MARMARA ÜNİVERSİTESİ
TEKNİK BİLİMLER MESLEK YÜKSEKOKULU

LOGISIM İLE 8-BİT ÖRNEK BİLGİSAYAR TASARIMI

Bitirme Projesi

Fatih YILDIRIM
361516038

Bölüm: Bilgisayar Teknolojileri
Program: Bilgisayar Programcılığı (Uzaktan Eğitim)

Danışman: Prof.Dr. Vedat TOPUZ

2020

MARMARA ÜNİVERSİTESİ
TEKNİK BİLİMLER MESLEK YÜKSEKOKULU

LOGISIM İLE 8-BİT ÖRNEK BİLGİSAYAR TASARIMI

Bitirme Projesi

Fatih YILDIRIM
361516038

Bölüm: Bilgisayar Teknolojileri
Program: Bilgisayar Programcılığı (Uzaktan Eğitim)

Proje Sınavı:
Prof.Dr. Vedat TOPUZ
Dr.Öğr.Üyesi Zehra Aysun ALTIKARDEŞ
Öğr.Gör. Fatih KAZDAL
Öğr.Gör. Ercan ERKALKAN

2020

Özgünlük Bildirisi

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

LOGISIM İLE 8-BİT ÖRNEK BİLGİSAYAR TASARIMI

(ÖZET)

Günümüz dünyasında bilgisayarların ve mikroişlemcilerin kullanım alanları oldukça ileri boyutlara ulaşmıştır. Bu mikroişlemciler, evimize girerken anahtar görevi de görebiliyor ya da toplu taşıma araçları kullanırken sanal cüzdan vazifesini de üstlenebiliyor. Bu temel kullanımlar dışında mühendislik, tıp, astronomi, fen bilimleri, arkeoloji gibi birçok alanda da hayatımızı kolaylaştırıyor.

1950'li yıllarda tasarlanan bilgisayarlarda vakum tüplü diyotlarda kullanıldı. Sonrasında gelişen yarı-iletken teknolojisi ile bu bilgisayarların kapladığı alan büyük oran küçüldü. Buna paralel olarak saat hızı arttırılarak performansı da arttırıldı. Sonrasında ise, mikroişlemcinin bileşenlerinin mimarileri geliştirilerek yine performans artışı sağlandı.

Bu çalışma ile temel bir bilgisayarın çalışma sistemi, kesme mantığı ve bileşenlerinin tasarımı açıklanmaya çalışıldı. Çalışma kapsamında tasarlanan bu bilgisayar Logisim üzerinde simüle edildi.

İÇİNDEKİLER

I.GİRİŞ

İngilizce olan ve bugün “Bilgisayar” olarak çevirdiğimiz “Computer” kelimesi “Compute” ‘den gelmektedir. “Compute” hesaplamak demektir. 1950’li yıllarda bile “Computer” yani “Hesaplayıcı, Hesap Eden” olarak çalışan insanlar vardı. İnsanların bu görevde çalışması bazı hatalar doğurmuştur, çünkü insan hata yapabilen canlıdır. Ancak makinalar hata yapamaz. İşte tam da bu yüzden bilgisayar bilimleri daha hızlı ve daha doğru “Hesaplamalar” yapmak için geliştirildi. O günlerde tasarlanan bilgisayarlar şimdikilerin atası ve bu projenin de konusu olan Temel Bilgisayarlar idi.

Günümüz bilgisayarları, gelişen yarıiletken teknolojisi ile beraber günümüz bilgisayarları daha küçük boyutlara ve daha yüksek işlemci hızlarına sahiptir. Mimari olarak farklılıklar olsa temelde bütün bilgisayarlar ileride açıklayacağım kesmeler ve zamanlamalarla çalışmaktadır.

Tasarımını yapmış olduğum temel bilgisayar,

- 4096 x 16 Kelimelik belleğe,
- AR, DR, PC, AC, IR, TR, OUTR, INPR ve SC olmak üzere 9 adet kaydediciye,
- I, S, E, R, IEN, FGI ve FGO olmak üzere 7 adet bayrağa,
- 3x8 ve 4x16 olmak üzere 2 adet Decoder’e,
- 1 adet ALU’ya ve son olarak
- 1 adet 16-Bitlik ortak yola sahiptir.

Kaydedicilerin tasarımında JK Flip Flop, bayrakların tasarımında ise D Flip Flop kullandım. Geriye kalan Decoder ve bellek elemanları hazır olarak bulunmaktadır.

Projenin simülasyon çalışmasını tasarım aşamasında da kullanmış olduğum GNU lisansa sahip Logisim programı üzerinden yapılabilir.

II. PROJENİN TANIMI VE PLANI

Bu projede, 8-Bitlik örnek bir bilgisayarın Logisim programı üzerinde tasarımı yapılacaktır.

Projenin tamamlanması için 12 haftalık bir süreç planlanmıştır. Bu süreçte en fazla zaman (6 hafta) literatür taramasına ayrılmıştır. Projenin zaman planlaması Şekil 2.1'de bulunan Gantt diyagramında yer almaktadır.

İş Grubu	Görevler	ZAMANLANDIRMA ÇİZELGESİ (Hafta)											
		1	2	3	4	5	6	7	8	9	10	11	12
Projeye Hazırlık	Projenin Planlanması	■											
	Literatür Taraması		■	■	■	■	■	■					
Çizim Ve Test	Tasarlanan modüllerin çizim ortamına aktarılması								■	■	■		
	Çizilen bilgisayarın test edilmesi											■	
Projenin Teslimi	Proje Dökümanlarının Hazırlanması												■

Şekil 2.1 – Proje Planlamasının Gantt Diyagramı

Tasarım bileşenler halinde yapılmıştır. Uygulamada yer alan bu bileşenlerin geliştirilme süreleri hafta bazında Şekil 2.2'deki Gantt diyagramında yer almaktadır.

Yapılan İş	Görevler	ZAMANLANDIRMA ÇİZELGESİ (HAFTA)					
		1	2	3	4	5	6
Literatür Taraması	IEEE Taraması	■					
Literatür Taraması	Kitap Taraması		■	■			
Modüllerin Test Çizimleri	AR, DR, PC, AC, IR, TR, OUTR, INPR, SC				■	■	
Modüllerin Test ve Çizimleri	I, S, E, R, IEN, FGI, FGO						■

Şekil 2.2 – Tasarımdaki bileşenlerin geliştirilme sürelerinin Gantt diyagramı

III. KURUMSAL BİLGİLER

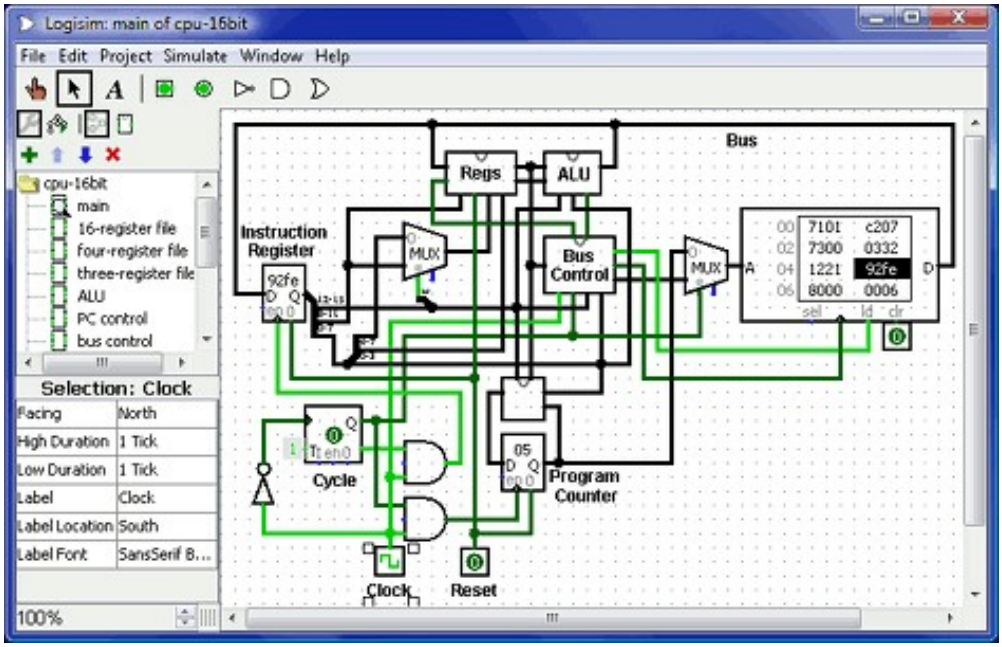
3.1. Logisim Nedir?

Java tabanlı olan bu program, kullanıcıların sayısal mantık devrelerini grafiksel olarak düzenlemelerine ve simüle etmelerine olanak sağlar. Basit bir araç çubuğu ile istenilen sayısal devre elemanı buradan alınarak istenilen tasarım yapılıp simüle edilebilir.

Gerekmesi halinde araç çubuğundaki elemanlardan farklı bir devre elemanı oluşturup buna blok şekli verilmesine olanak sağlar.

Program, Doç.Dr. Carl BURCH liderliğindeki bir ekip tarafından geliştirilmiş olup açık kaynak olarak bulunmaktadır.

Logisim programına ait ekran görüntüsü Şekil 3.1'de yer almaktadır.



Şekil 3.1 – Logisim programına ait ekran görüntüsü

[https://www.hendrix.edu/research/default.aspx?id=55783]

[http://www.cburch.com/logisim/]

3.2. Mikroişlemci Nedir?

Bellekte saklı bir komut dizisini ardışıl olarak yerine getirerek veri kabul edebilen ve bunları işleyebilen sayısal bir elektronik eleman olarak tanımlanır. Bilgisayarlar 3 farklı yapıdan oluşur. Bunlar,

- Merkezi işlem birimi (CPU)
- Giriş çıkış birimleri
- Bellek

3.3. Mikroişlemci Nasıl Çalışır?

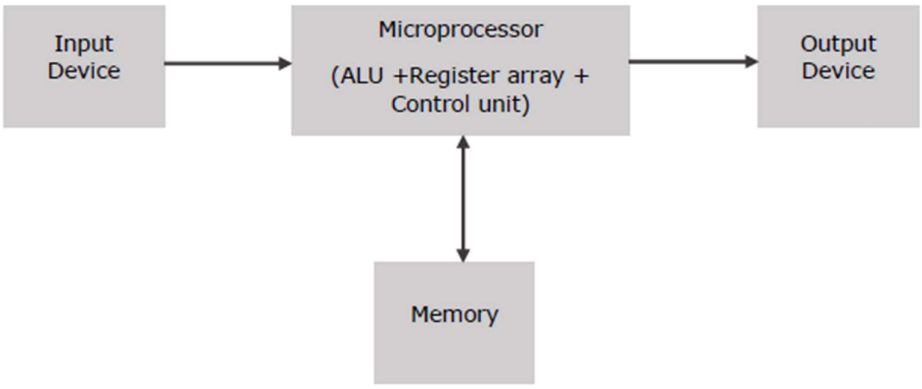
Bir mikroişlemcinin çalışmasında, kontrol birimi tarafından yerine getirilen iki işlem vardır; 1-Komut okuma(**fetch**) ve 2-Komut yürütme(**execute**).

Komut okuma, mikroişlemcinin hafızadan bir işlem kodu (**operation code ya da opcode**) komut kaydedicisine (**Instruction Register- IR**) getirme işlemidir. IR'ye gelen kod ile hangi işlemin yapılacağı komut kod çözücüsü tarafından belirlenir. Gerekli olan zamanlama kontrol birimi tarafından üretilir.

Son olarak komutun yürütülmesi gerçekleştirilir. Durma komutu gelene kadar bu işleyiş döngü halinde devam eder.

3.4 Mikroişlemcinin Birimleri Nelerdir?

Basit bir mikroişlemci kaydediciler (Register), Aritmetik- Mantık Birimi (ALU), zamanlama ve kontrolü sağlayan Kontrol birimlerinden oluşur. Basit bir mikroişlemcinin içeriği Şekil 3.2'de yer almaktadır.



Şekil 3.2.- Mikroişlemcinin İçeriğinin Bloklar Halinde Gösterimi

3.4.1 Kaydediciler

Aldığı bilgiyi geçici olarak depolayan, yaz-bozlardan oluşmuş bellek birimleridir. Kaydediciler, bellek adreslerini tutmak ve icra edilecek buyrukların adreslerini hesaplamada kullanılır. Tasarımda kullanılan kaydediciler ve görevleri Şekil 3.3'de yer almaktadır.

Kaydedici	Bit Sayısı	Adı	Görevi
DR	16	Veri Kaydedicisi	Bellek Verisini Tutar
AR	12	Adres Kaydedicisi	Bellek için Adres Tutar
AC	16	Akümülatör	Aritmetik ya da mantıksal işlemlerden sonra sonucu tutar.
IR	16	Buyruk Kaydedicisi	Buyruğun kodunu tutar.
PC	12	Program Sayıcı	Buyruğun adresini tutar.
TR	16	Geçici Yazaç	Geçici veriyi tutar.
INPR	8	Giriş Yazacı	Giriş verisini tutar.
OUTR	8	Çıkış Yazacı	Çıkış verisini tutar.
SC	4	Sıra Sayıcı	Birimlerin çalışma sırasını sağlamak için zamanlama sinyali üretir.

Şekil 3.3.- Temel Bir Bilgisayarda Kullanılan Kaydediciler ve Görevleri

Her kelime 16-bittir. Bir buyruğun 12-bitlik kısmı verinin adresi için kullanılır. 3-Bit işlem kodu için kalan 1-bit ise doğrudan ya da dolaylı adreslemeyi belirlemek için kullanılır.

DR bellekten okunan veriyi tutar. Bellekten okunan buyruk IR'de tutulur. TR, işlem sırasında verilerin geçici olarak tutulmasında kullanılır. PC, bir programın çalışması sırasında ardışık değerler alarak, yani sayarak, belleğe daha önce yerleştirilmiş buyrukların sırasıyla okunmasını sağlar. SC'den çıkan sinyaller 4x16 kod çözücüye gelir ve bu kod çözücünün çıkışları zamanlama sinyalini oluşturur.

3.4.2. Aritmetik ve Mantık Birimi (ALU)

Aldığı veriler üzerinde aritmetik ve mantıksal işlemler yapıp elde ettiği sonucu AC'ye aktaran birimdir. Her ne kadar ayrı bir gibi gözükse de AC, DR ve INPR ile birlikte çalışır. Yaptığı işlemler Şekil 3.4'te yer almaktadır. Detaylı bağlantı şeması ise Şekil 4.17'de yer almaktadır.

D_0T_5: $AC \leftarrow AC \wedge DR$
D_1T_5: $AC \leftarrow AC + DR$
D_2T_5: $AC \leftarrow DR$
pB_{11}: $AC(0\text{–}7) \leftarrow INPR$
rB_9: $AC \leftarrow \overline{AC}$
rB_7: $AC \leftarrow \text{shr } AC, \quad AC(15) \leftarrow E$
rB_6: $AC \leftarrow \text{shl } AC, \quad AC(0) \leftarrow E$
rB_{11}: $AC \leftarrow 0$
rB_5: $AC \leftarrow AC + 1$

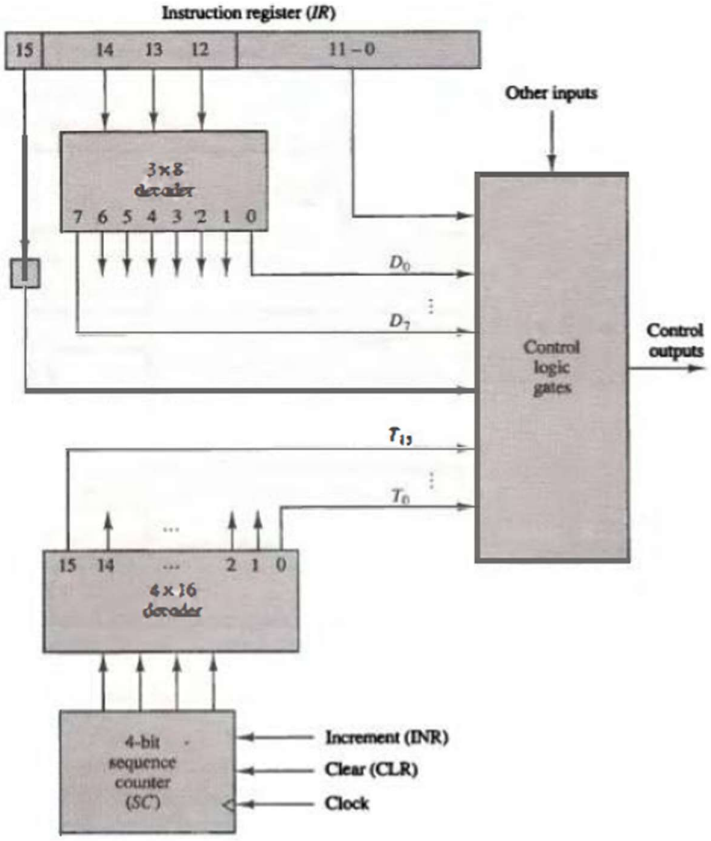
Şekil 3.4.- ALU Tarafından Yapılan Aritmetik ve Mantıksal İşlemler

3.4.3. Kontrol Birimi

Bu birimden çıkan kontrol sinyalleri ile,

- 9 tane kaydedicinin girişleri kontrol edilir.
- Belleğin oku-yaz girişleri kontrol edilir.
- Yaz-Bozları 1,0 ve tümleyen sinyaller kontrol edilir.
- Yol seçicide kullanılan s_2, s_1 ve s_0 sinyalleri üretilir.
- AC'nin toplayıcı ve mantık devrelerini kontrol eden sinyaller üretilir.

Kısaca, sistemin tüm işleyişinden ve işlemlerin zamanında yapılmasından sorumludur. Kontrol biriminin iç tasarımına ait referans şema Şekil 3.5'te yer almaktadır.



Şekil 3.5.- Kontrol Birime ait Referans Şema

IV. TASARIM VE GERÇEKLEME

4.1. Proje Geliştirme Aşamaları

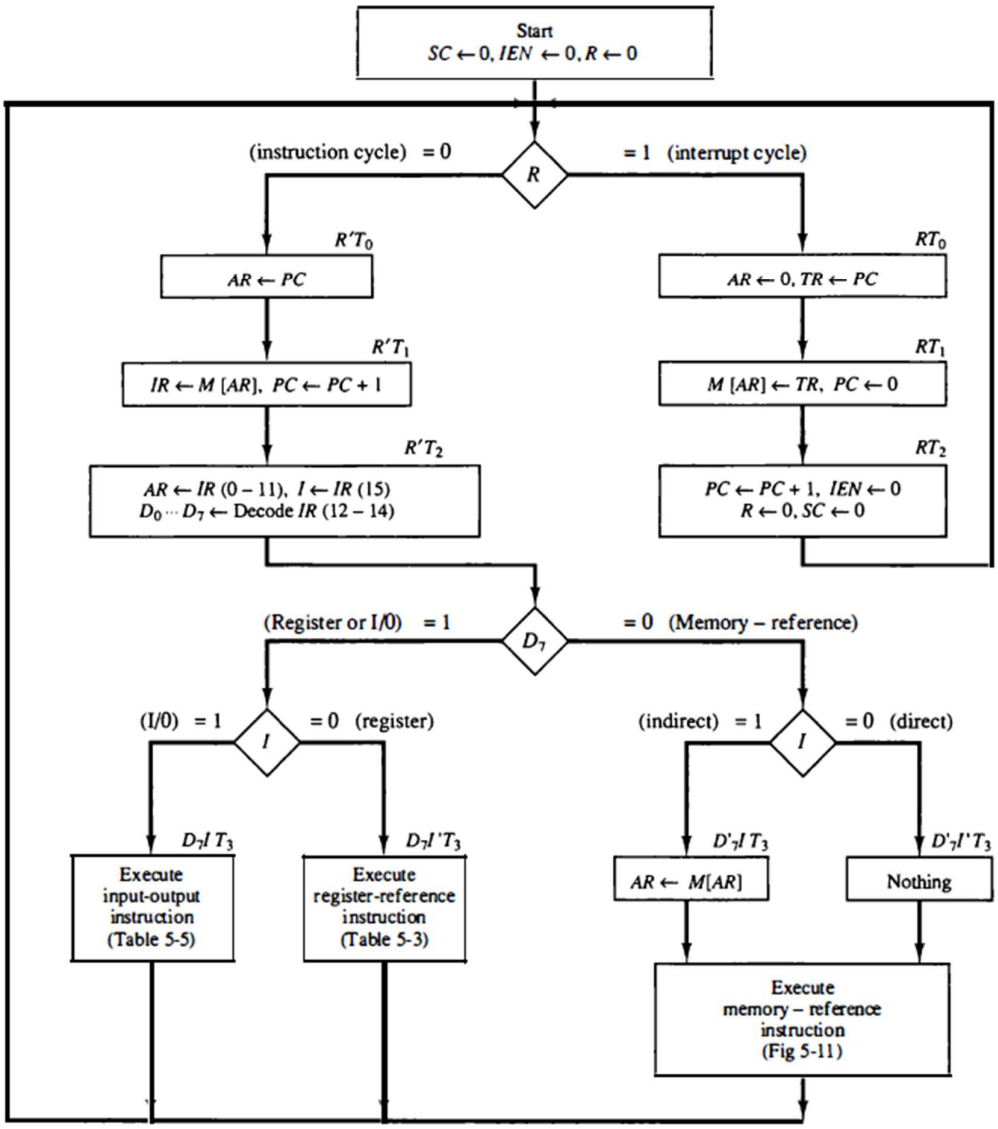
Projenin geliştirilmesi sürecine gerekli literatür taraması ile başlanmış, projede kullanılacak bileşenlerin belirlenmesi ve sonrasında ise bu bileşenlerin birbirleriyle olan ilişkilerinin oluşturulmasıyla devam edilmiştir. Daha sonraki adımlarda temel mantık kapıları ile bu bileşenlerin tasarımlarının yapılıp bloklar haline getirilmesi ve son aşama olarak da bu blokların birleştirilmesi sürecine geçilmiştir.

Proje geliştirilirken izlenen adımlar sırasıyla aşağıda sırasıyla belirtilmiştir.

- Proje kapsamının ve belirlenmesi
- Proje için gerekli kitabın temin edilmesi
- Temel mantık devrelerinin çalışmasının kavranması
- Tasarımda kullanılacak diğer elemanların çalışmasının kavranması
- Projede yer alacak olan bileşenlerin ve birbirleriyle olan ilişkilerinin kavranması
- Zaman ihtiyacının belirlenmesi
- Proje için zaman planlamasının yapılması
- Proje için tasarım ortamının edinilmesi
- Kaydedici tasarımının yapılması ve blok haline getirilmesi
- Blok haline getirilen kaydedici kullanılarak 9 adet kaydedicinin tasarlanması
- Bayrakların tasarımlarının yapılması
- Kontrol biriminin tasarımının yapılması
- Aritmetik mantık birimine ait tasarımın yapılması
- Ortak yolun tasarlanması
- Tasarlanan bütün elemanların birleştirilmesi

4.1.1. Proje Kapsamının Belirlenmesi

Geliştirme aşamasına geçmeden önce, ilk olarak bilgisayarın özellikleri, tasarımının nasıl yapılması gerektiği ve çalışacağı ortam belirlendi. Bununla ilgili olarak bilgisayarın çalışma dair akış diyagramı Şekil 4.1'dedir.



Şekil 4.1. – Bilgisayarın Çalışmasını Açıklayan Akış Grafiği

Akış diyagramın elde ettiğim sonuçlara göre önce kaydedicilerin ve bayrakların tasarımını yapıp sonrasında bütün elemanları ana ekranda birleştirmem gerektiği sonucuna vardım.

4.1.2. Proje İçin Gerekli Kitabın Temin Edilmesi

Tasarıma referans olması ve tasarım aşamasında temel mantık devrelerinin işleyişinin anlaşılabilmesi için, projede kullanılacak olan kitap temin edildi.

4.1.3. Temel Mantık Devrelerinin Çalışmasının Kavranması

Bilgisayarın tasarımı mantık kapıları ile yapılacağı için temel mantık kapılarının nasıl çalıştığı, doğruluk tabloları ve Boolean Cebri çalışıldı.

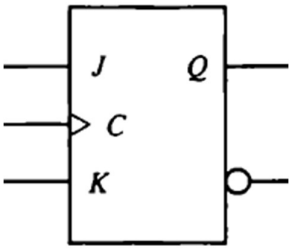
4.1.4. Tasarımda Kullanılacak Diğer Elemanların Çalışmasının Kavranması

Tasarımda kullanılacak olan tam toplayıcılar, MUX, yaz-bozlar, Decoderler ve bunların durum tabloları üzerinde çalışıldı. 1 tane yaz-boz 1-bitlik veri depolar. Bu bilgiyi kullanarak kaydediciler de yaz-boz olarak JK ve D yaz-bozlarını kullandım. Toplamda 4 tane yaz boz çeşidi olmasına karşın, JK ve D tiplerini diğer 2 yaz boza göre daha stabil çalışmalarından ötürü tercih ettim.

MUX'lar girişteki veriyi, seçme uçlarında aldıkları sinyale göre çıkışa yönlendirirler.

Decoder ise, ikili kodlanmış veriyi çözerek ilk halini bulmak için kullanılır.

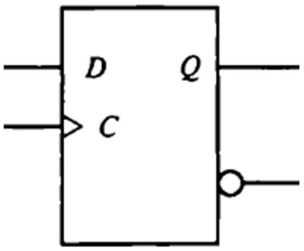
Yaz-bozların uyarma tabloları Şekil 4.2 ve Şekil 4.4'de yer almaktadır.



Şekil 4.1 – JK Yaz-Bozu'na ait Grafik

$Q(t)$	$Q(t+1)$	J	K
0	0	0	×
0	1	1	×
1	0	×	1
1	1	×	0

Şekil 4.2 – JK Yaz-Bozu'na ait Uyarma Tablosu

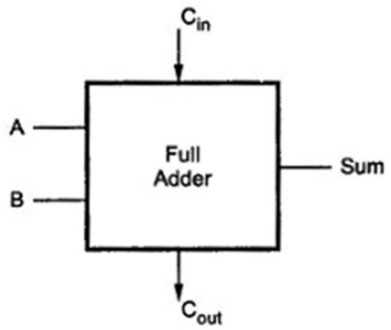


Şekil 4.3 – D Yaz-Bozu'na ait Grafik

$Q(t)$	$Q(t+1)$	D
0	0	0
0	1	1
1	0	0
1	1	1

Şekil 4.4 – D Yaz-Bozu'na ait Uyarma Tablosu

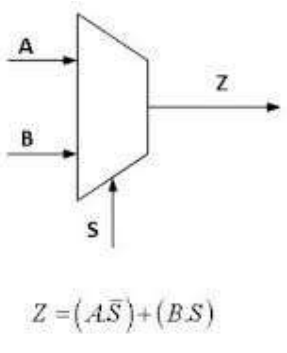
Benzer şekilde tam toplayıcı, MUX ve decoderin doğruluk tablosu sırasıyla Şekil 4.6, Şekil 4.7 ve Şekil 4.9'da yer almaktadır.



Şekil 4.5. – Tam Toplayıcı Devresine ait Grafik

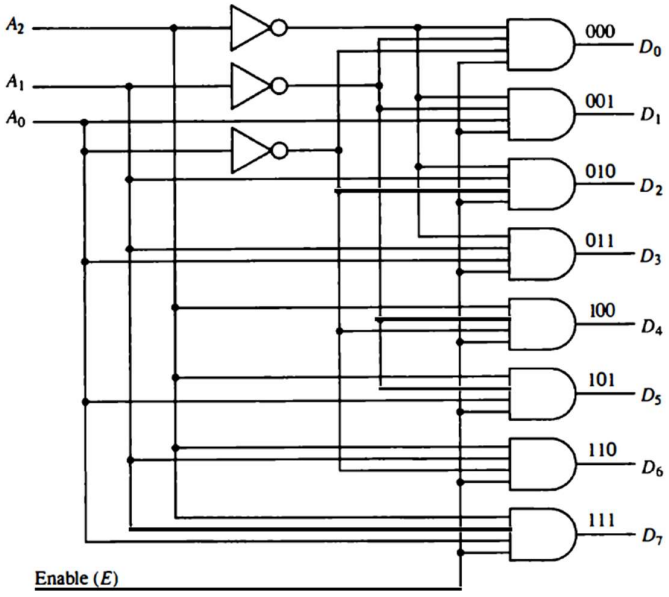
A	B	C_{in}	Carry	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Şekil 4.6. – Tam Toplayıcı Devresine ait Doğruluk Tablosu



S	A	B	Z
0	0	0	0
	0	1	0
	1	0	1
	1	1	1
1	0	0	0
	0	1	1
	1	0	0
	1	1	1

Şekil 4.7. – MUX'a ait Grafik ve Doğruluk Tablosu



Şekil 4.8. – 3X8 Decoder'e ait Grafik

E	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	×	×	×	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	1
1	0	0	1	0	0	0	0	0	0	1	0
1	0	1	0	0	0	0	0	0	1	0	0
1	0	1	1	0	0	0	0	1	0	0	0
1	1	0	0	0	0	0	1	0	0	0	0
1	1	0	1	0	0	1	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0
1	1	1	1	1	0	0	0	0	0	0	0

Şekil 4.9. – 3x8 Decoder'e ait Doğruluk Tablosu

4.1.5. Projede Yer Alacak Olan Bileşenlerin ve Birbiriyle Olan İlişkilerinin Kavranması

Tasarımda kullanılacak olan 9 adet kaydedici (**AR, PC, DR, AC, IR, TR, OUTR, INPR, SC**) ve 7 adet bayrağın (**I, S, E, R, IEN, FGI, FGO**) birbirlerine nasıl bağlanacağı üzerinde çalışıldı. Kullanılacak olan kaydedicilerin detaylı tasarımı başlık 4.1.8 ve 4.1.9' da anlatılmıştır.

Şekil 4.11'den da görüleceği gibi bütün bileşenler 16-bitlik bir ortak yol üzerinden birbirlerine bağlanmaktadır. Ancak hangi bileşenin ortak yola ne zaman veri aktaracağı ve ne zaman veri alacağını seçmek için yol kontrol yapısı oluşturulması gerekmektedir. Şekil 4.10'da yol kontrol yapısının fonksiyonu için gerekli olan doğruluk tablosu görülmektedir.

Inputs							Outputs			Register selected
x_1	x_2	x_3	x_4	x_5	x_6	x_7	S_2	S_1	S_0	for bus
0	0	0	0	0	0	0	0	0	0	None
1	0	0	0	0	0	0	0	0	1	*AR*
0	1	0	0	0	0	0	0	1	0	*PC*
0	0	1	0	0	0	0	0	1	1	*DR*
0	0	0	1	0	0	0	1	0	0	*AC*
0	0	0	0	1	0	0	1	0	1	*IR*
0	0	0	0	0	1	0	1	1	0	*TR*
0	0	0	0	0	0	1	1	1	1	Memory

Şekil 4.10.- Bütün Kaydedicilerin Veri Yolunu Kullanma Sırasını Gösteren Doğruluk Tablosu

Burada dikkat edilmesi gereken hiçbir zaman aynı anda iki girişin de lojik-1 seviyede olamayacağı. Sırasıyla bütün çıkışlar için üretilen fonksiyonlar Şekil 4.11'de yer almaktadır.

$$S_0 = x_1 + x_3 + x_5 + x_7$$
$$S_1 = x_2 + x_3 + x_6 + x_7$$
$$S_2 = x_4 + x_5 + x_6 + x_7$$

Şekil 4.11.- Yol Kontrol Birimi için Gerekli Olan Seçme Uçlarına Ait Fonksiyon

AR'nin yola veri aktarabilmesi için, aktarım yapabileceğini de Şekil 4.12 ve EK-1'den anlıyoruz, $s_2s_1s_0 = 001$ olması gerekiyor. Bu durumda EK-1'deki tablo yardımı ile x_1'e ait fonksiyonu Şekil 4.12'deki AR transfer fonksiyonundan buluruz. Bu durumda elde edilen x_1'e ait fonksiyon Şekil 4.13'te yer almaktadır.

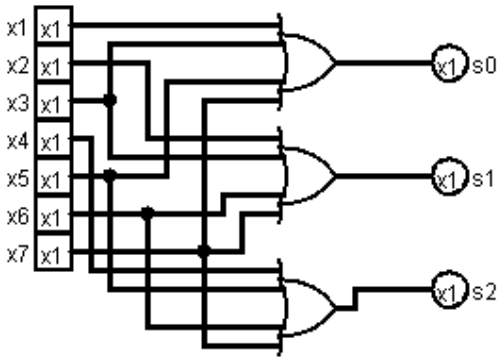
$$D_4T_4: \quad PC \leftarrow AR$$
$$D_5T_5: \quad PC \leftarrow AR$$

Şekil 4.12.- AR Transfer Fonksiyonu

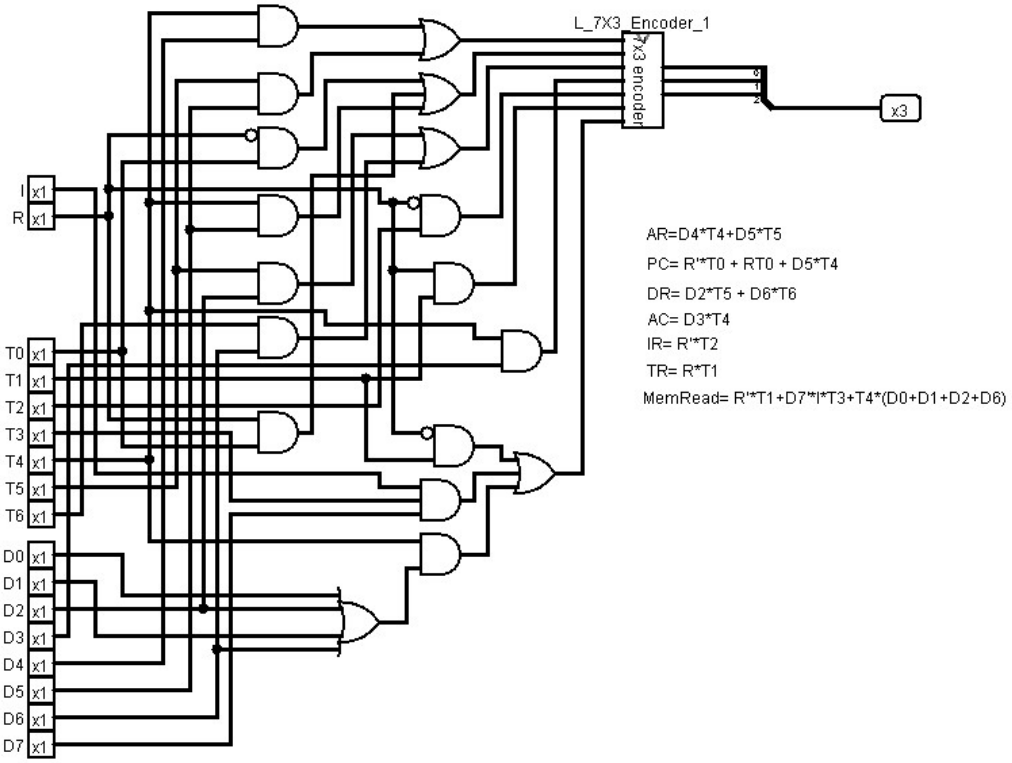
$$x_1 = D_4T_4 + D_5T_5$$

Şekil 4.13.- x_1 Fonksiyonu

Benzer yöntemi kullanarak diğer fonksiyonları da türetip devrenin tasarımını yaptım. Buna göre elde etmiş olduğum 7x3 Encoder devresi ve Yol Kontrol devresi sırasıyla Şekil 4.14 ve Şekil 4.15'te yer almaktadır.

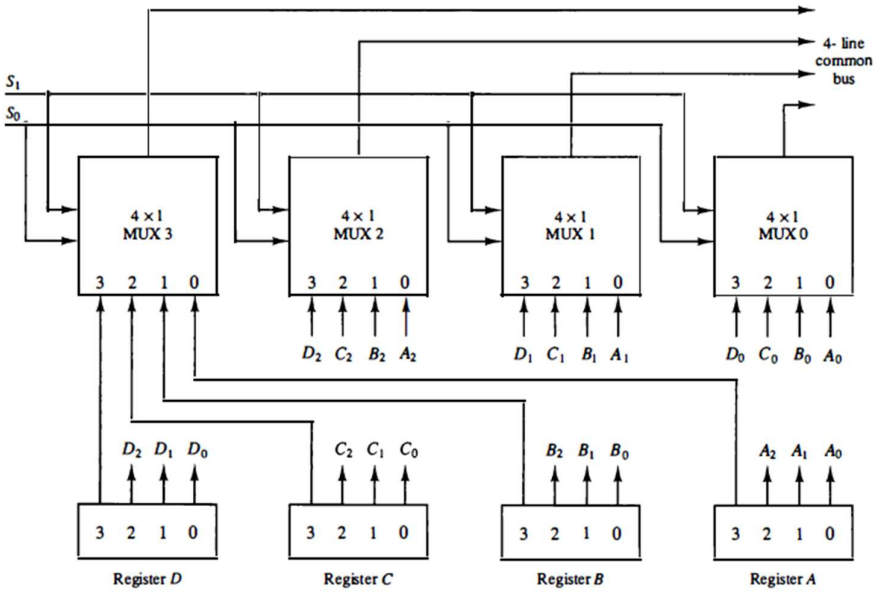


Şekil 4.14.- 7x3 Encoder Devresi



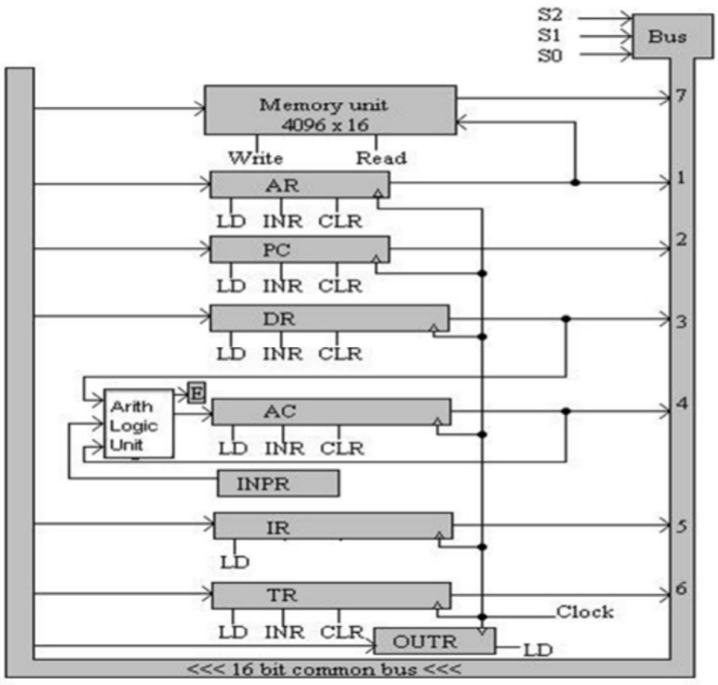
Şekil 4.15.- Yol Kontrol Yapısı

Geriye kalan 16-bitlik yolun tasarımını ise Şekil 4.16'daki şemayı referans alarak yaptım. Bütün kaydedicilerin en düşük anlamlı bitinden başlanarak yola bağlanır ve seçme girişleri de Yol Kontrol yapısının çıkışından elde edilen $s_2s_1s_0$ uçları ile sağlanır.



Şekil 4.16.- Ortak Yol'a ait Referans Şema

Ortak Yol, Yol Kontrol ve Yol Seçici (7x3 Encoder) tasarımını tamamladıktan sonra devreyi Şekil 4.17'deki gibi topladım.



Şekil 4.17. – Bilgisayara ait Ortak Yol ve Birbirleri Aralarındaki İlişkileri

4.1.6. Zaman İhtiyacının Belirlenmesi

Projede gerçekleştirilecek adımların ve hangi zamanda hangi aşamada olunması gerektiği zaman planlaması diyagramında belirlenmiştir (bkz. Şekil 2.1). Yapılan plana göre 12 haftada tamamlanması gereken projede en uzun süre 6 hafta ile literatür taramasın ayrılmış olup literatür taramasının içeriği Şekil 2.2.'de detaylı olarak açıklanmış olup bu plana göre hareket edilmiştir.

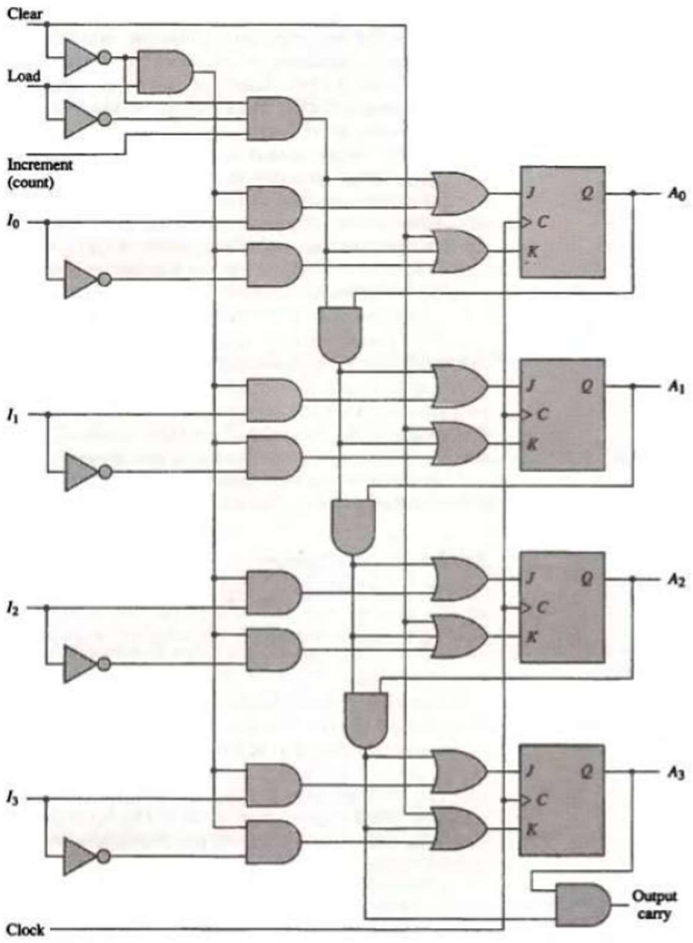
4.1.7. Proje İçin Tasarım Ortamının Edinilmesi

Projenin tasarımının bilgisayar destekli ortama aktarılabilmesi için Logisim programı kuruldu.

4.1.8. Kaydedici Tasarımının Yapılması ve Bloklar Haline Getirilmesi

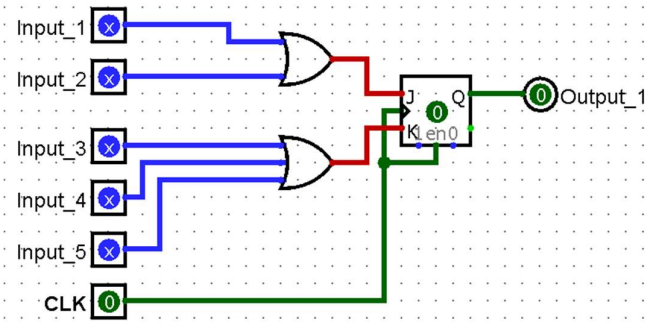
Kaydedicilerin tasarımı için önceki başlıkta (bkz. 4.1.4) yer alan JK yaz bozlarını kullanıldı. Önce 1-bitlik ardından da ihtiyaca göre 8, 12 ve 16-bitlik kaydedicilerin tasarımları yapıldı, bloklar haline getirilerek ana tasarıma eklendi.

Tasarıma referans olan kaydedici şeması Şekil 4.18'de yer almaktadır.



Şekil 4.18. – 4-Bitlik Paralel Yüklemeli Kaydedici Şeması

Görüleceği üzere veri girişleri yaz bozlara paralel olarak yüklenmiş ve Artış (INC), Veri girişi (LOAD) ve Temizleme (CLR) yetki uçları ile kaydedicinin kontrolü sağlanmıştır. Buna göre tasarlamış olduğum 1-bitlik kaydedici 4.19'dadır.



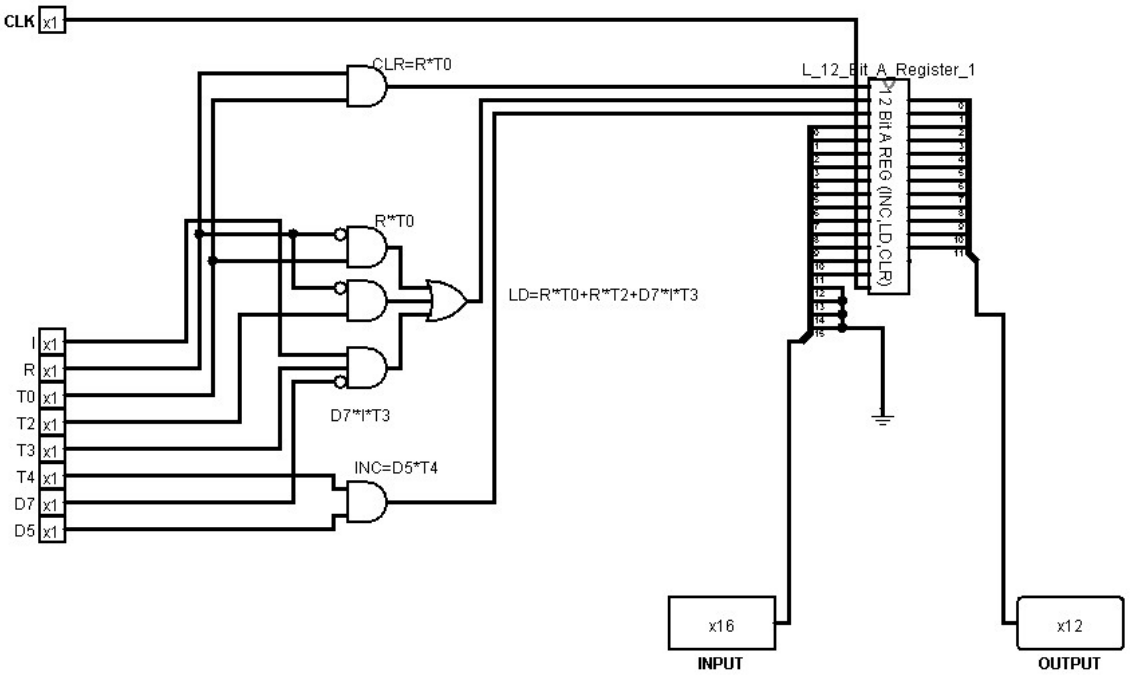
Şekil 4.19 – 1-Bitlik Kaydedici

Şekil 4.18'de de görüldüğü üzere veri girişleri dışında yükleme, artış ve temizleme olmak üzere 3 adet harici kontrol girişi bulunmaktadır. Bu girişler, kaydedicinin değerinin ne zaman artacağını ne zaman sıfırlanacağını ve ne zaman veri girişi ile yükleneceğini kontrol etmek için kullanılır.

4.1.9. Bloklar Haline Getirilen Kaydediciler Kullanılarak 9 Adet Kaydedicinin Tasarlanması

Bloklar haline getirilen bu kaydedicileri kullanarak AR, PC, DR, AC, IR, TR, OUTR, INPR ve SC olmak üzere bilgisayarın temel bileşenleri arasında bulunan bu 9 adet kaydedicinin tasarımı yapıldı.

Tasarım aşamasında EK-1'de yer alan Kontrol Fonksiyonları tablosundan her kaydediciye ait Yükleme (**Load**), Temizleme (**Clear**) ve Artış (**INC**) yetki uçları için gerekli fonksiyonlar çıkarıldı. Örnek olması açısından Şekil 4.20'de AR (Adres Kaydedicisi)' ye ait tasarım şeması bulunmaktadır.



Şekil 4.20.- AR'nin Temel Mantık Kapıları ve 12-Bitlik Kaydedici ile Tasarlanmış Hali

4.1.10. Bayrakların Tasarımlarının Yapılması

Bayrak genel olarak denetleme işlemi için kullanılır. Örneğin FGI 1-Bitlik bayrak olup, girişte yeni bir veri olup olmadığını kontrol eder. Eğer varsa durumu lojik-1 olur. Bilgiler bilgisayar tarafından kabul edilince de durumu lojik-0 olur.

Benzer şekilde FGO ise çıkış bayrağı olarak kullanılır. Başlangıçtaki değeri lojik-1'dir. Bilgisayar bu bite bakar ve değerini lojik-1 olarak görünce de AC, OUTR kaydedicisine veriyi aktarır. Sonrasında da FGO lojik-0 seviyesine çekilir.

Diğer bayrakların görevleri ise kısaca şöyledir,

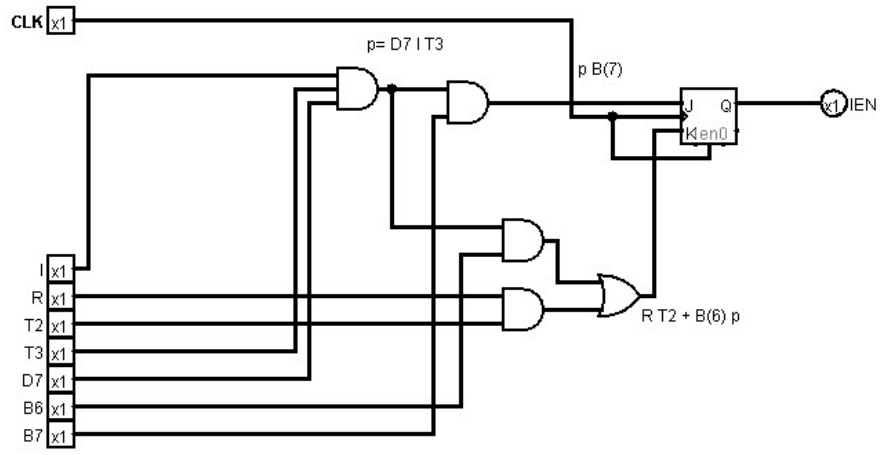
R: R=0 iken bilgisayar buyruk sürecini icra eder.

IEN: Kesim izin bayrağıdır. IEN = 1 iken bilgisayar kesim sürecindedir. Lojik -0 olduğunda ise bilgisayarın kesime gitmesine izin verdiğini gösterir.

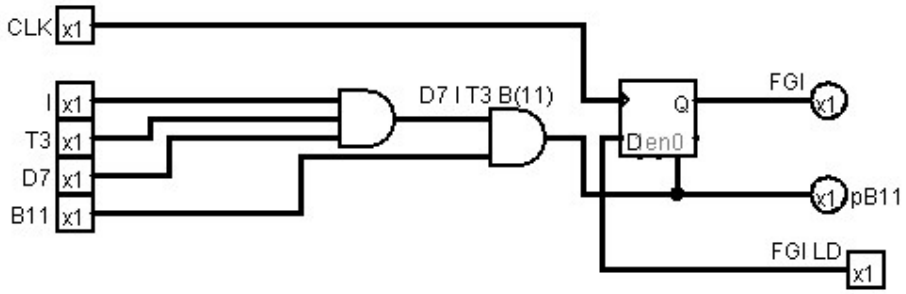
E: ALU'daki taşma bilgisini gösteren bayraktır. Eğer taşma varsa lojik-1 olur.

JK ve D yaz-bozları kullanılarak tasarımda yer alan I, S, E, R, IEN, FGI ve FGO bayraklarının tasarımı yapıldı.

Şekil 4.21'de IEN bayrağının ve 4.22'de FGI bayrağının tasarım şeması yer almaktadır.



Şekil 4.21.- IEN Kesme İzin Bayrağı



Şekil 4.22.- FGI Giriş Verisi Kontrol Bayrağı

4.1.11. Kontrol Biriminin Tasarımının Yapılması

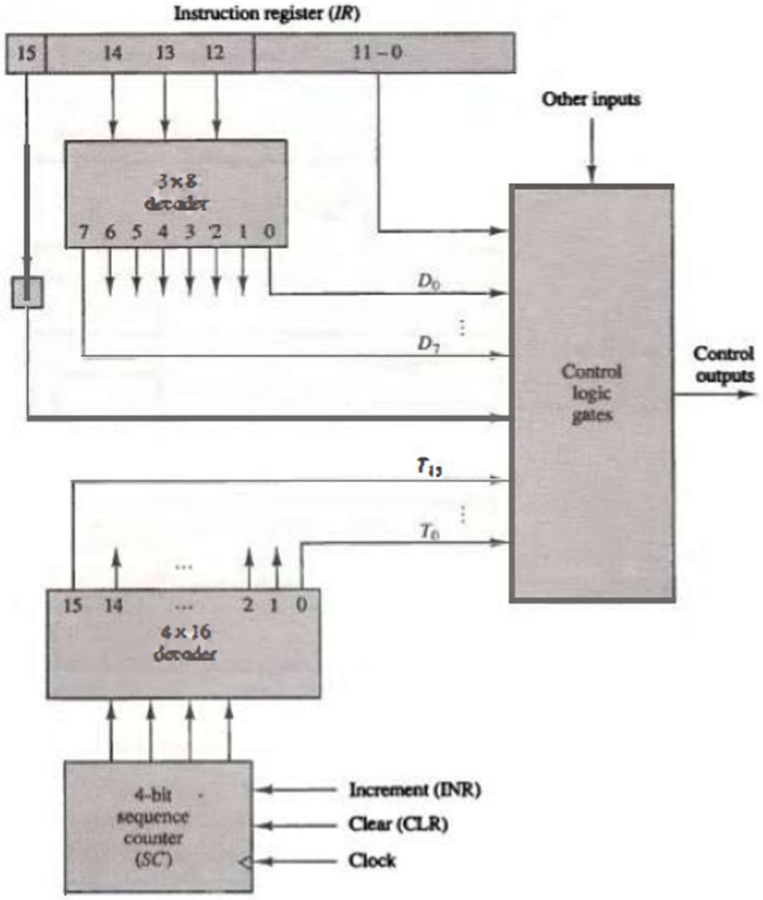
Bilgisayarın çalışmasındaki kontrol sinyallerinin üretimini sağlayan birim olan Kontrol biriminin 3x8 ve 4x16'lık Decoderler kullanılarak tasarımı yapıldı.

Burada da gerekli giriş sinyalleri EK-1'de yer alan Kontrol Fonksiyonları tablosundan elde edilerek Şekil 4.23'teki gibi bulundu. Sonrasında Şekil 4.24'te yer alan şema, referans alınarak elde edilen RTL fonksiyonları ile Şekil 4.25'teki gibi Kontrol Birimi tasarımı yapıldı.

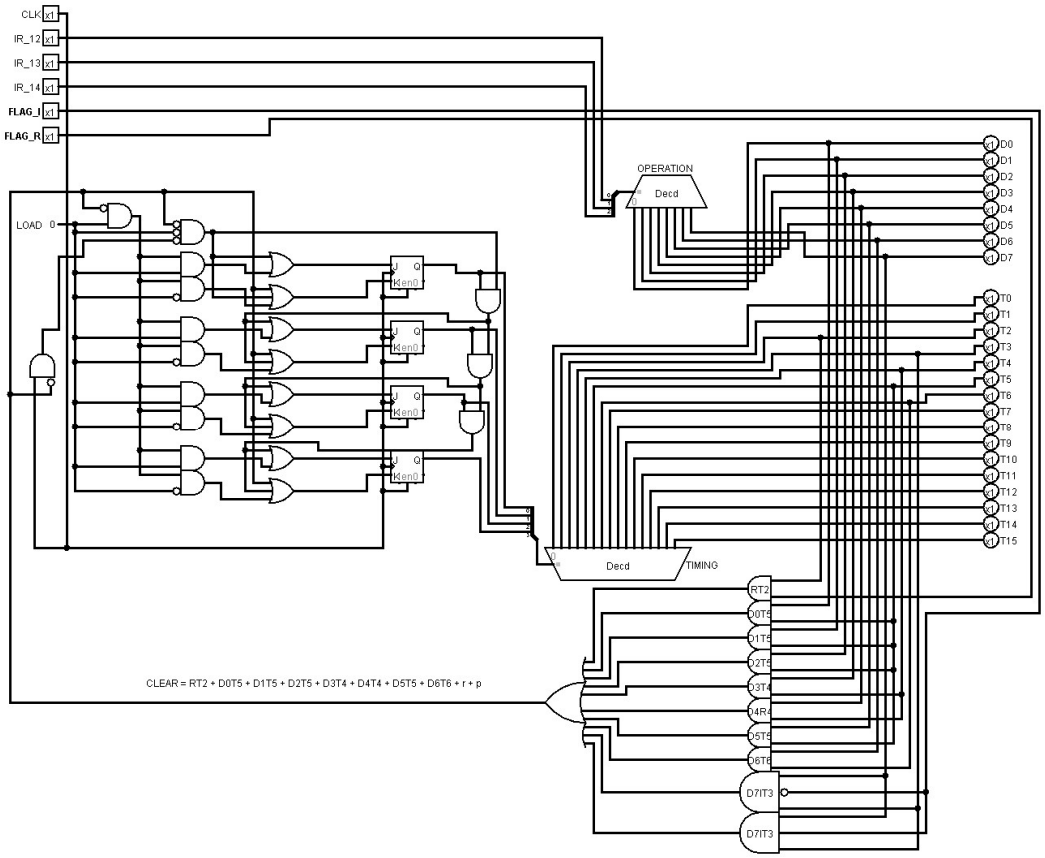
RT_2	SC←0
D_0T_5	SC←0
D_1T_5	SC←0
D_2T_5	SC←0
D_3T_4	SC←0
D_4T_4	SC←0
D_5T_5	SC←0
D_6T_6	If (DR=0), SC←0
$D_7I'T_3$	SC←0
D_7IT_3	SC←0

Şekil 4.23.- SC'nin Temizleme (CLR) Sinyal Ucu için Elde Edilen RTL Fonksiyonları

SC (Sıra Sayıcı) gelen her saat darbesiyle artar. Ancak Şekil 4.23'te yer alan sinyallerden herhangi biri geldiğinde ise Lojik-0 seviyesine çekilir.



Şekil 4.24.- Kontrol Biriminin Referans Şeması

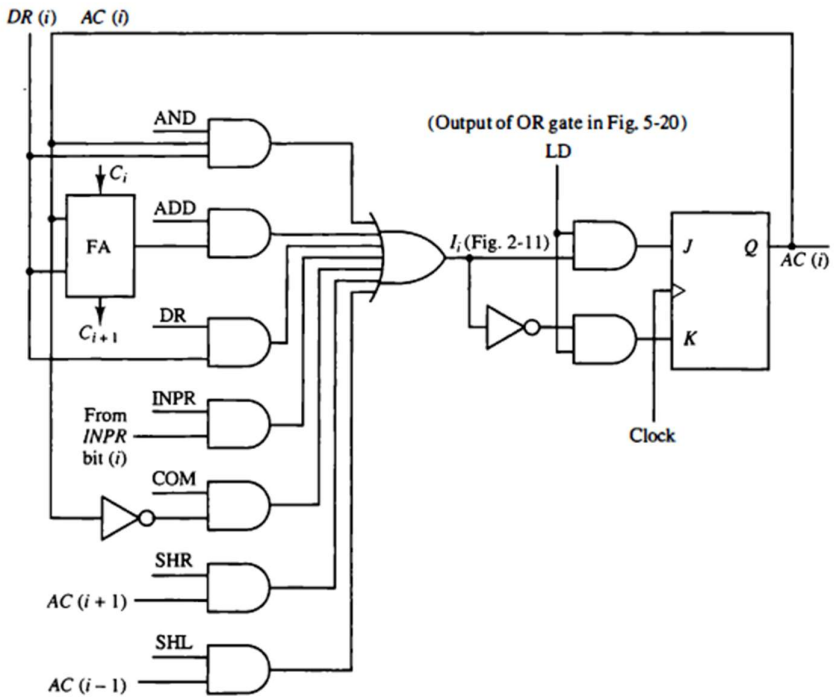


Şekil 4.25.- Kontrol Biriminin Tasarım Şeması

4.1.12. Aritmetik Mantık Birimine Ait Tasarımın Yapılması

Daha önce tasarımı yapılmış olan (bkz. Başlık 4.1.9) kaydedici kullanılarak AC'ye giren veriler üzerinde mantıksal ve aritmetik işlemler yapan Aritmetik Mantık Birimi'nin tasarımı yapıldı.

Aritmetik Mantık Birimi 16 eşit parçaya ayrılabilir. Her bir parça da AC'nin 1-bitini temsil eder. Şekil 4.26'da da gösterildiği gibi girişler I_i ile ve çıkışlar AC(i) ile gösteriliştir. Yükle (LD) girişi, AC'nin LD girişinden alınır, izinlendirildiğinde 16 tane I_i girişi AC'nin 0-15 bitine aktarılır.

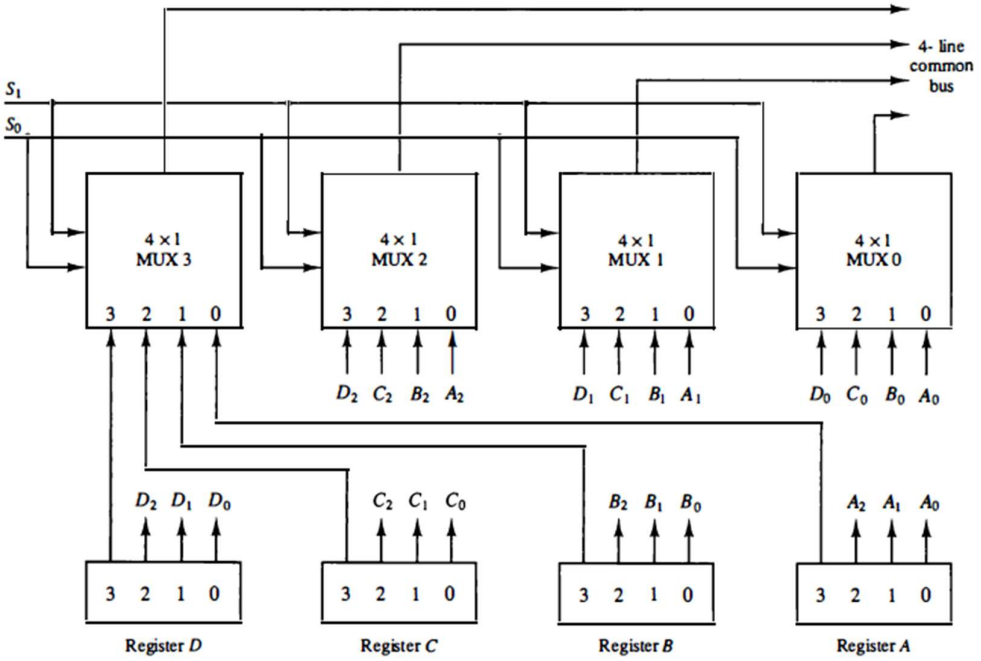


Şekil 4.26.- Aritmetik Mantık Biriminin Tasarım Şeması

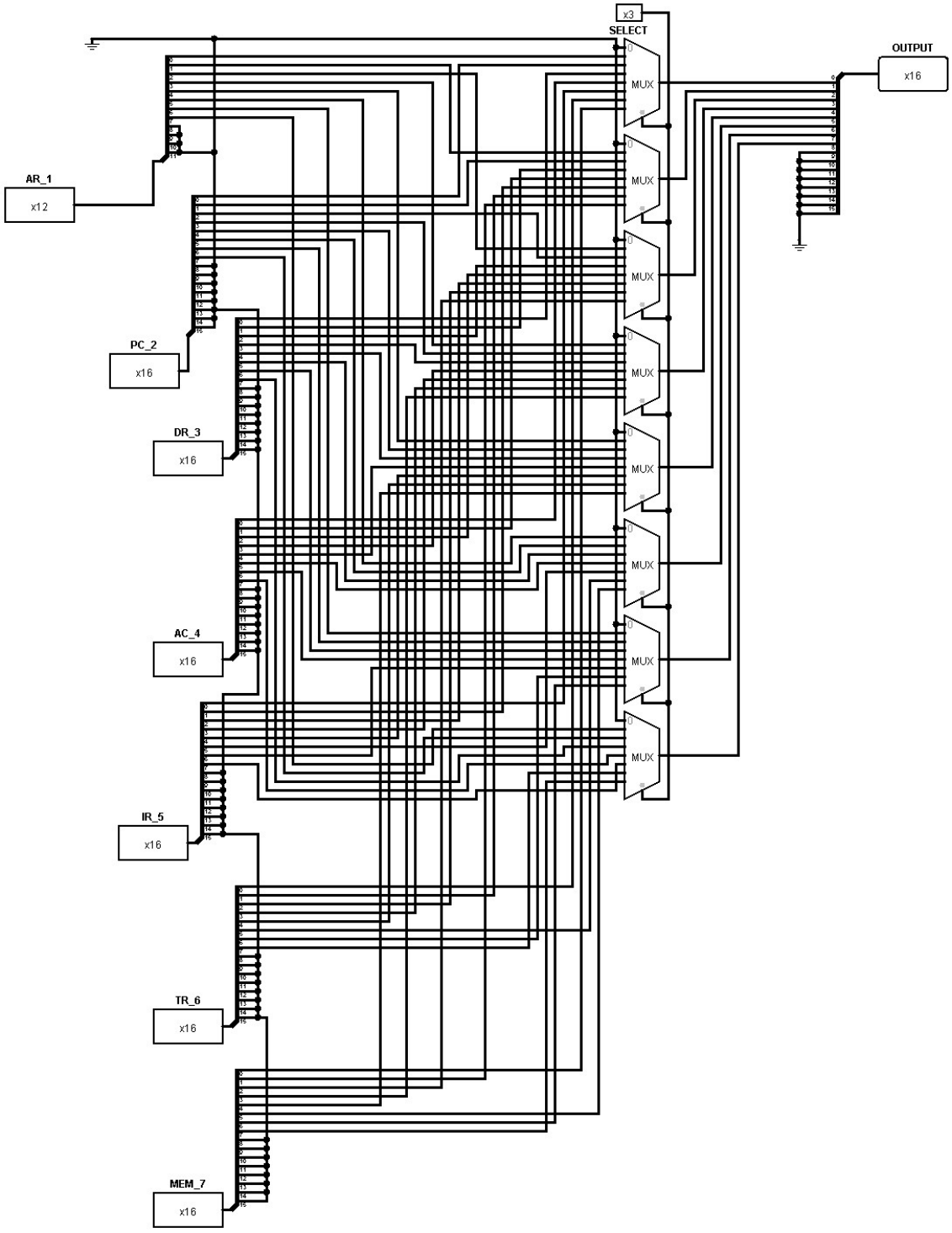
4.1.13. Ortak Yolun Tasarlanması

Bütün kaydedicilerin ve belleğin veri alışverişinde bulanabileceği 16-bitlik ortak veri yolunun tasarımı yapıldı.

Önceki bölümde (bkz.4.1.4) yol seçicilerin (MUX) nasıl çalıştığı doğruluk ve grafik üzerinde anlatılmıştı. Burada, bahsedilen özelikler kullanılarak kaydedicilerin ve belleğin kullanabileceği 16-bitlik bir ortak yol için bu bilgiler kullanıldı. Tasarıma ait referans şema Şekil 4.27'de ve tasarımı Şekil 4.28'de yer almaktadır. Girişlerde bulunan S_1S_0 ile çıkışa, yani yola hangi kaydedicinin veri bırakacağı seçilir (bkz. 4.1.5 Projede Yer Alacak Olan Bileşenlerin ve Birbiriyle Olan İlişkilerinin Kavranması)



Şekil 4.27.- 16-Bitlik Ortak Yol Tasarımına ait Referans Şema



Şekil 4.28.- 16-Bitlik Ortak Yol

V. SONUÇ VE ÖNERİLER

Projenin tasarımında kullanılan Logisim programı açık kaynak ve grafik ara yüz desteği sayesinde mevcut fonksiyonların kolaylık uygulamaya dönüştürebilme özelliğinin olması sebebiyle özellikle tercih edildi. Ancak programa uzun süredir yazılım güncellemesi yapılmaması sebebiyle pek stabil çalışmamasından dolayı bu tür büyük projelerin çalışmasında problemlerin çıkma olasılığı yüksek ki bende de öyle oldu. Örneğin, bütün bileşenleri çizip uygulamayı kapatıp tekrar açtığımda kapıların girişlerindeki yolların bozulmalarına ya da simülasyon aşamasında hata vermesine çok kez şahit oldum. Bu yüzden, büyük projelerin tasarımının VHDL ya da Verilog gibi donanım tanımlama dilleriyle EK-1'den elde edilen transfer fonksiyonlarının da yardımıyla Quartus, Xilinx ISE gibi ortamlarda yazılıp FPGA kartları üzerinde gerçeklemesiyle daha doğru sonuçlar elde edileceği kanaatindeyim.

Bunun dışında, transfer fonksiyonları olduğu gibi aktardım ve simülasyon ortamı olduğu için net olarak gözlemleyemedim ancak mantık kapılarıyla yapılan sistemlerde fonksiyonlar düzenlenmezse sakıncalı durumların (Hazards) ve kapılardan kaynaklı olarak karşımıza çıkan yayılım gecikmesi gibi sorunların karşımıza çıkması muhtemel bir gerçektir. Bu yüzden mevcut projenin gerçeklenmesinde bunlara dikkat edilmesi ve FPGA üzerinde gerçeklenmesi ile daha kesin sonuçlar alınabilir.

EK- 1

TABLE 5-6 Control Functions and Microoperations for the Basic Computer

Fetch	$R'T_0$:	$AR \leftarrow PC$
	$R'T_1$:	$IR \leftarrow M[AR]$, $PC \leftarrow PC + 1$
Decode	$R'T_2$:	$D_0, \ldots, D_7 \leftarrow$ Decode $IR(12\text{–}14)$, $AR \leftarrow IR(0\text{–}11)$, $I \leftarrow IR(15)$
Indirect	D_7IT_3:	$AR \leftarrow M[AR]$
Interrupt:		
	$T_0'T_1'T_2'(IEN)(FGI + FGO)$:	$R \leftarrow 1$
	RT_0:	$AR \leftarrow 0$, $TR \leftarrow PC$
	RT_1:	$M[AR] \leftarrow TR$, $PC \leftarrow 0$
	RT_2:	$PC \leftarrow PC + 1$, $IEN \leftarrow 0$, $R \leftarrow 0$, $SC \leftarrow 0$
Memory-reference:		
AND	D_0T_4:	$DR \leftarrow M[AR]$
	D_0T_5:	$AC \leftarrow AC \wedge DR$, $SC \leftarrow 0$
ADD	D_1T_4:	$DR \leftarrow M[AR]$
	D_1T_5:	$AC \leftarrow AC + DR$, $E \leftarrow C_{out}$, $SC \leftarrow 0$
LDA	D_2T_4:	$DR \leftarrow M[AR]$
	D_2T_5:	$AC \leftarrow DR$, $SC \leftarrow 0$
STA	D_3T_4:	$M[AR] \leftarrow AC$, $SC \leftarrow 0$
BUN	D_4T_4:	$PC \leftarrow AR$, $SC \leftarrow 0$
BSA	D_5T_4:	$M[AR] \leftarrow PC$, $AR \leftarrow AR + 1$
	D_5T_5:	$PC \leftarrow AR$, $SC \leftarrow 0$
ISZ	D_6T_4:	$DR \leftarrow M[AR]$
	D_6T_5:	$DR \leftarrow DR + 1$
	D_6T_6:	$M[AR] \leftarrow DR$, if $(DR = 0)$ then $(PC \leftarrow PC + 1)$, $SC \leftarrow 0$
Register-reference:		
		$D_7I'T_3 = r$ (common to all register-reference instructions)
		$IR(i) = B_i$ $(i = 0, 1, 2, \ldots, 11)$
	r:	$SC \leftarrow 0$
CLA	rB_{11}:	$AC \leftarrow 0$
CLE	rB_{10}:	$E \leftarrow 0$
CMA	rB_9:	$AC \leftarrow \overline{AC}$
CME	rB_8:	$E \leftarrow \overline{E}$
CIR	rB_7:	$AC \leftarrow$ shr AC, $AC(15) \leftarrow E$, $E \leftarrow AC(0)$
CIL	rB_6:	$AC \leftarrow$ shl AC, $AC(0) \leftarrow E$, $E \leftarrow AC(15)$
INC	rB_5:	$AC \leftarrow AC + 1$
SPA	rB_4:	If $(AC(15) = 0)$ then $(PC \leftarrow PC + 1)$
SNA	rB_3:	If $(AC(15) = 1)$ then $(PC \leftarrow PC + 1)$
SZA	rB_2:	If $(AC = 0)$ then $PC \leftarrow PC + 1$
SZE	rB_1:	If $(E = 0)$ then $(PC \leftarrow PC + 1)$
HLT	rB_0:	$S \leftarrow 0$
Input-output:		
		$D_7IT_3 = p$ (common to all input–output instructions)
		$IR(i) = B_i$ $(i = 6, 7, 8, 9, 10, 11)$
	p:	$SC \leftarrow 0$
INP	pB_{11}:	$AC(0\text{–}7) \leftarrow INPR$, $FGI \leftarrow 0$
OUT	pB_{10}:	$OUTR \leftarrow AC(0\text{–}7)$, $FGO \leftarrow 0$
SKI	pB_9:	If $(FGI = 1)$ then $(PC \leftarrow PC + 1)$
SKO	pB_8:	If $(FGO = 1)$ then $(PC \leftarrow PC + 1)$
ION	pB_7:	$IEN \leftarrow 1$
IOF	pB_6:	$IEN \leftarrow 0$

KAYNAKLAR

[1] M. Morris MANO, *Computer System Architecture* 3rd Edition, Prentice Hall US Edition, 1992

[2] M. Morris MANO, *Sayısal Tasarım,* Çev. S BOĞOSYAN, M GÖKAŞAN, S KURTULAN. İstanbul: Literatür Yayınları, 2018

[3] Ercan ERKALKAN, Bilgisayar Donanımı Dersi Basılmamış Ders Notları

[4] Hendrix College, "Carl BURCH, PhD" 2020

https://www.hendrix.edu/research/default.aspx?id=55783

[5] cburch, "What is Logisim?", 2020

http://www.cburch.com/logisim/

ÖZGEÇMİŞ

Adı Soyadı: Fatih YILDIRIM

Sürekli Adresi: Deveci Mahallesi Yamaç Sokak No:16/3 Mut/Mersin

Doğum Yeri ve Yılı: Mersin, 1996

Yabancı Dili: İngilizce

İlk Öğretim: Mut Cumhuriyet İlköğretim Okulu, 2010

Orta Öğretim: Mut Anadolu Lisesi, 2014

Lisans: Karadeniz Teknik Üniversitesi, 2015-

Bölüm: Elektrik-Elektronik Mühendisliği (%30 İngilizce)

Ön Lisans: Marmara Üniversitesi, 2016-

Program Adı: Bilgisayar Programcılığı (Uzaktan Öğretim)

Çalışma Hayatı: (Staj)

Seyisco Bilişim Elektronik Dan. Eğit. San. Ve Tic. A.Ş. / 07/2018- 08/2018