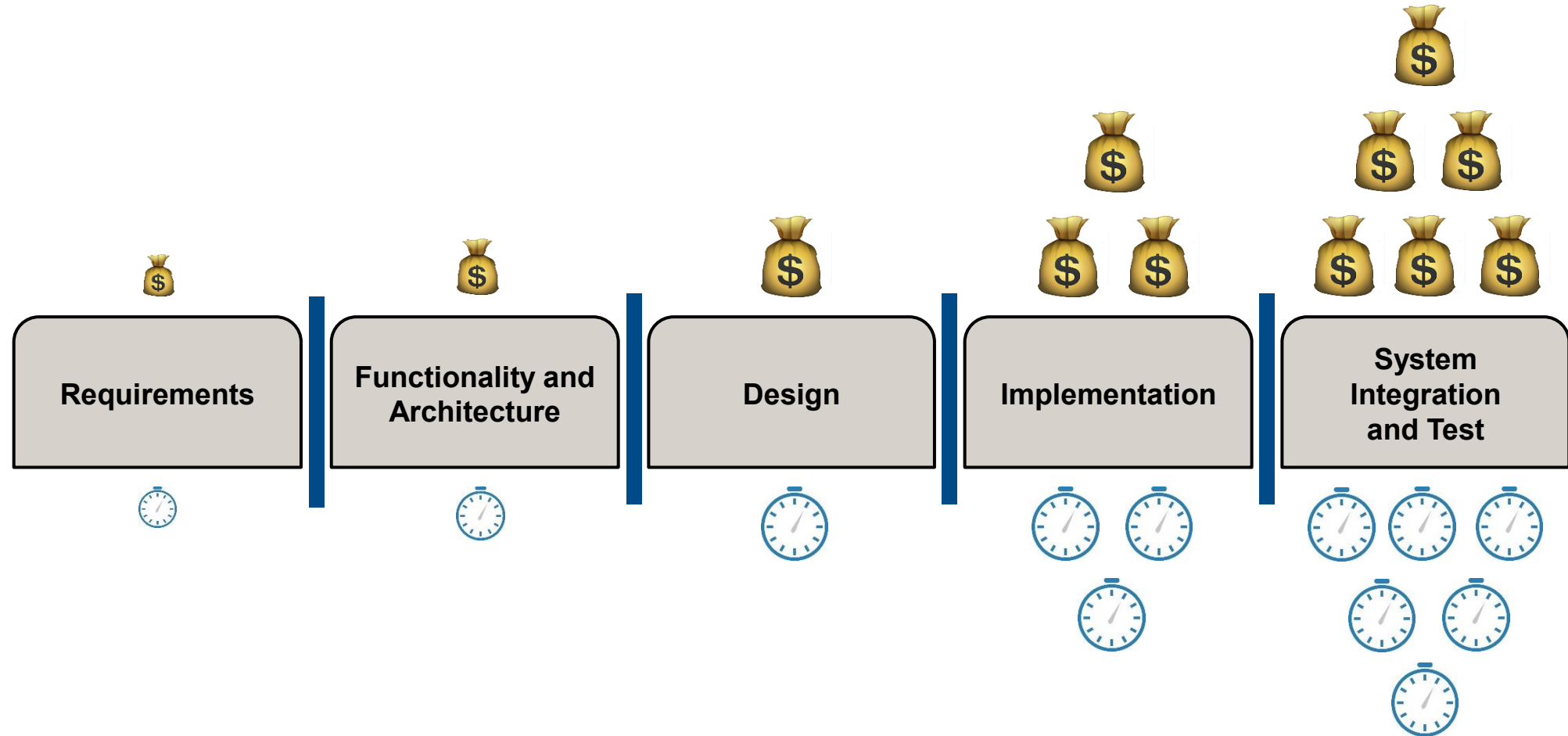
Traditional Development Process



Manual steps introduce **errors** and **slow down** the development process

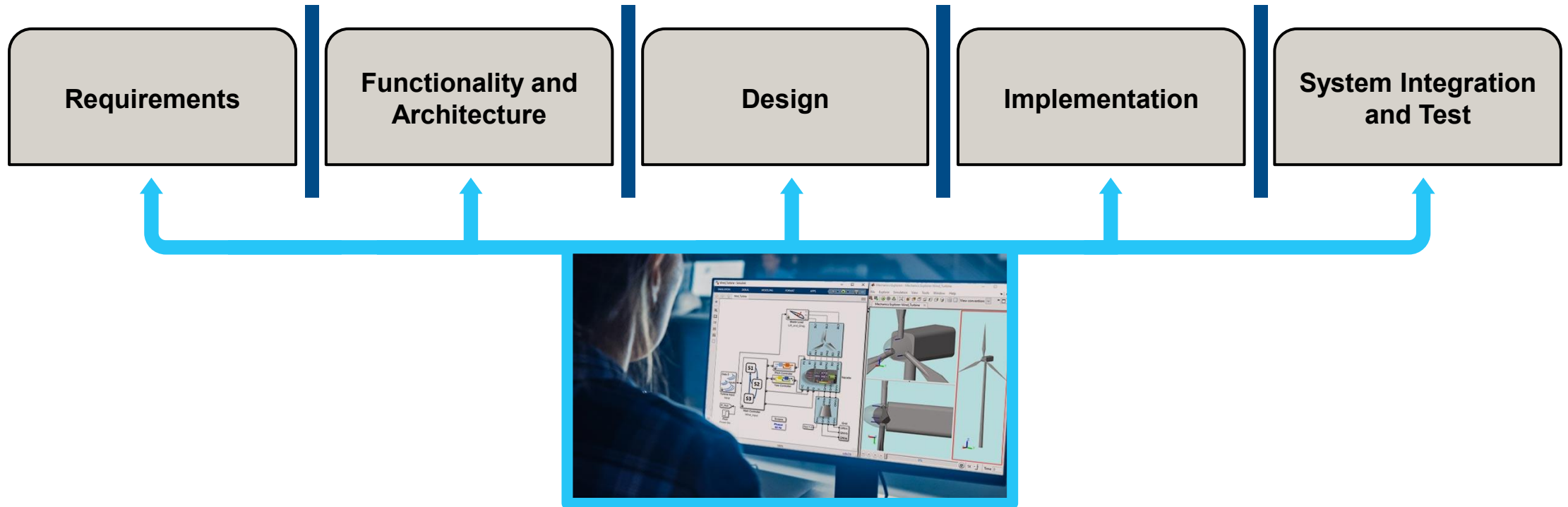
Issues found late in the process are more **costly** and **time-consuming** to fix

Cost to Fix Issues



Time to Fix Issues

Model-Based Design

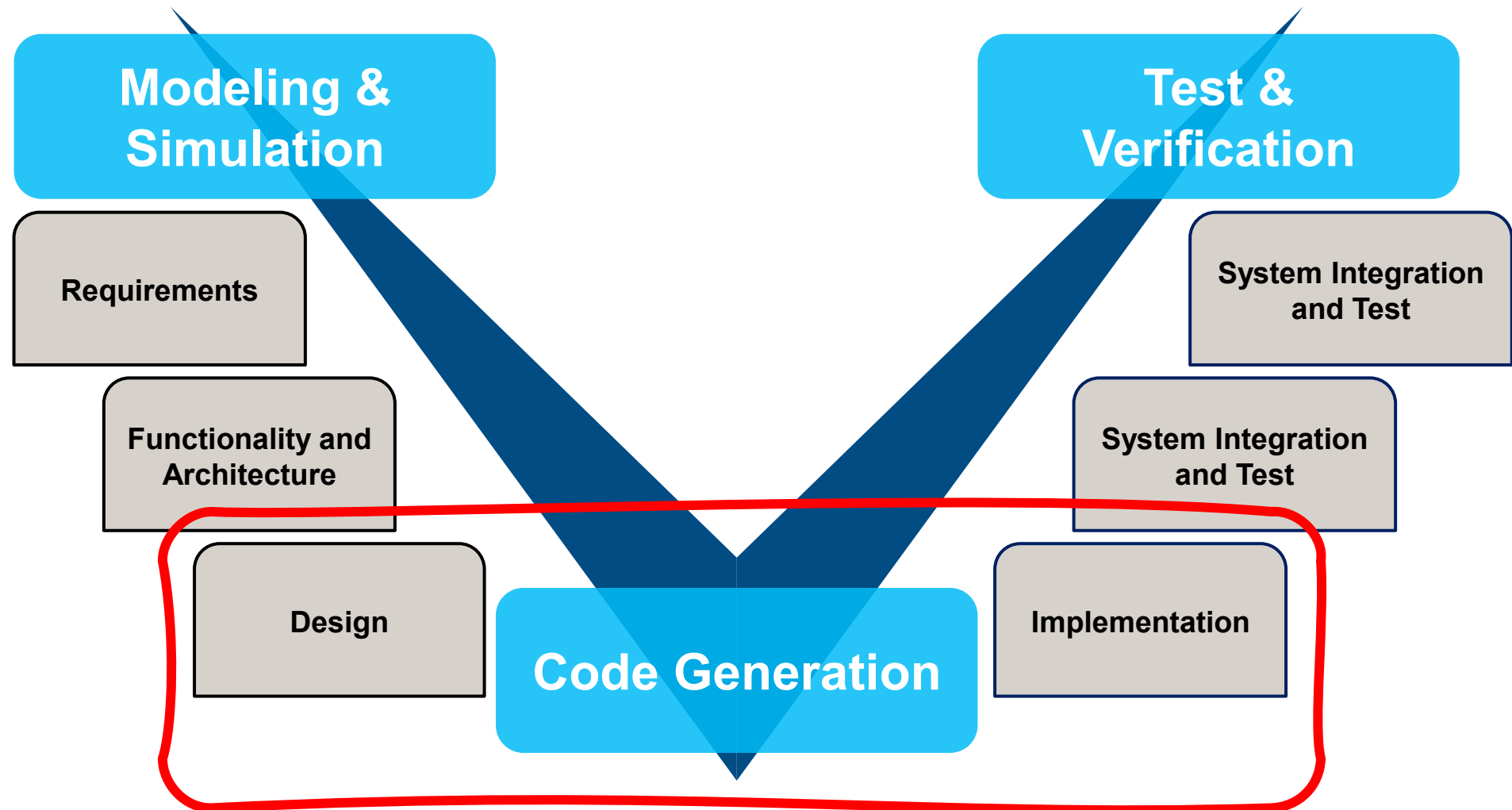


Models are at the **center** of your development process

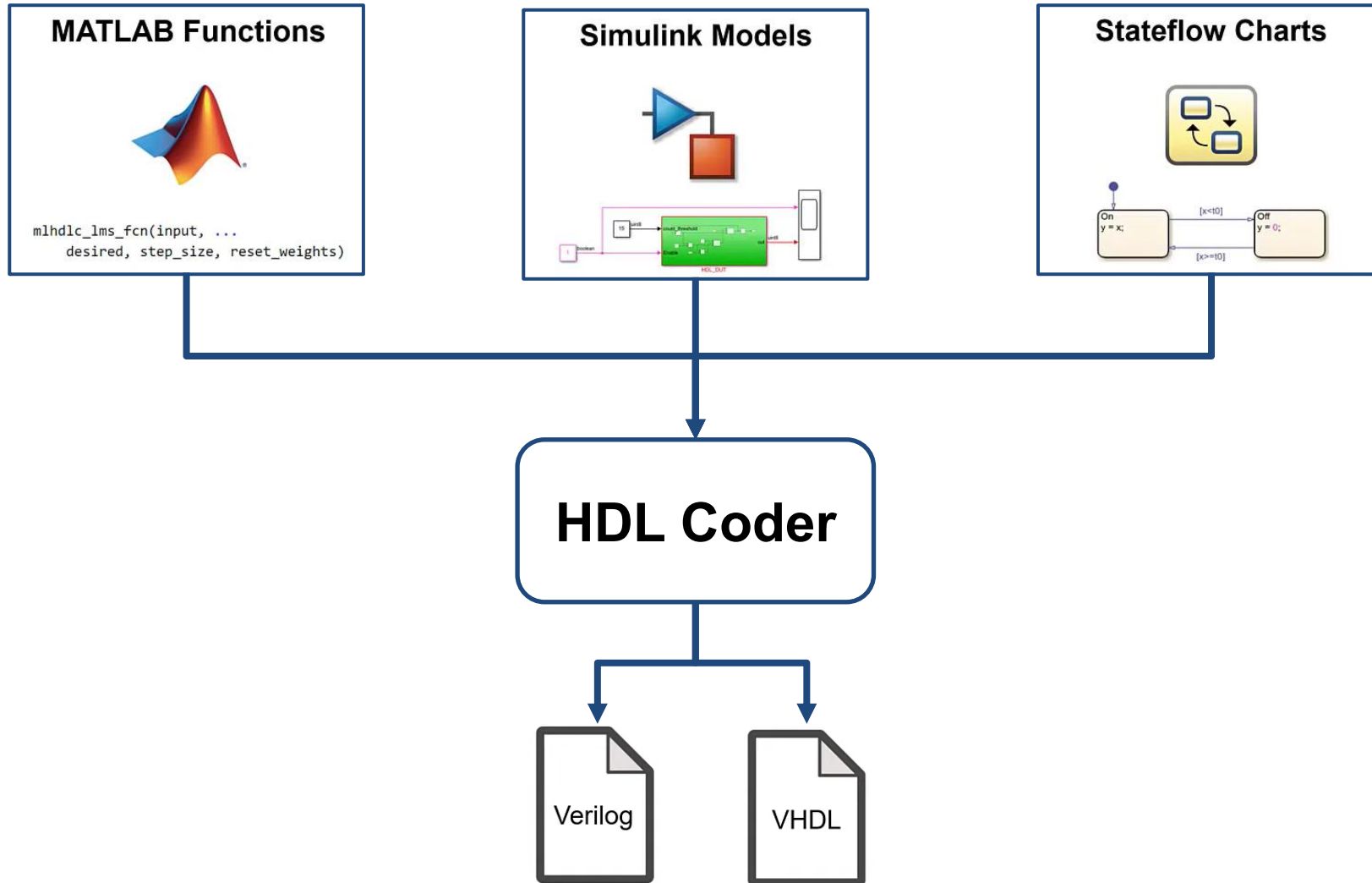
Requirements **capture** and artifact **traceability** throughout the process

V Model

Three key pieces to Model-Based Design



HDL Coder Özeti

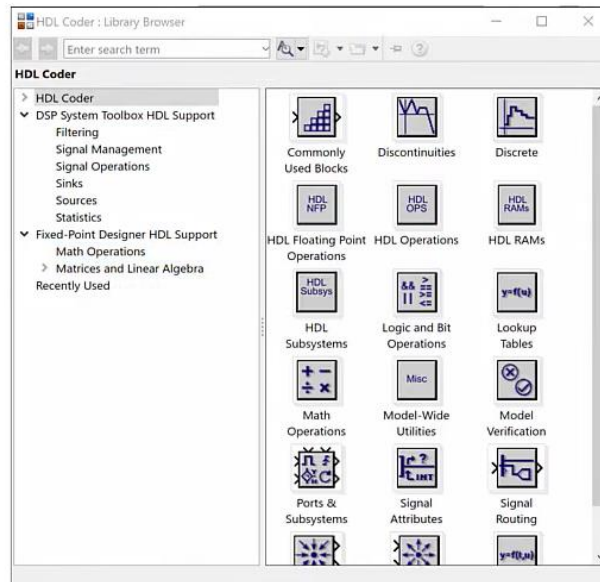


HDL Coder Uyumluluğu

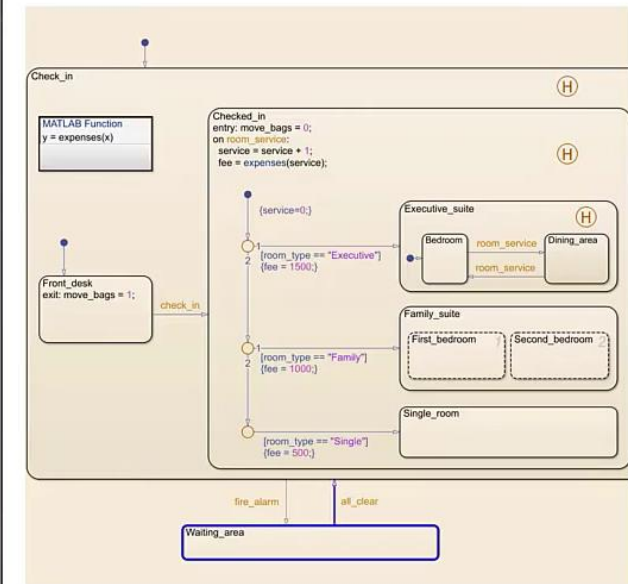
200+ MATLAB Functions

dsp.DCBlocker	Block DC component (offset) from input signal
dsp.Delay	Delay input signal by fixed samples
dsp.VariableFractionalDelay	Delay input by time-varying fractional number of sample periods
Filter Design and Analysis	
dsp.HighpassFilter	FIR or IIR highpass filter
dsp.LowpassFilter	FIR or IIR lowpass filter
dsp.CICCompensationDecimator	Compensate for CIC decimation filter using FIR decimator
dsp.CICCompensationInterpolator	Compensate for CIC interpolation filter using FIR interpolator
Filter Implementation	
Single-Rate Filters	
dsp.FIRFilter	Static or time-varying FIR filter
dsp.HighpassFilter	FIR or IIR highpass filter
dsp.LowpassFilter	FIR or IIR lowpass filter
dsp.BiquadFilter	IIR filter using biquadratic structures
Multirate and Multistage Filters	
dsp.FarrowRateConverter	Polynomial sample rate converter with arbitrary conversion factor
dsp.FIRDecimator	Polyphase FIR decimator
dsp.FIRInterpolator	Polyphase FIR interpolator
dsp.FIRRateConverter	Sample rate converter
dsp.DigitalDownConverter	Translate digital signal from intermediate frequency (IF) band to b

300+ Simulink Blocks

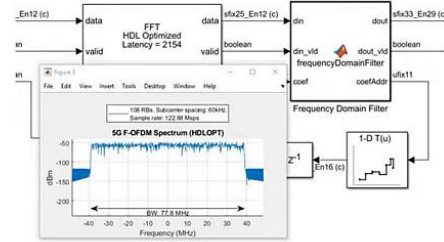


Stateflow Charts



Model Tabanlı Full Adder Tasarımı

Wireless Communications



DSP

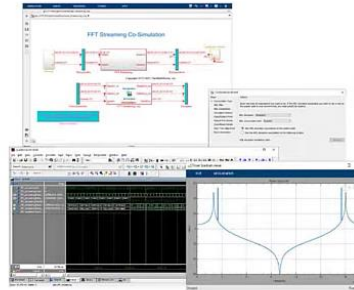
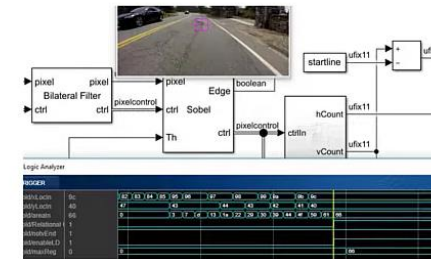
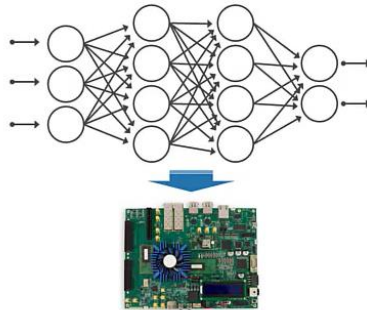


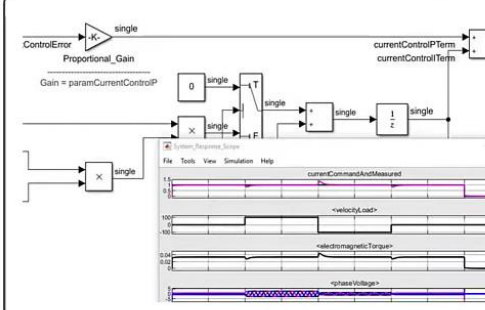
Image and Video Processing



Deep Learning Networks

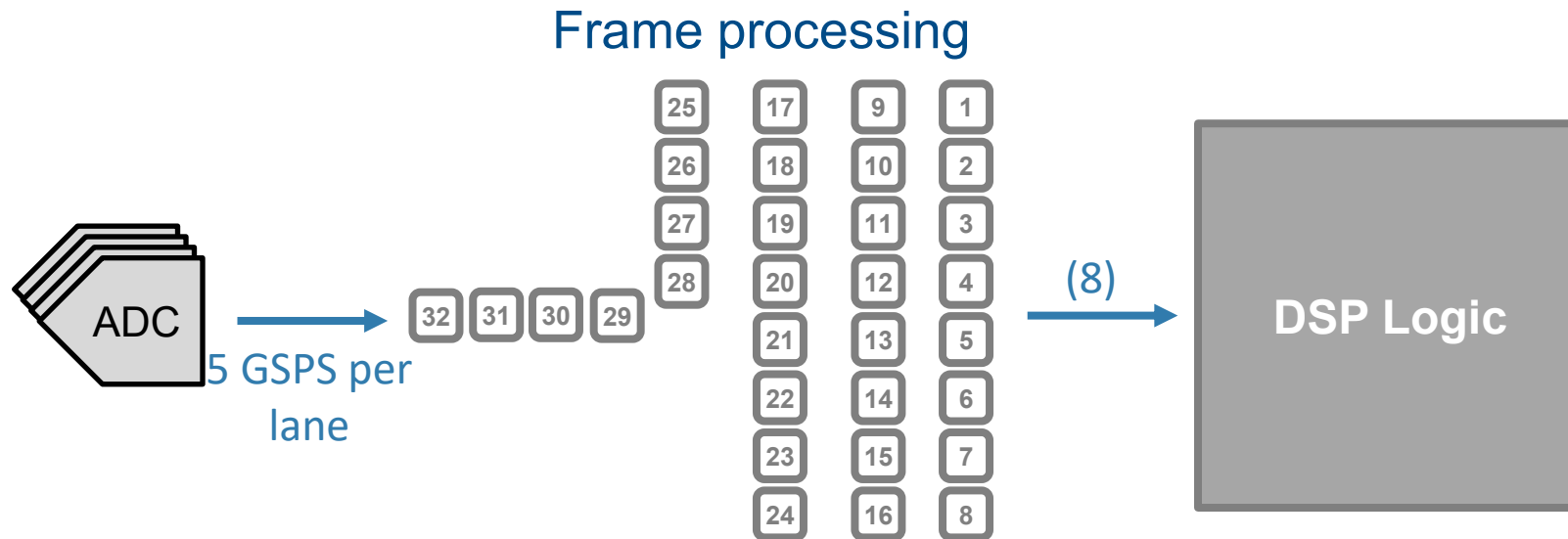
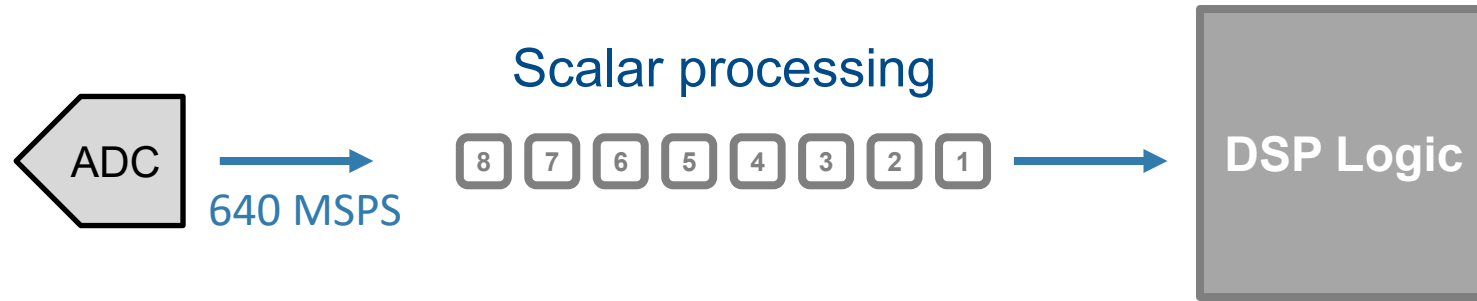


Motor and Power Control

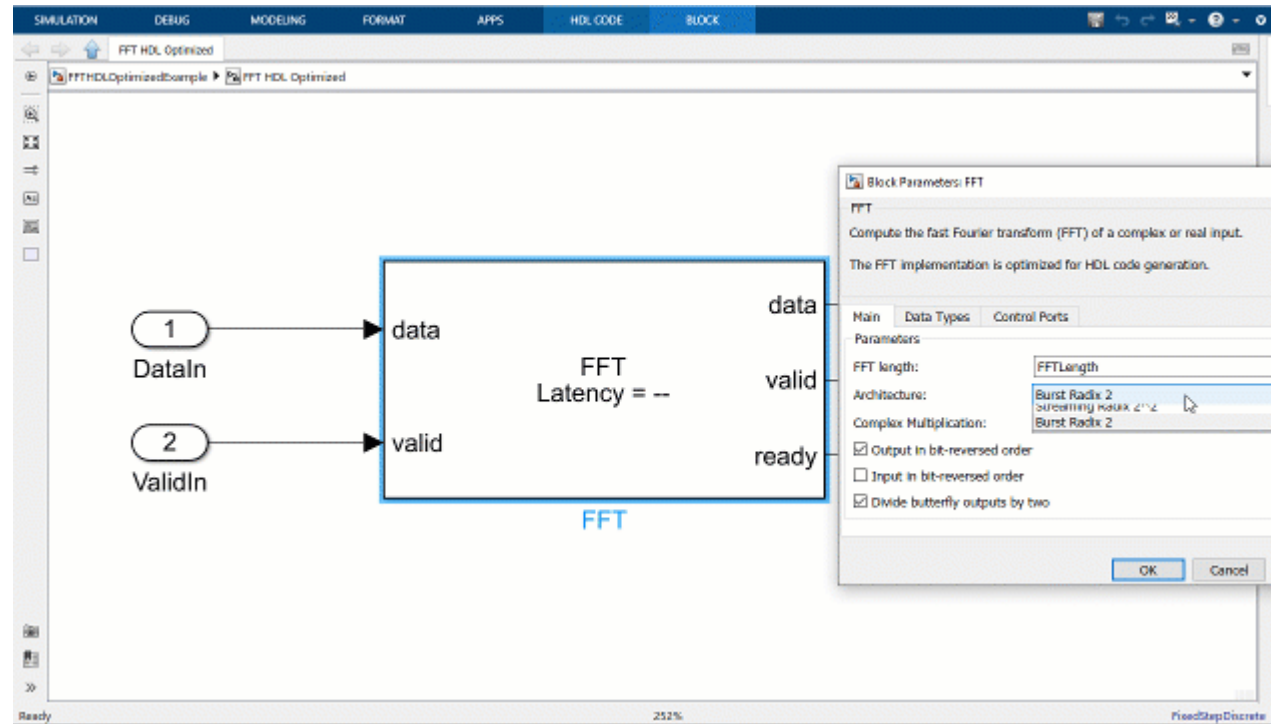


Frame-Based Processing

Parallel implementations for gigasample-per-second throughput



FFT Implementation Exploration



Minimize resources`

- Burst Radix 2
- Latency = 2721 cycles
- Fmax = 672 MHz

Resource summary			
Resource	Usage	Available	Utilization (%)
CLB LUTs	793	274080	0.29
CLB Registers	1229	548160	0.22
DSPs	4	2520	0.16
Block RAM Tile	3	912	0.33

Low latency

- Streaming Radix 2^2
- Latency = 599 cycles
- Fmax = 646 MHz

Resource summary			
Resource	Usage	Available	Utilization (%)
CLB LUTs	3161	274080	1.15
CLB Registers	4917	548160	0.90
DSPs	16	2520	0.63
Block RAM Tile	1	912	0.11

High throughput

- Streaming Radix 2^2
- Latency = 139 cycles
- Fmax = 469 MHz
- Vector input size = 8

3.75 GSPS

Resource summary			
Resource	Usage	Available	Utilization (%)
CLB LUTs	15968	274080	5.83
CLB Registers	28026	548160	5.11
DSPs	108	2520	4.29
Block RAM Tile	1.500000e+00	912	0.16

Implementation metrics are post-synthesis, targeting Xilinx Zynq® UltraScale+ speed grade -2

Desteklenen Fonksiyonlar ve Bloklar

- You can generate efficient code for a subset of **MATLAB Built-in Functions** and **Toolbox Functions** that you can call from MATLAB code. [\[Function List\]](#)

hdl.interpn

Find interpolated value based on n -dimensional input data
Since R2024b

R2025b

[collapse all in page](#)

Syntax

```
Vq = hdl.interpn(X1,X2,...,Xn,V,Xq1,Xq2,...,Xqn)
Vq = hdl.interpn(X1,X2,...,Xn,V,Xq1,Xq2,...,Xqn,interpolation,extrapolation)
```

Description

`Vq = hdl.interpn(X1,X2,...,Xn,V,Xq1,Xq2,...,Xqn)` returns the interpolated values of a function of n variables at a specific query point using linear interpolation and linear extrapolation. `X1,X2,...,Xn` contain the coordinates of the sample points in each dimension. `V` contains the corresponding function values at each sample point. `Xq1,Xq2,...,Xqn` contain the coordinates of the query point.

[example](#)

`Vq = hdl.interpn(X1,X2,...,Xn,V,Xq1,Xq2,...,Xqn,interpolation,extrapolation)` also specifies interpolation and extrapolation approximation methods.

[example](#)

Extended Capabilities

[collapse all](#)

✓ HDL Code Generation

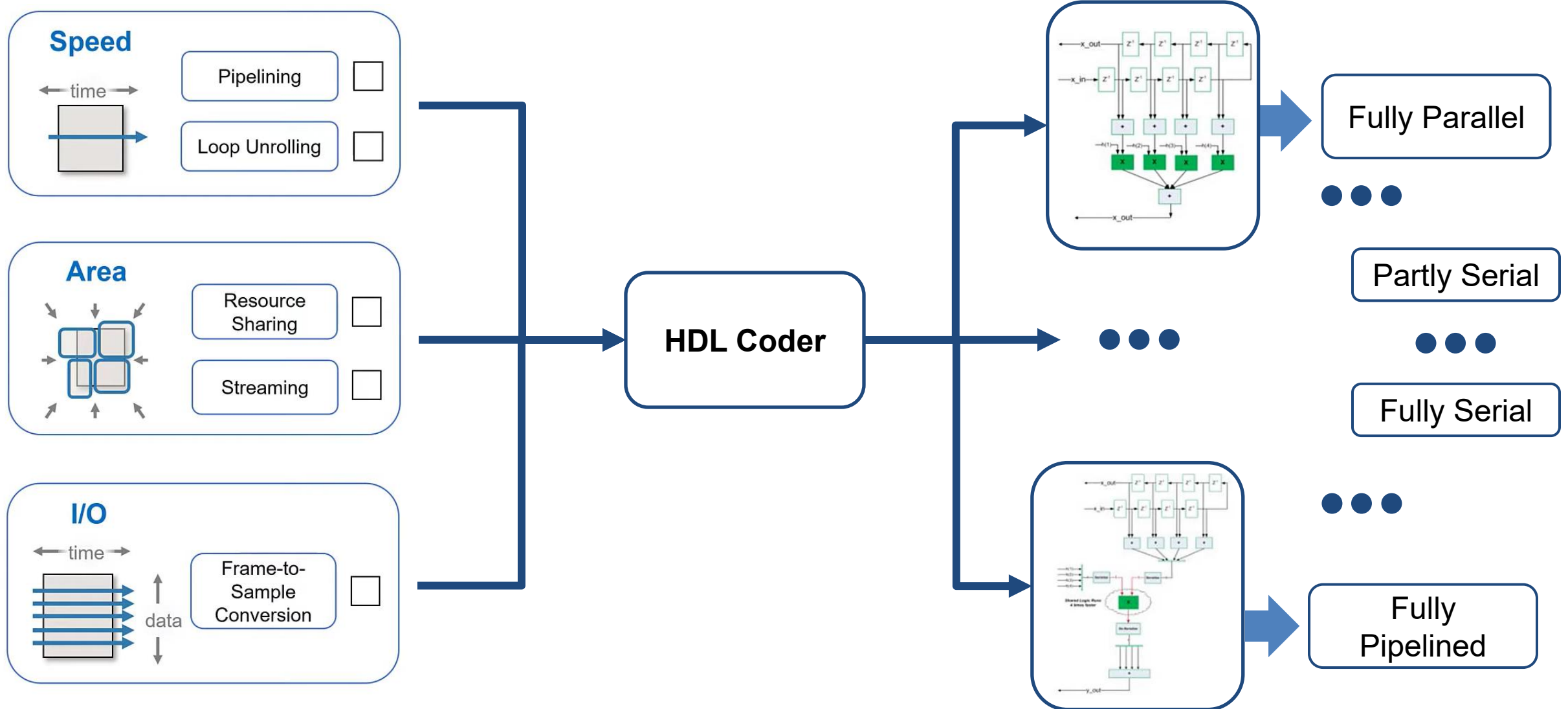
Generate VHDL, Verilog and SystemVerilog code for FPGA and ASIC designs using HDL Coder™.

The approximation methods supported for code generation are "linear" for interpolation, and "linear" and "nearest" for extrapolation.

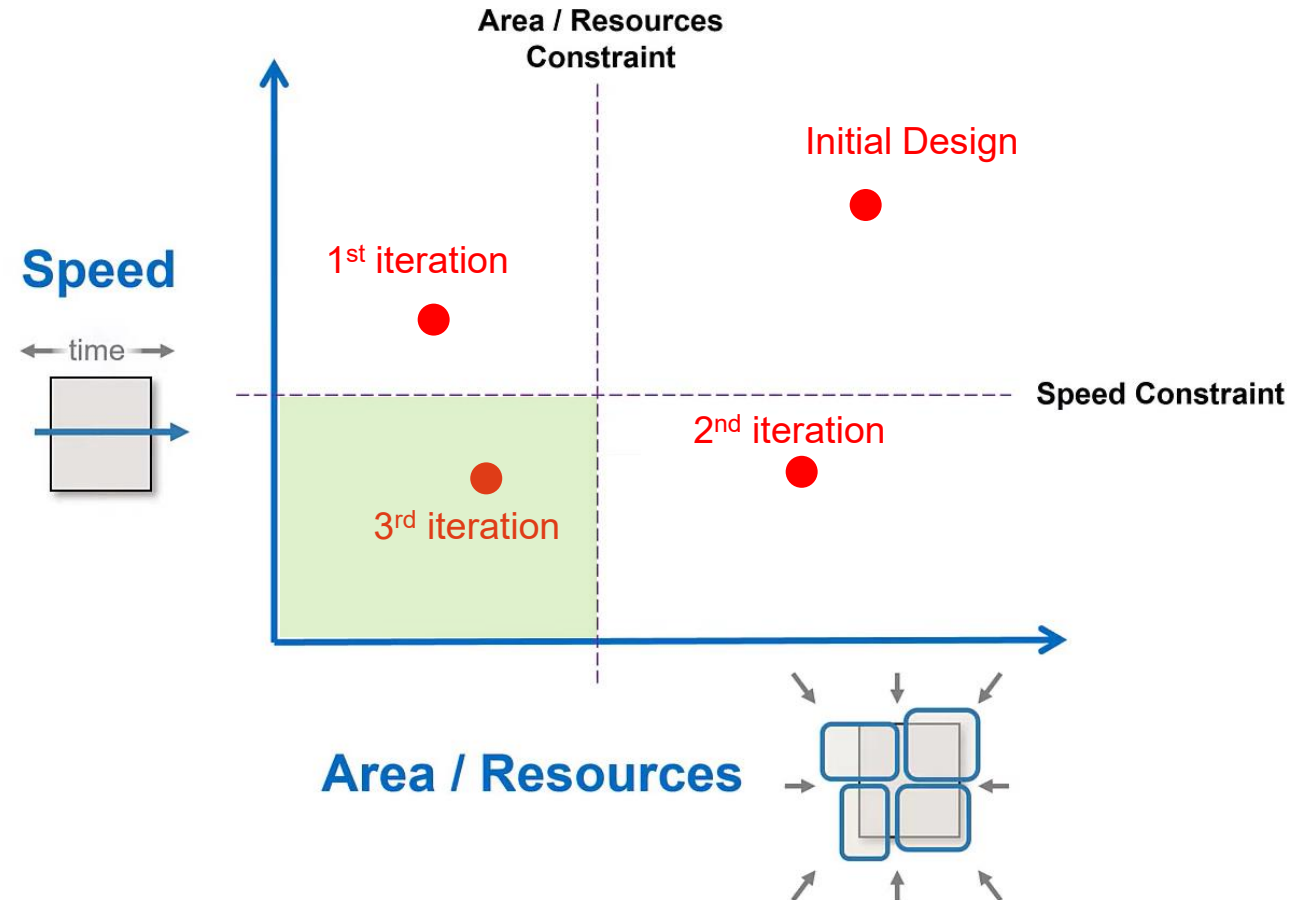
For code generation, the `hdl.interpn` function does not support:

- Fixed-point data type
- Multiple query points

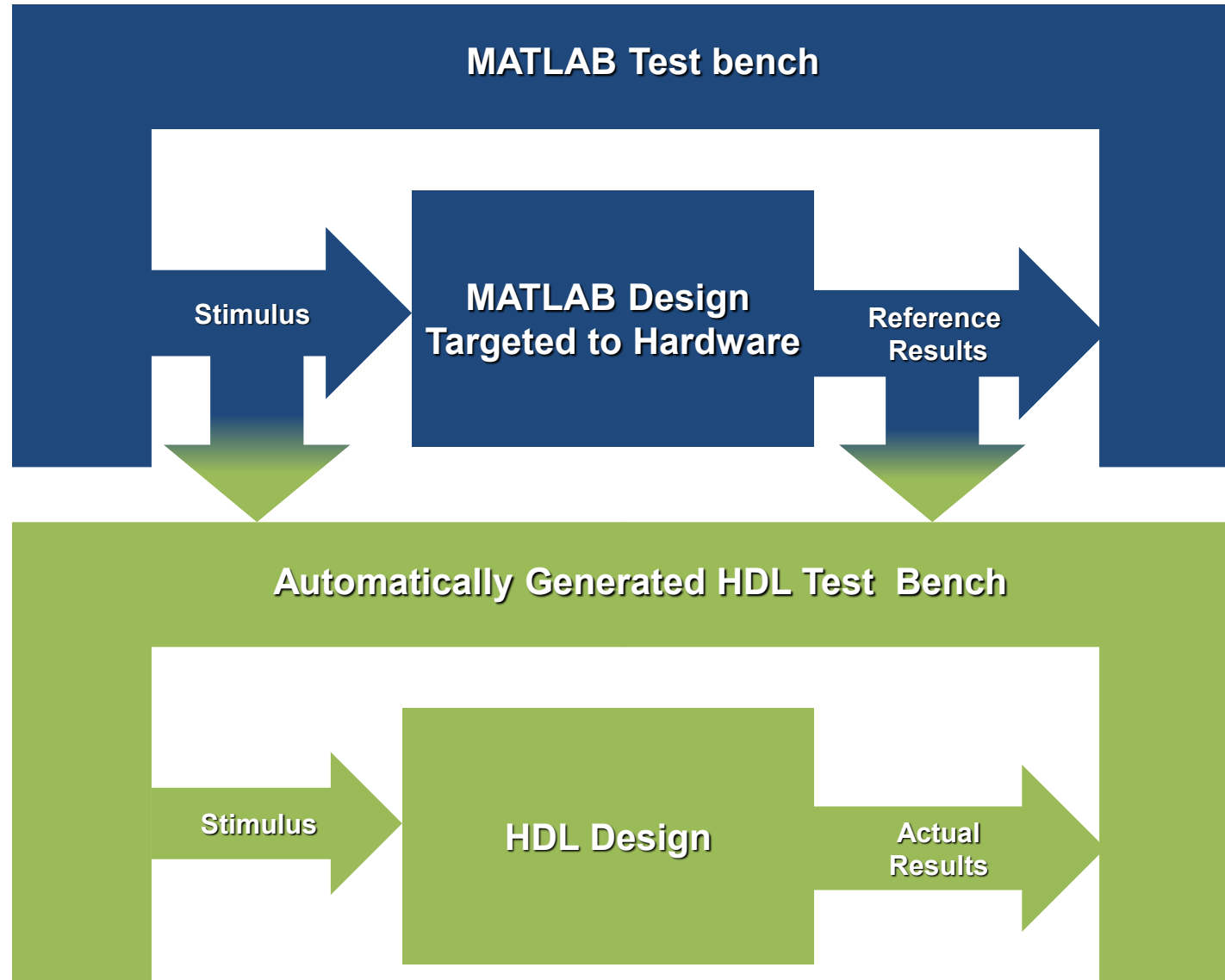
Otomatik Optimizasyon Seçenekleri



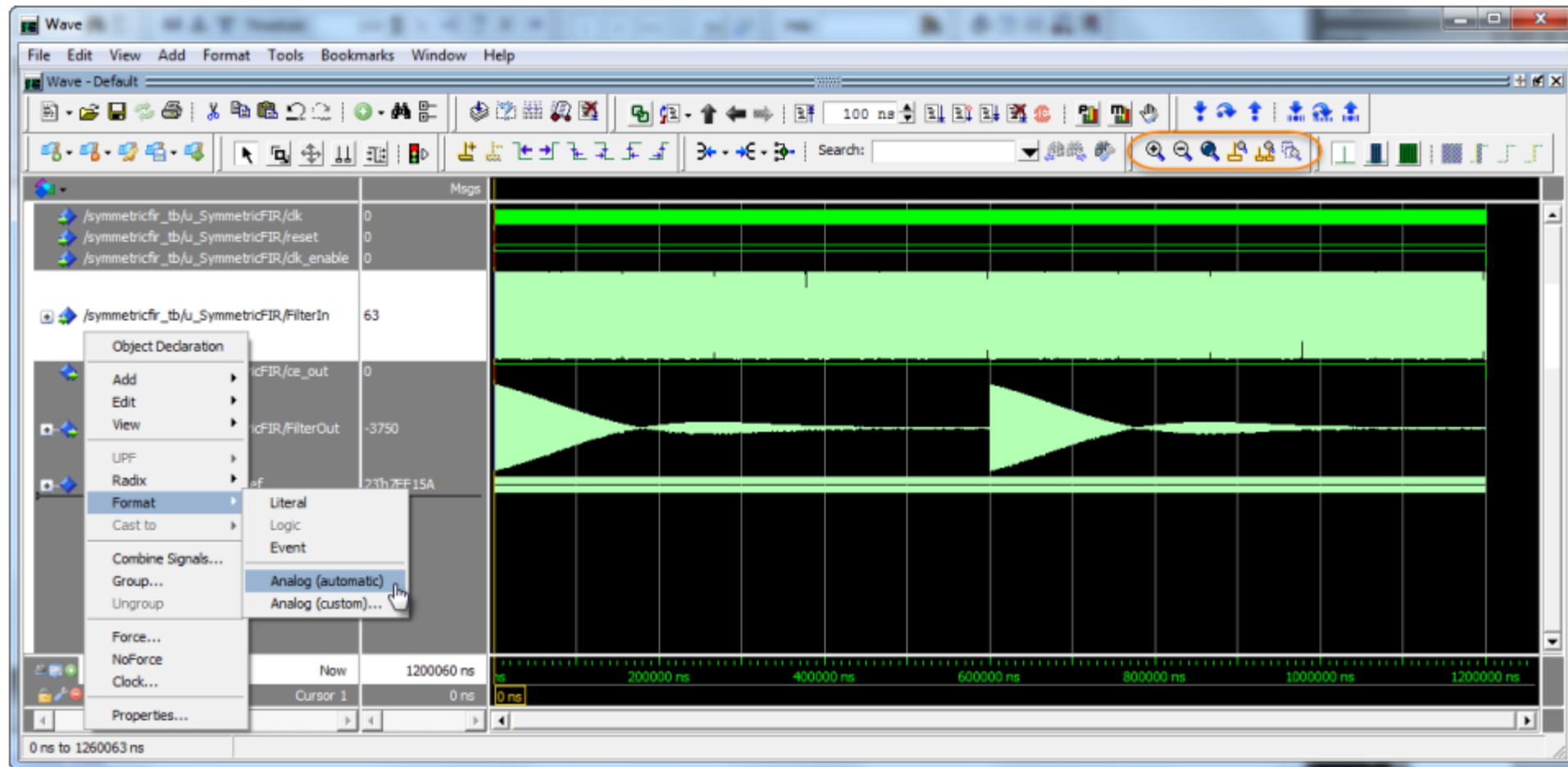
İteratif Tasarım Optimizasyonu



Testbench Tabanlı Doğrulama

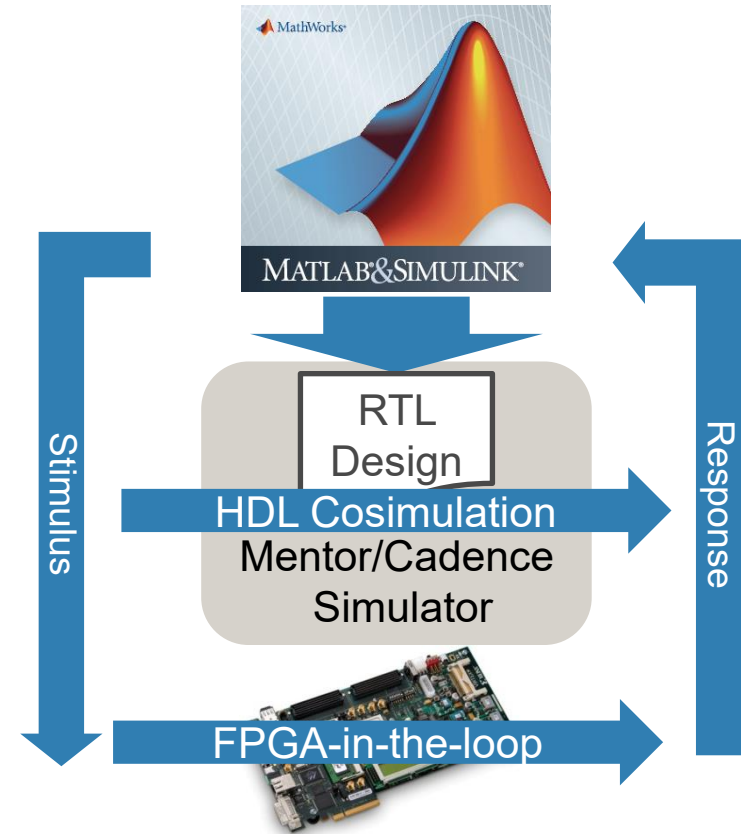


Testbench Tabanlı Doğrulama



HDL Verifier

- **Cosimulate** RTL back-to-back with the model to debug before FPGA deployment
- Simulate **FPGA-in-the-loop** with your MATLAB/Simulink tests





TEŞEKKÜRLER

