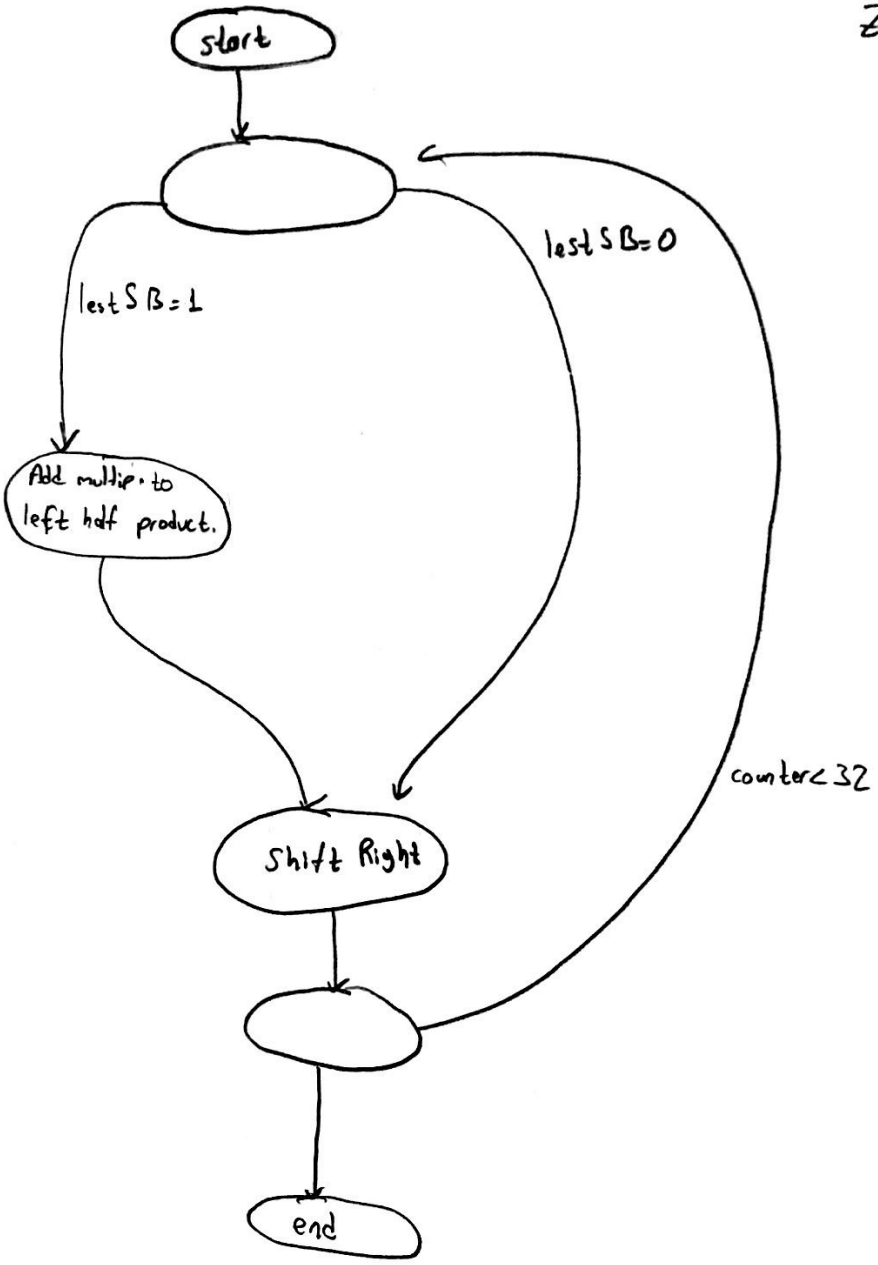
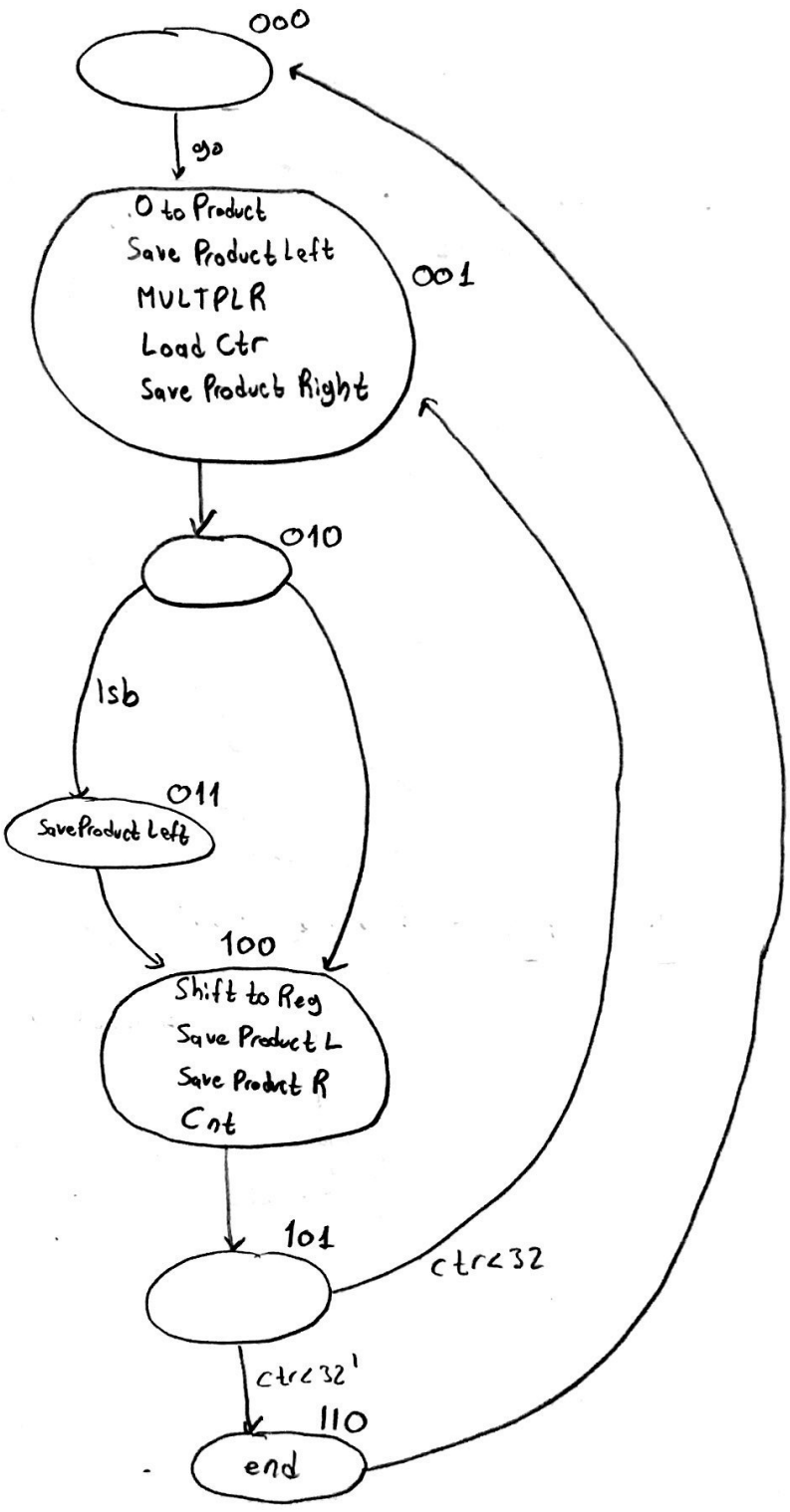
Code Flow Chart:

161044063
Zafer ALTAY



* Eğer elimizdeki sayının lsb'si 1 ise multiplicand', product'ın ilk 32 bitine ekleyip tekrar product'ın ilk 32 bitine yazar. Ardından sonraki clock'ta sağa shift eder. Lsb 0 ise direkt sağa shift eder. 32 kere çalıştır.

FSM



CU Input						CU Output		
S_2	S_1	S_0	go	lsb	ctr<32	n_2	n_1	n_0
0	0	0	0	x	x	0	0	0
0	0	0	1	x	x	0	0	1
0	0	1	x	x	x	0	1	0
0	1	0	x	0	x	1	0	0
0	1	0	x	1	x	0	1	1
0	1	1	x	x	x	1	0	0
1	0	1	x	x	0	1	1	0
1	0	1	x	x	1	0	1	0
1	1	0	x	x	x	0	0	0

$$n_2 = S_2' S_1 S_0' lsb' + S_2' S_1 S_0 + S_2 S_1' S_0' + S_2 S_1' S_0 ctr32'$$

$$S_2' S_1 (S_0' lsb' + S_0) \qquad S_2 S_1' (S_0' + S_0 ctr32')$$

$$= S_2' S_1 (lsb' + S_0) + S_2 S_1' (S_0' + ctr32')$$

$$n_1 = S_2' S_1' S_0 + S_2' S_1 S_0' lsb + S_2 S_1' S_0 ctr32 + S_2 S_1' S_0 ctr32'$$

$$S_2 S_1' S_0$$

$$= S_1' S_0 + S_2' S_1 S_0' lsb$$

$$n_0 = S_2' S_1' S_0' go + S_2' S_1 S_0' lsb + S_2 S_1' S_0'$$

$$= S_1' S_0' (S_2' go + S_2) + S_2' S_1 S_0' lsb$$

S_2 S_1 S_0	0 to Product	Load Ctr	Save Prod L	Save Prod R	MULTPLR	Shift to Reg	CNT
0 0 1	1	1	1	1	1	0	0
0 1 1	0	0	1	0	0	0	0
1 0 0	0	0	1	1	0	1	1

0 to Product = $S_2'S_1'S_0$

Load Ctr = $S_2'S_1'S_0$

MULTPLR = $S_2'S_1'S_0$

CNT = $S_2S_1'S_0'$

Shift to Reg = $S_2S_1'S_0'$

Save Prod R = $S_2'S_1'S_0 + S_2S_1'S_0' = S_1'(S_2'S_0 + S_2S_0') = \underline{S_1'(S_2 \text{ XOR } S_0)}$

Save Prod L = $S_2'S_1'S_0 + S_2'S_1S_0 + S_2S_1'S_0'$

$S_2'S_0 + S_2S_1'S_0'$

* Datapath içinde 2 tane 32 bit register kullandım. İsimleri registerLeft ve Register right olan bu registerler 64 bit product registeri temsil etmektedir.

→ Register left, ilk 32 biti temsil eder. Register left inputuna gelen 32 bitlik sayı 2 tane multiplexer ile kontrol edilerek geliyor.

→ ilk multiplexer inputun tamamen 0'lardan oluşan 32 bitlik input mu yoksa input'un product registerden gelen ilk 32 bitle multiplicantın toplamı mı olduğuna karar verir. Sadece program ilk çalıştığında 32 bitlik 0' inputunu seçer. Bu seçimi "0 toProduct" inputuyla yapar. Bu input 0' oldukça multiplexer her zaman 32 bit adder'ın outputunu seçecektir.

→ İkinci multiplexer ise ilk multiplexer'ın seçtiğini veya shifter'ın sonucunu seçerek product Register'ın ilk 32 bitine yollar. Yani register Left'e yollar. Bu seçimi "shift to Reg" inputu ile yapar. 0 olursa ilk multiplexer'ın seçimi 1 olursa shifter'ın sonucunu yazar.

→ Register Right için de yani son 32 bit için de input için 2 adet multiplexer kullandım.

→ İlk multiplexer, multiplier (programın ilk çalıştığı durum) ile register Right (ilk çalıştığı an hariç sürekli seçilecek durum) outputu arasında seçim yapar. Bunu "MULTPLR" inputuyla seçer.

→ İkinci multiplexer ise ilk multiplexer'ın sonucu ile shifter'ın son 32 biti arasında seçim yapar. Bunu "shift to Reg" ile seçer (diğer multiplexer da olduğu gibi).

- Toplamak gerekirse, devre çalışmaya başlayınca ilk durum olarak product registerların ilk 32 bitine 0, son 32 bitine multiplier yazılır. Ardından ise durumu göre shifter sonucu veya ilk 32 bitin multiplicantla toplanmış hali yazılır. (Bu işlem yapılınca son 32 bit tekrar eski sonucunu alır yani değişmez) Bu işlemlerin kontrolü control unit ile yapılır. Control Unit çizdiğim fsm diyagramına göre tasarlanmıştır.

- Datapath'de ki diğer elemanlardan 32 bit adder, 1 bit adderla oluşturduğum 32 tane elemanın birleşmesiyle oluşur. Sadece pozitif sayıları toplayacağımız için b'nin xor'unu almamıza gerek kalmadı. Multiplicand'ın productLeft ile toplanması sonucu elde edilen carryOut değeri bir 1 bitlik register'a kaydedilir. Çünkü eğer kaydedilmezse shift işleminin yapıldığı fsm adımına geçince, carry out shift edilecek sayının 32 bitiyle, multiplicand'ın toplamının carry'si olur. Bu yüzden register ile önceki işlemin carry sonucunu saklayıp bu sorunu ortadan kaldırdım.

- 64 bit shifter'ı kendim tasarlamam çünkü her seferinde sadece 1 bit shift edeceğimiz için devrenin kendi shifter'ından çok daha hesaplı ve 64 bitlik tasarladığım için de kullanımı çok daha kolay oldu.

- DataPath'de bir counter tutup her döngü de 1 artırıp 32 ile kıyasladım. 32 den küçük olduğu verdiği output'a göre controlUnit'e yeni durumlar belirlettim.

ControlUnit'in verdiği sinyalleri FSM diyagramına göre ayarladım. FSM'de ki case'ler toplam 3 bitle ifade edilebildiği için controlUnitte 3 bit register kullandık.