



GEBZE TEKNİK ÜNİVERSİTESİ
ELEKTRONİK MÜHENDİSLİĞİ

ELM 235

LOJİK DEVRE TASARIM LABORATUVARI

LAB 0x2 Deney Raporu

Lojik Devreler ve Tasarım Laboratuvarı

Hazırlayanlar
1) 1801022024 – M. Cemal Eryiğit
2) 1801022077 – Burak Kamil Çiftci

1. Giriş

Bu deney kapsamında; senkron tasarım ve asimetrik mantık devreleri hakkında gerekli araştırmalar yapılmıştır. Araştırmalar sonrasında deney için gerekli olan bilgiler sağlanmıştır.

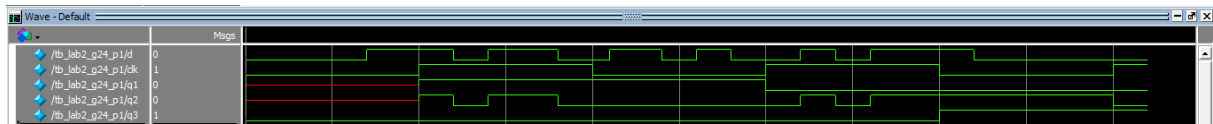
Problem 1- Hafıza elemanları karşılaştırması

1.1 Teorik Araştırma

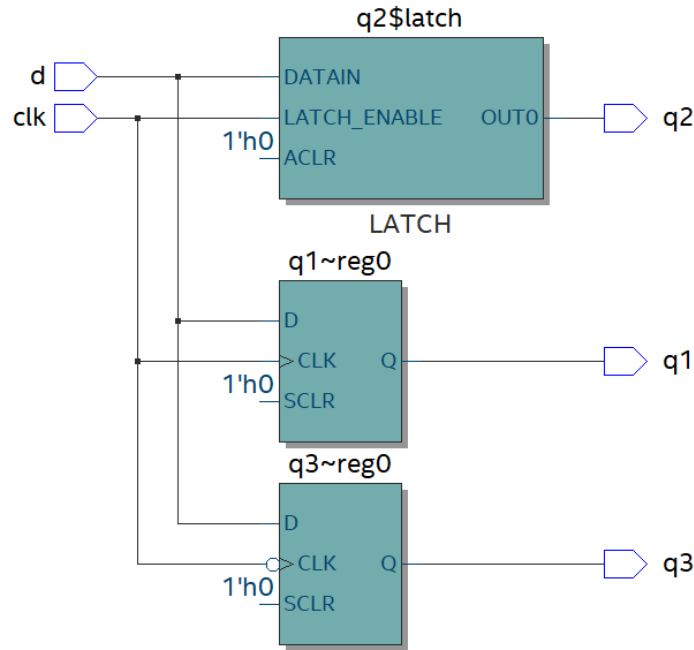
Bu problemin teorik araştırmasında lojik hafıza elemanları olan Latch, Rising-edge triggered Flip-Flop ve Falling-edge triggered Flip-Flop hakkında araştırmalar yapılmıştır ve bu elemanların systemverilog dili ile nasıl kullanılacağı öğrenilmiştir.

1.2 Deneyin Yapılışı

Problemde istenilen Latch, Rising-edge triggered Flip-Flop ve Falling-edge triggered Flip-Flop hafıza elemanları girişleri tek sinyale, çıkışları ise ayrı sinyallere bağlanmıştır.



Şekil 1 Dalga Görüntüsü



Şekil 2 RTL Devresi

Flow Summary		Analysis & Synthesis Resource Usage Summary		
<<Filter>>		<<Filter>>		
Flow Status	Successful - Tue Jun 29 12:41:14 2021	1	Estimated Total logic elements	3
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition	2		
Revision Name	lab2_g24_p1	3	Total combinational functions	1
Top-level Entity Name	lab2_g24_p1	4	Logic element usage by number of LUT inputs	
Family	MAX 10	1	-- 4 input functions	0
Device	10M08DAF484C8G	2	-- 3 input functions	1
Timing Models	Final	3	-- <=2 input functions	0
Total logic elements	4 / 8,064 (< 1 %)	5		
Total registers	2	6	Logic elements by mode	
Total pins	5 / 250 (2 %)	1	-- normal mode	1
Total virtual pins	0	2	-- arithmetic mode	0
Total memory bits	0 / 387,072 (0 %)	7		
Embedded Multiplier 9-bit elements	0 / 48 (0 %)	8	Total registers	2
Total PLLs	0 / 2 (0 %)	1	-- Dedicated logic registers	2
UFM blocks	0 / 1 (0 %)	2	-- I/O registers	0
ADC blocks	0 / 1 (0 %)	9		
		10	I/O pins	5
		11		
		12	Embedded Multiplier 9-bit elements	0
		13		
		14	Maximum fan-out node	d~input
		15	Maximum fan-out	3
		16	Total fan-out	15
		17	Average fan-out	1.15

Compilation Hierarchy Node	Combinational ALUTs	Dedicated Logic Registers	Memory Bits	UFM Blocks	DSP Elements	DSP 9x9	DSP 18x18	Pins	Virtual Pins	ADC blocks	Full Hierarchy Name	Entity Name	Library Name
lab2_g24_p1	1 (1)	2 (2)	0	0	0	0	0	5	0	0	lab2_g24_p1	lab2_g24_p1	work

Şekil 5 Analiz ve Kaynak Sentez kullanım Özetleri

```

/* Hazirlayanlar:
* M.Cemal Eryigit
* Burak Kamil Ciftci
* ELM235 2021 Bahar Lab2 - Problem 1
*/

module lab2_g24_p1(
input logic clk, d,
output logic q1,q2,q3
);

```

```
always_ff @(posedge clk)

q1 <= d;

always_latch

if(clk) q2 <= d;

always_ff @(negedge clk)

q3 <= d;

endmodule
```

```
/* Hazirlayanlar:

* M.Cemal Eryigit
* Burak Kamil Ciftci
* ELM235 2021 Bahar Lab2 - Problem 1
*/

`timescale 1ns/1ps

module tb_lab2_g24_p1 ();

logic d;

logic clk;

logic q1, q2, q3;

lab2_g24_p1 uut0(.q1(q1),.q2(q2),.q3(q3),.d(d),.clk(clk));

always

begin

clk = 0; #10;

clk = 1; #10;

end

initial begin

d = 0; #7; d = 1; #5; d = 0; #2;

d = 1; #4; d = 0; #3; d = 1; #3;

d = 0; #2; d = 1; #2; d = 0; #4;
```

```
d = 1; #2; d = 0; #2; d = 1; #6;  
  
d = 0; #10;  
  
$stop;  
  
end  
  
endmodule
```

1.3 Deneyin Yorumu

RTL Devresinde görüldüğü üzere devrede giriş aynı sinyale bağlanmıştır, çıkış ise ayrı ayrı sinyallere bağlanmıştır. Latch, Rising-edge triggered Flip-Flop ve Falling-edge triggered Flip-Flop arasındaki simülasyon sonuçları farklı dalga görüntüsünde görüldü. Bu simülasyon farkı hafıza devreleri arasındaki yapı farkları sonucunda oluşmuştur. Latchlerde d'nin q'ya bağlı değişiminin belli bir süreli sinyal çizgisinde gerçekleşirken, flip-flop'larda bu değişimin rising veya falling edge'lerde değiştiğini görüyoruz.

Problem – 2 ALU Tasarımı

2.1 Teorik Araştırma

Bu problem için teorik bir araştırma bulunmamaktadır.

2.2 Deneyin Yapılışı

A) 32-bitlik NZVC destekli ALU tasarlayın. Tasarlarken always_comb bloğu ve if statement lar kullanabilirsiniz. ALU nun opcode ları ve gerçeklemeniz gereken fonksiyonları Tablo 1 de verilmiştir.

B) Latch oluşmaması için always_comb bloğu içinde kullandığınız bütün atamaların her bir dalda olduğuna emin olun. Örnek olarak if (...) a = 4; gibi bir atamada if bloğu içine girmezse kod a ya atama yapılmadığı için latch oluşacaktır.

C) Flagların hangi koşullarda olduğunu iyi anlayın, mesela negative flag sonucun sadece 31. bitine bakacak, zero flag sonucun bütün bitleri 0 ise yanacak, carry sadece toplama, çıkarma ve shift operasyonlarında oluşabilir, vs. Shift ile alakalı Carry oluşturma şemaları EK B de verilmiştir.

D) Devrenizi basit birkaç girişle test edin ve doğruluğunu gözlemleyin.

op ²	operasyon	açıklama
000	addition	A + B
001	subtraction	A - B
100	shift left logical	A sinyalini B kadar sola kaydır ¹
010	xor	A XOR B
101	shift right logical	A sinyalini B kadar sağa kaydır ¹ (0 padded)
110	shift right arithmetic	A sinyalini B kadar sağa kaydır ¹ (sign padded)
011	or	A OR B
111	and	A AND B

Tablo 1

```

module lab2_g24_p2 (
input logic [31:0] a, b, [ 2:0] op,
output logic [31:0] out, n, z, v, c,hata
);
always_comb begin
assign n = (out[31]) ? 1:0;
assign z = (out == 32'b0) ? 1:0;
assign c = {~c, out} == (a + b);
assign v =( ~(a[31]^b[31])^out[31]^c) ? 0:1;
case(op)
3'b000:
out = a+b;
3'b001:
out = a-b;
3'b100:
out = a >> b;
3'b010:
out = a ^ b;
3'b101:
out = a << b;
3'b110:
out= a >>> b;
3'b011:
out = a | b;
3'b111:
out = a & b;
default:
hata=out?0:1;
endcase
end
endmodule

```


[illegible]


```

        if ({n, z, v, c} != {nexp, zexp, vexp, cexp}) begin

            flagerr++; // carry hatalarini say

            $display("nzvc icin yanlis deger, op: %b, beklenen nzvc:
            %b,%b,%b,%b, devre: %b,%b,%b,%b", op, nexp, zexp, vexp, cexp, n, z, v, c);

            $display("|--a: %b, b: %b, s: %b", a, b, s);

        end

        ntest++;

    end

    $display("Simulasyon tamamlandi. %d farkli testte, toplamda %d
    sonuc hatasi, %d flag hatasi mevcut", ntest, serr, flagerr);

end

else

    $display("dosyayi acamadik");

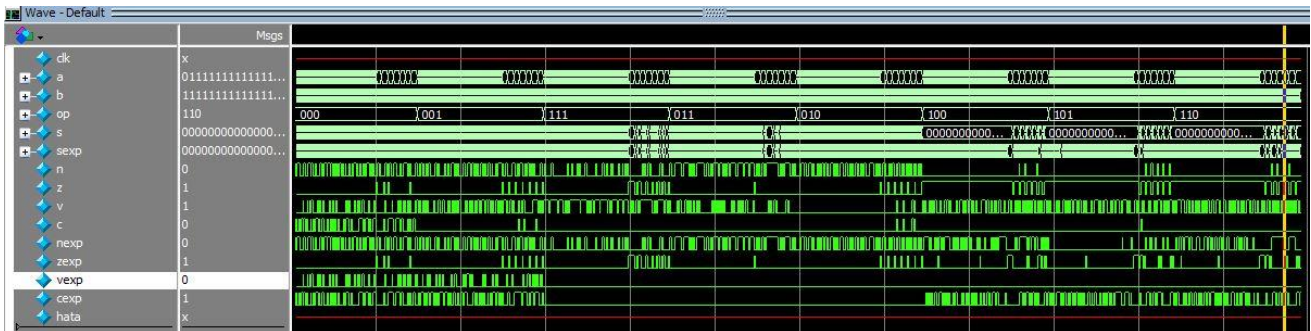
    $fclose(fd);

    $stop;

end

endmodule


```



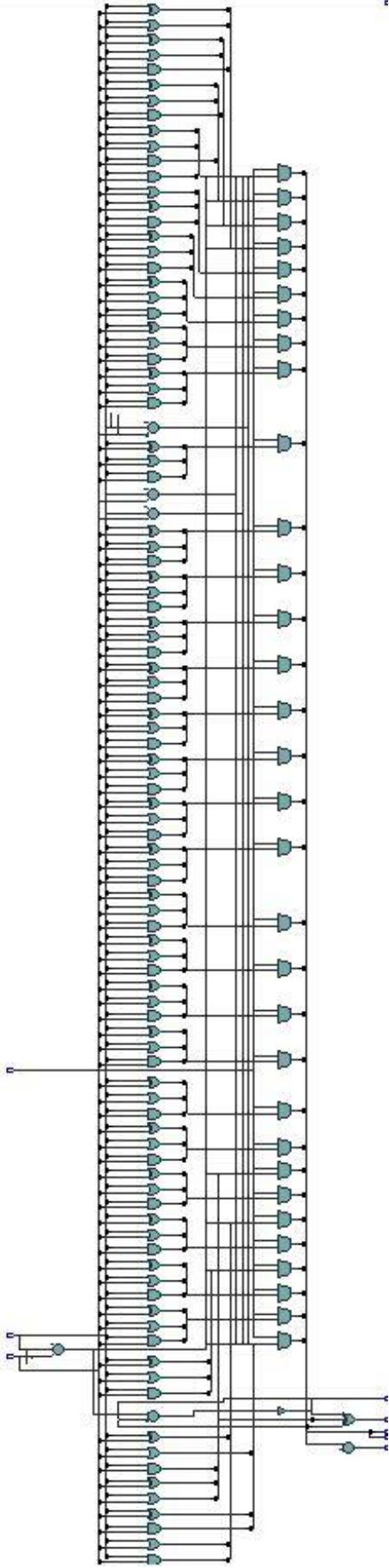
Şekil 7 dalga görüntüsü

Flow Summary	
<<Filter>>	
Flow Status	Successful - Tue Jun 29 19:21:52 2021
Quartus Prime Version	20.1.1 Build 720 11/11/2020 SJ Lite Edition
Revision Name	lab2_g24_p2
Top-level Entity Name	lab2_g24_p2
Family	MAX 10
Device	10M50DAF484C6GES
Timing Models	Preliminary
Total logic elements	572 / 49,760 (1 %)
Total registers	0
Total pins	104 / 360 (29 %)
Total virtual pins	0
Total memory bits	0 / 1,677,312 (0 %)
Embedded Multiplier 9-bit elements	0 / 288 (0 %)
Total PLLs	0 / 4 (0 %)
UFM blocks	0 / 1 (0 %)
ADC blocks	0 / 2 (0 %)

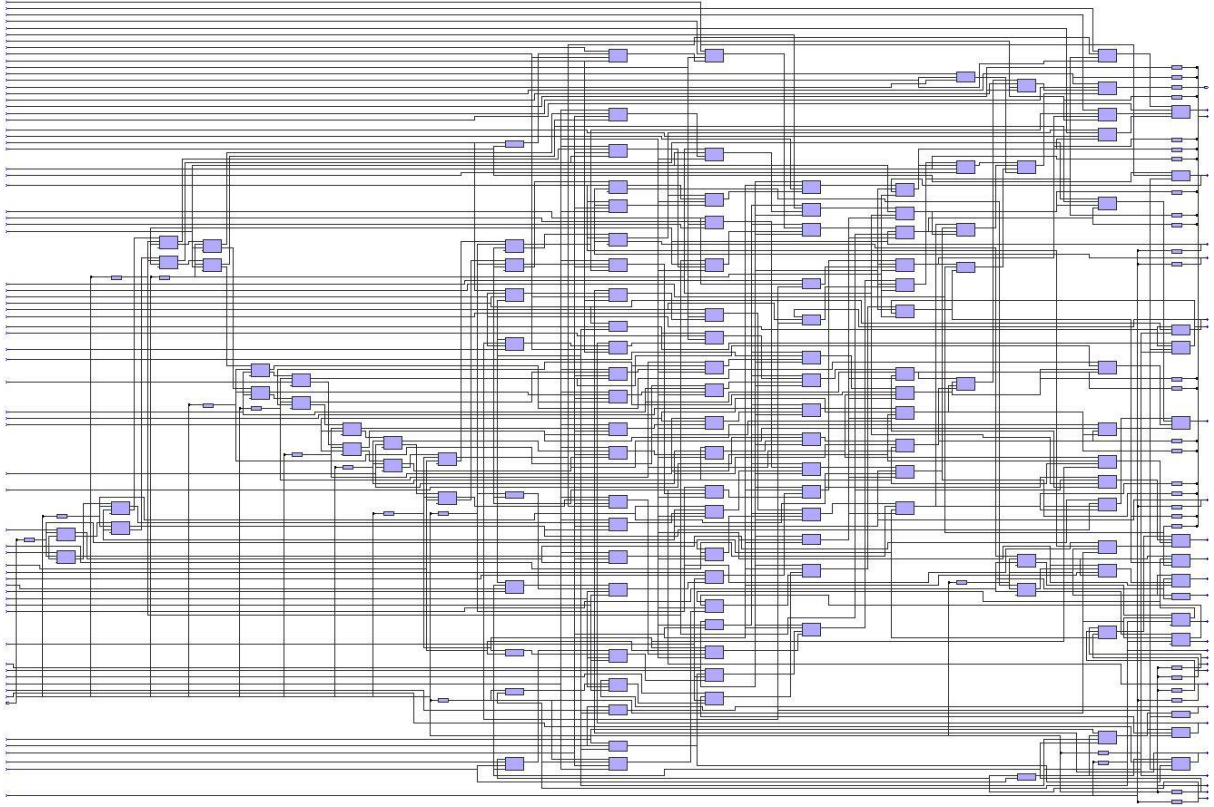
Analysis & Synthesis Resource Usage Summary		
<<Filter>>		
	Resource	Usage
1	Estimated Total logic elements	571
2		
3	Total combinational functions	571
4	Logic element usage by number of LUT inputs	
1	-- 4 input functions	302
2	-- 3 input functions	233
3	-- <=2 input functions	36
5		
6	Logic elements by mode	
1	-- normal mode	508
2	-- arithmetic mode	63
7		
8	Total registers	0
1	-- Dedicated logic registers	0
2	-- I/O registers	0
9		
10	I/O pins	104
11		
12	Embedded Multiplier 9-bit elements	0
13		
14	Maximum fan-out node	b[1...put
15	Maximum fan-out	88
16	Total fan-out	2117
17	Average fan-out	2.72

Analysis & Synthesis Resource Utilization by Entity														
Table of Contents														

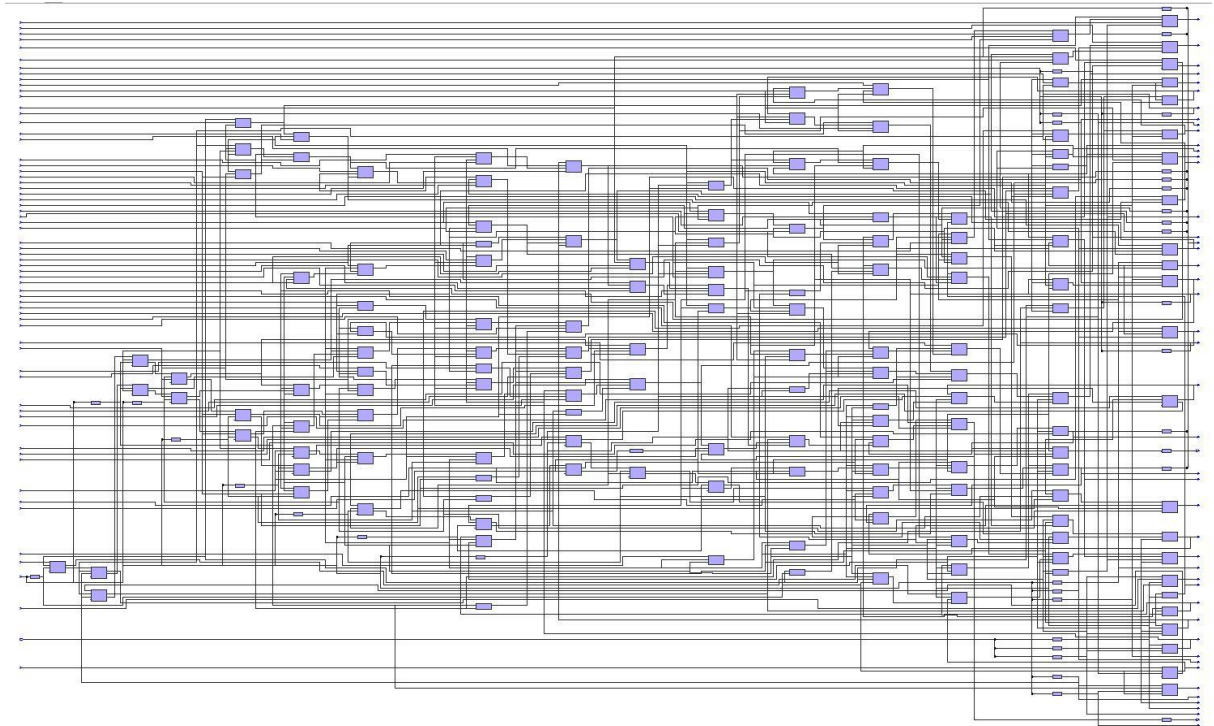
Şekil 8 Analiz ve kaynak sentez kullanım özetleri



Şekil 9 RTL devre şeması



Şekil 10 Eşleştirme Ardı Devre Şeması



Şekil 11 Fitting Technology Map Viewer

2.3 Deneyin Yorumu

Basit şekilde yazılan testbench dosyasında 9 adet farklı giriş çıkış kombinasyonu denendi. 2 adet 32 bitlik ve 1 adet 3 bitlik girişimiz olduğu için her n v z c bayrağını kullanabilmek için özel 32 bit değerler seçilmeye özen gösterildi. Verilen testbench dosyasında ise giriş çıkış için fazla miktarda kombinasyon vardı. Bu giriş çıkış değerleri txt formatında bir dosyadaydı. Bu dosya dosya okuma fonksiyonları kullanılarak testbenchde giriş çıkış olarak kullanıldı. Bunun bize sağladığı daha çok veri olduğu için neredeyse tüm hataları simülasyonda görmemize olanak sağladı. Şekil 6 ve Şekil 7' de belirtilen simülasyon sonuçları arasında belirli farklar gözlenmektedir. Bu farkların sebebi iki testbench arasındaki kod farkları ve kontrol mekanizması olmaktadır.