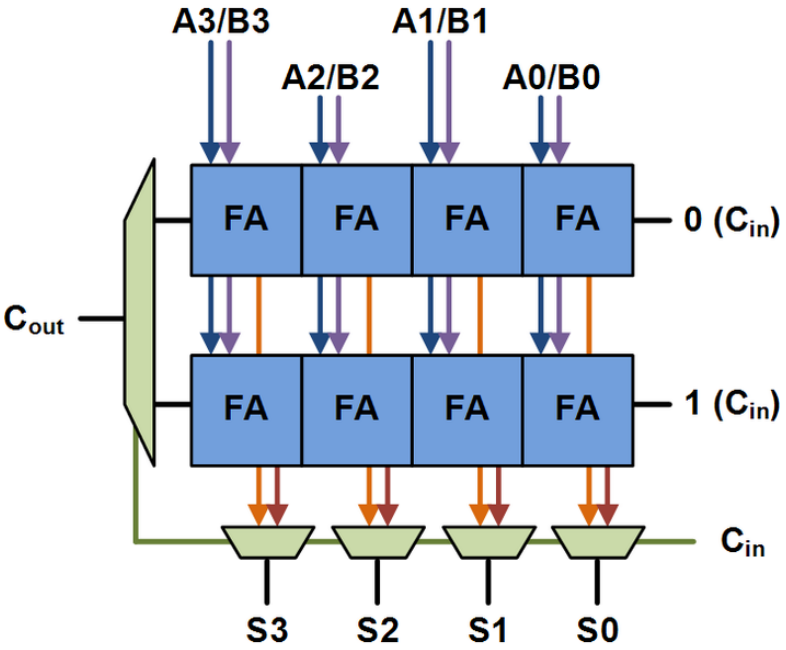
BIL 264 2022-2023 GÜZ DÖNEMİ PROJESİ

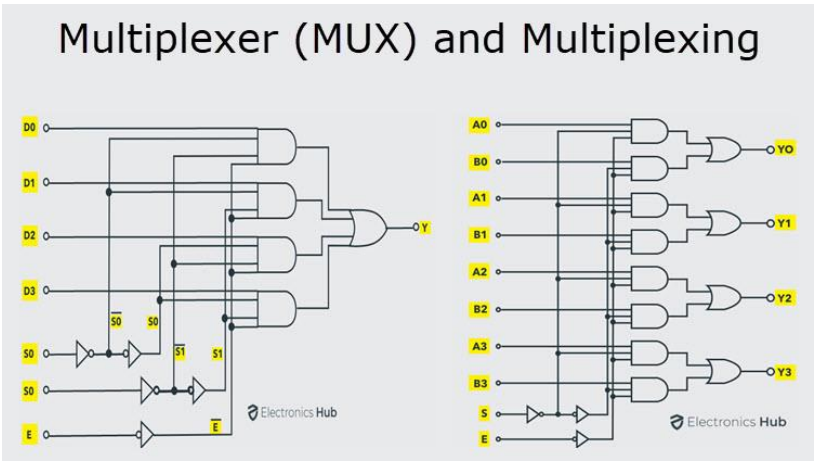
Grup 41

İsim	Soyisim	Numara	Bölüm
Ahmet Enes	Seyhun	171201018	Elektrik Elektronik
İrem	Davulcu	181201058	Elektrik Elektronik

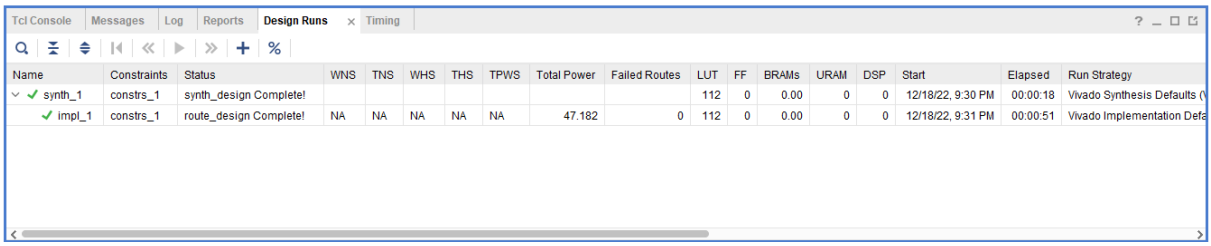
1. CARRY SELECT ADDER



Bu toplayıcının temel mantığı carry_in girişi 1 ve 0 olduğu zaman farklı toplama işlemlerinin sonucunu hesaplayıp çıkışı seçim girişimize göre vermektir. Yukarıdaki fotoğrafta 4 bitlik bir carry select adder yapısını görmekteyiz. Projemizde 64 bitlik tasarımı gerçekleştirirken 4 bitlik carry select adder yapısından faydalandık. Bunu yaparken full adder ve dolayısıyla half adderlere ihtiyaç duyduk ve kodunu yazarak projemizde çağırarak kullandık. Seçim mantığına dayalı bir toplayıcı olduğundan ötürü mimarisinde mux görmekteyiz. Mux iç yapısı aşağıdaki fotoğrafta gösterilmiştir:

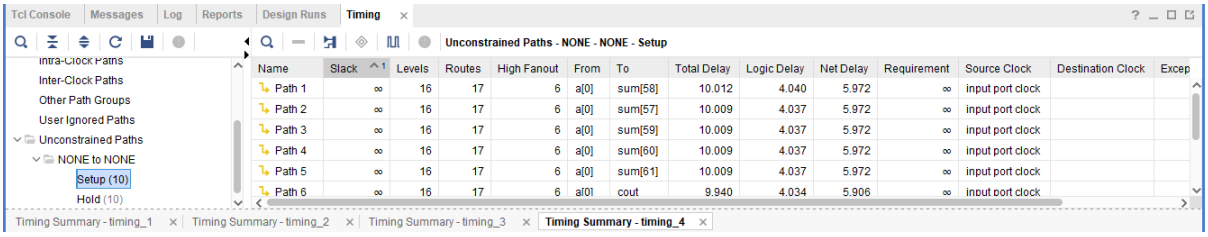


Carry Select Adder Static Güç tüketimi ve toplam Lut Sayısı:



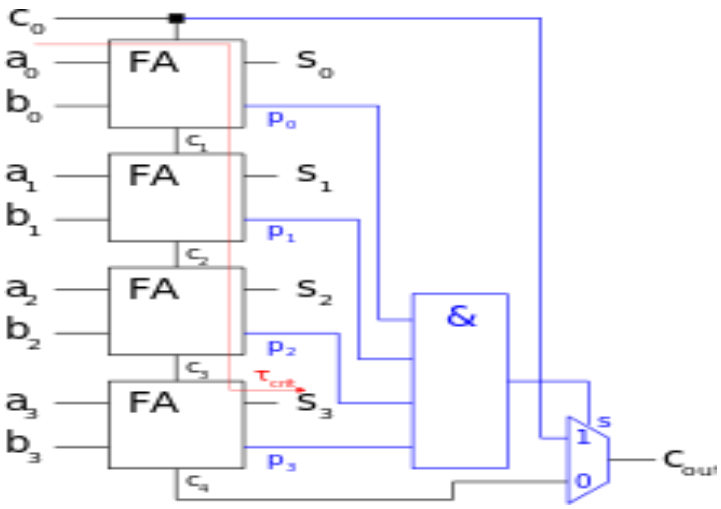
Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	synth_design Complete!								112	0	0.00	0	0	12/18/22, 9:30 PM	00:00:18	Vivado Synthesis Defaults
impl_1	constrs_1	route_design Complete!	NA	NA	NA	NA	NA	47.182	0	112	0	0.00	0	0	12/18/22, 9:31 PM	00:00:51	Vivado Implementation Defa

Carry Select Adder Zaman Analizi:



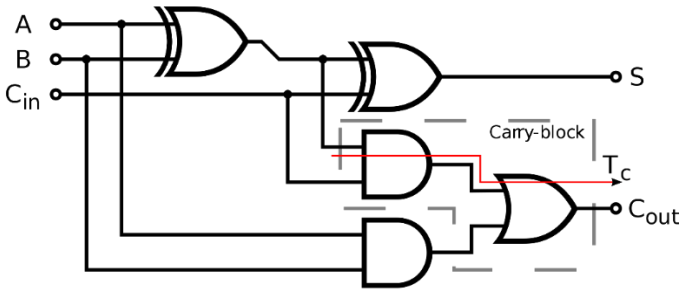
Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Excep
Path 1	∞	16	17	6	a[0]	sum[58]	10.012	4.040	5.972	∞	input port clock		
Path 2	∞	16	17	6	a[0]	sum[57]	10.009	4.037	5.972	∞	input port clock		
Path 3	∞	16	17	6	a[0]	sum[59]	10.009	4.037	5.972	∞	input port clock		
Path 4	∞	16	17	6	a[0]	sum[60]	10.009	4.037	5.972	∞	input port clock		
Path 5	∞	16	17	6	a[0]	sum[61]	10.009	4.037	5.972	∞	input port clock		
Path 6	∞	16	17	6	a[0]	cout	9.940	4.034	5.906	∞	input port clock		

2. CARRY SKIP ADDER

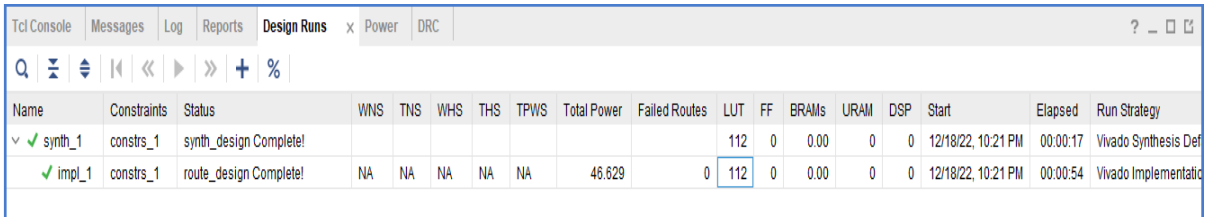


Yukarıda 4 bitlik bir carry skip adder iç yapısı gösterilmiştir. Girişler verildikten sonra full adderlarda carry outlar taşınıp son c4 hesabını yapar. Bununla full adderların sum çıkışından elde edilen değerin propagete işlemiyle hesaplanmasıyla çıktıyı 2x1 muxa sokup çıkışı taşıyarak elde ediliyor. Yani 4bit carry skip adderda ripple carry adder mux ve propegate işlemi yapılmaktadır. Bu şekilde olan 4bitlik carry skip adderlardan 16 tane çağırarak 64bit carry skip adder elde ettim.

Full adder iç yapısı aşağıda gösterilmiştir :



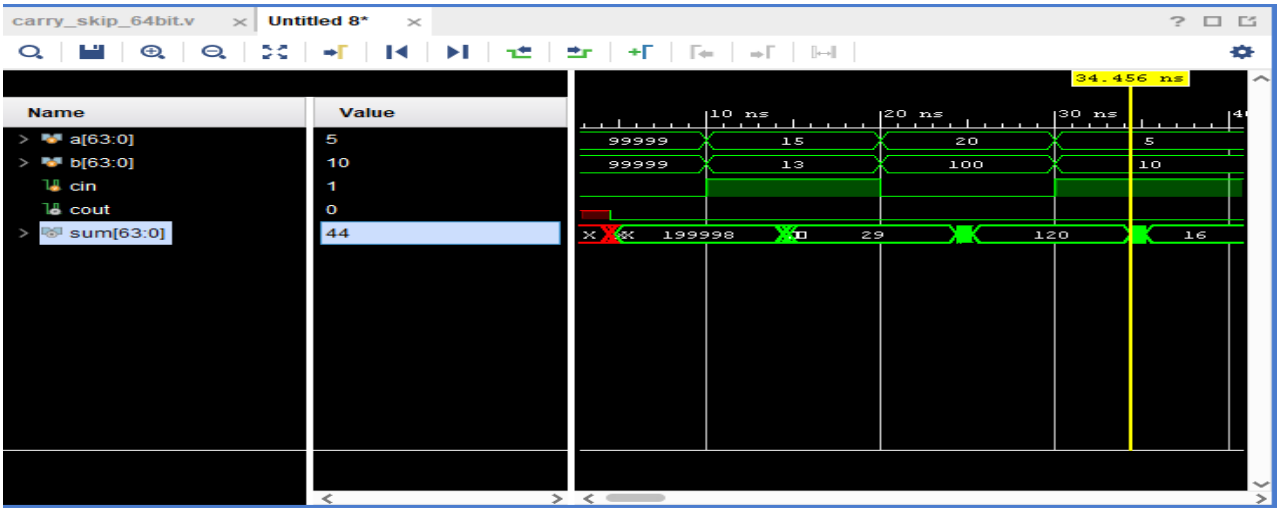
64 bit Carry Skip Adder Toplam Güç tüketimi ve Lut Sayısı:



Tcl Console | Messages | Log | Reports | Design Runs | Power | DRC

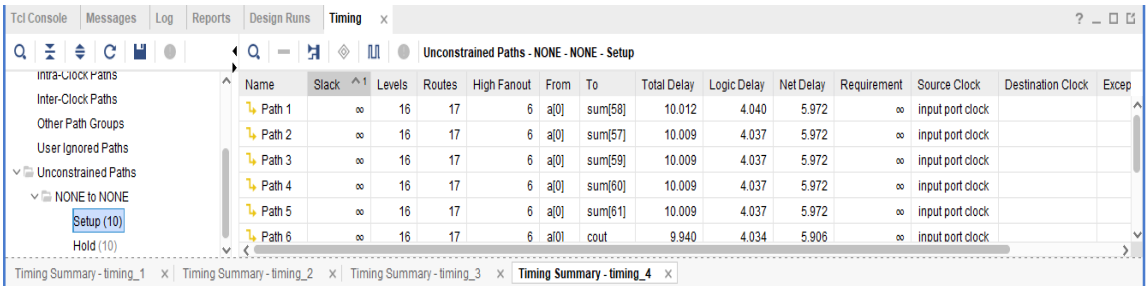
Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	synth_design Complete!								112	0	0.00	0	0	12/18/22, 10:21 PM	00:00:17	Vivado Synthesis Def
impl_1	constrs_1	route_design Complete!	NA	NA	NA	NA	NA	46.629	0	112	0	0.00	0	0	12/18/22, 10:21 PM	00:00:54	Vivado Implementatio

Sentez Sonrası Zaman Sentezi:

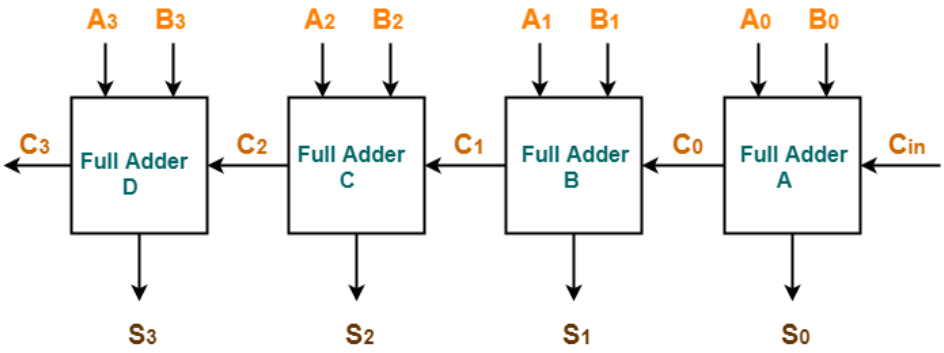


Sentez sonrası zaman sentezi yapma nedenim tasarımları kapı seviyesi yapıp flip flop kullanmadığımdan ve bazı toplama methodlarında aynı delay çıkmasıdır. Bu yöntemle çıktıların ne kadar delayla gerçeklediğini gözlemleme fırsatım oldu. Örnek olarak 20 120 toplamı ve cin 0 girişi durumunda 120 sonucu14.456 ns delayla çıkmıştır.

64 bit Carry Skip Adder Static Zaman Analizi:



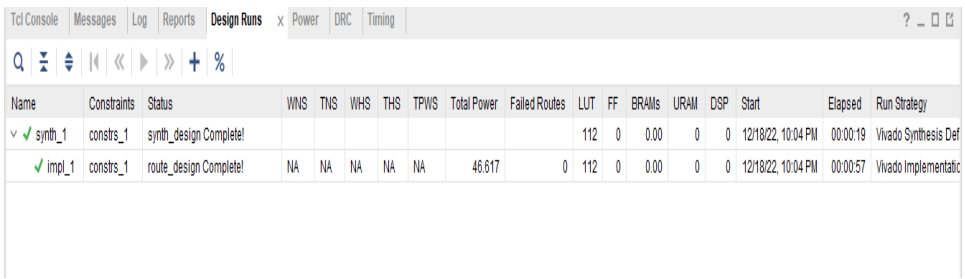
3. RIPPLE CARRY ADDER



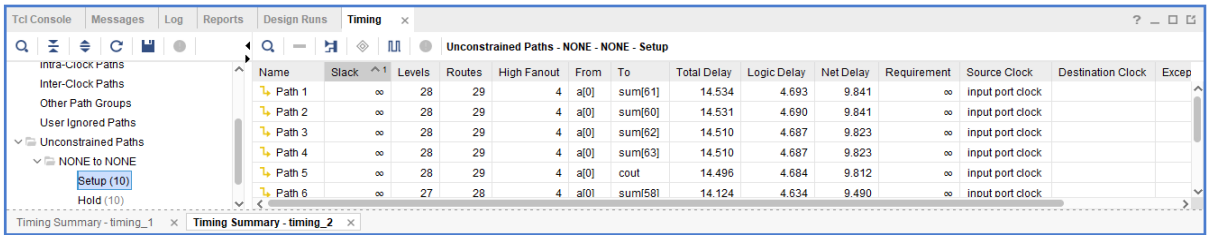
4-bit Ripple Carry Adder

Yukarıda 4 bitlik bir ripple carry adder iç yapısı gösterilmiştir. Projemizde yazdığımız toplayıcılarda kullanarak işlemi kolaylaştırmış olduk. Bu toplayıcının mantığı toplama işlemini gerçekleştirdiğimiz ilk full adder eldesini sıradaki full adder girişine bağlayarak bu şekilde devamında da eldeleri sonraki toplayıcıya (full adder) aktararak sonucu bulmaktır.

64 bit Ripple Carry Adder Toplam Güç tüketimi ve Lut Sayısı:

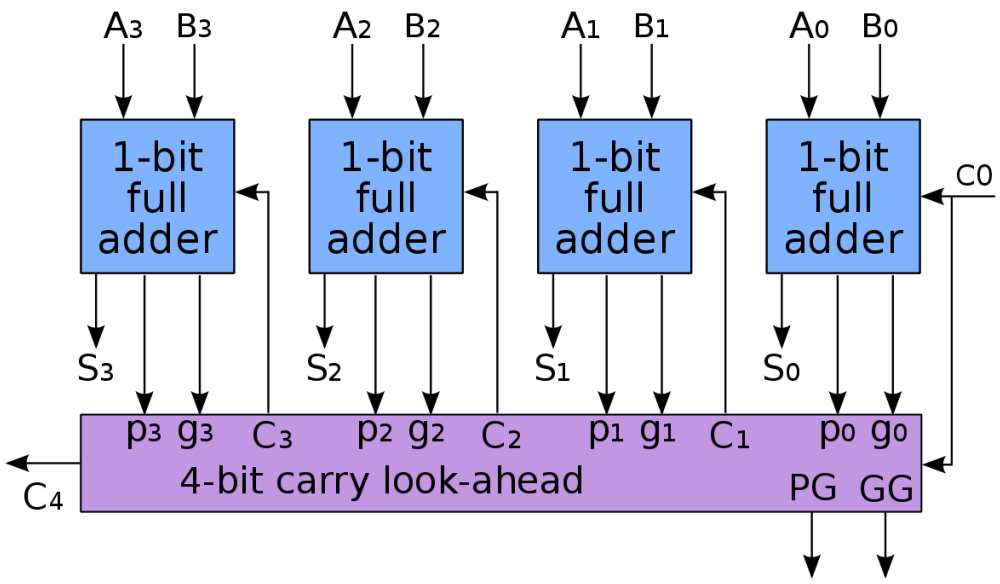


64 bit Ripple Carry Adder Zaman Analizi:



Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Excep
Path 1	∞	28	29	4	a[0]	sum[61]	14.534	4.693	9.841	∞	input port clock		
Path 2	∞	28	29	4	a[0]	sum[60]	14.531	4.690	9.841	∞	input port clock		
Path 3	∞	28	29	4	a[0]	sum[62]	14.510	4.687	9.823	∞	input port clock		
Path 4	∞	28	29	4	a[0]	sum[63]	14.510	4.687	9.823	∞	input port clock		
Path 5	∞	28	29	4	a[0]	cout	14.496	4.684	9.812	∞	input port clock		
Path 6	∞	27	28	4	a[0]	sum[58]	14.124	4.634	9.490	∞	input port clock		

4. CARRY LOOK AHEAD ADDER

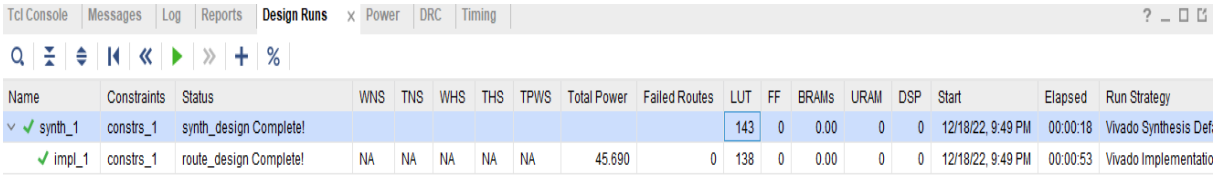


Yukarıda 4 bitlik bir carry look ahead toplayıcının iç yapısı gösterilmiştir. Bu toplayıcıda yayılma gecikmesi azaltılır. Herhangi bir aşamadaki taşıma çıkışı, yalnızca başlangıç aşamasının ilk taşıma bitine bağlıdır. Bu toplayıcıyı kullanarak ara sonuçları hesaplamak mümkündür. Bu toplayıcı, hesaplama için kullanılan en hızlı toplayıcıdır.

Taşıma bitleri C1, C2, C3 ve C4 şu şekilde hesaplanabilir:

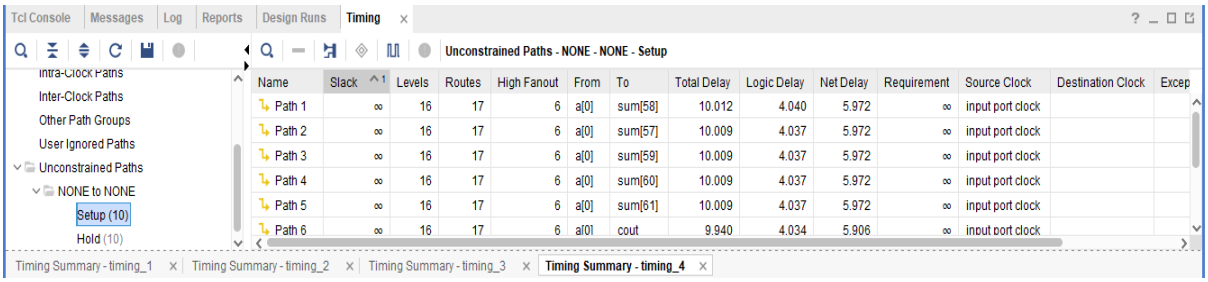
- C1 = C0.P0 + G0.
- C2 = C1.P1 + G1 = (C0.P0 + G0) .P1 + G1.
- C3 = C2.P2 + G2 = (C1.P1 + G1) .P2 + G2.
- C4 = C3.P3 + G3 = C0.P0.P1.P2.P3 + P3.P2.P1.G0 + P3.P2.G1 + G2.P3 + G3.

64 bit Carry Look Ahead Toplam Güç tüketimi ve Lut Sayısı:



Name	Constraints	Status	WNS	TNS	WHS	THS	TPWS	Total Power	Failed Routes	LUT	FF	BRAMs	URAM	DSP	Start	Elapsed	Run Strategy
synth_1	constrs_1	synth_design Complete!								143	0	0.00	0	0	12/18/22, 9:49 PM	00:00:18	Vivado Synthesis Def
impl_1	constrs_1	route_design Complete!	NA	NA	NA	NA	NA	45.690	0	138	0	0.00	0	0	12/18/22, 9:49 PM	00:00:53	Vivado Implementatio

64 bit Carry Look Ahead Zaman Analizi:



4 Farklı Toplama Yönteminin Kıyaslaması:

Genel değerlendirme yaparsak CLA, RCA'ye kıyasla daha iyidir. Ripple Carry bu 4 method içinde en çok güç tüketen ve gecikme süresinin fazla oluşundan dolayı en kötü olan methoddur. Statik zaman analizi yaptığımda Carry skip adder ile carry select adder aynı çıktı ama sentez sonrası zaman analizinden gördüğüm kadarıyla ve litaratür taramasından öğrendiğim bilgilerle Carry Select Adder düşük güç tüketimi ve düşük gecikme süresiyle diğer methodlardan daha iyidir.

PROJEDEKİ GÖREV DAĞILIMI:

Ahmet Enes Seyhun: UART kodunu yazdı. Carry look ahead toplayıcı ve bu toplayıcıda kullanılan diğer toplayıcıların da kodunu yazdı. Carry Skip adder64 bit adderı yazdı. Test benchleri yazarak zaman analizi ve statik analizi yaptı.

İrem Davulcu: Carry select ve ripple carry adder kodunu yazdı, toplayıcılarda kullanılan temel birimleri hazırladı (full adder,half adder) ve kod simülasyon ve test görevini üstlendi.

**Gruptaki diğer 2 arkadaşımız hiçbir görevde görev almadı ve mesajlara dönmediler. Son gün dahi proje hakkında bir şey söylemediler dersi veya projeyi bıraktıklarını düşünüyoruz.